Министерство образования Республики Беларусь
Белорусский государственный университет
Механико-математический факультет
Кафедры веб-технологий и компьютерного моделирования

А. С. Кравчук, А. И. Кравчук, Е. В. Кремень

Язык C++.
Сборник тематических заданий и примеров программ

Учебные материалы
для студентов специальности 6-05-0533-07
«Математика и компьютерные науки
(по профилизациям)»

Минск
2024

УДК 004.432.045C++(075.8)
К 772

Решение о депонировании вынес:
Совет механико-математического факультета
Протокол № 6 от 30 января 2024 г.

Авторы:
А. С. Кравчук, доктор физико-математических наук профессор кафедры экономической информатики БГЭУ,
А. И. Кравчук, кандидат физико-математических наук доцент кафедры веб-технологий и компьютерного моделирования БГУ,
Е. В. Кремень, кандидат физико-математических наук доцент кафедры веб-технологий и компьютерного моделирования БГУ.

Р е ц е н з е н т ы
Кафедра информационных технологий Белорусского государственного экономического университета (заведующая кафедрой Садовская М.Н., кандидат технических наук, доцент);
Медведев С.В., заведующий лабораторией синтеза технических систем Объединенного института проблем информатики НАНБ, доктор технических наук.

Кравчук, А. С. Язык C++. Сборник тематических заданий и примеров программ: учебные материалы для студентов специальности: 6-05-0533-07 «Математика и компьютерные науки (по направлениям)» / А. С. Кравчук, А. И. Кравчук, Е. В. Кремень. – Минск : БГУ, 2024. – 147 с. : табл. – Библиогр.: с. 147.

Сборник заданий предназначен для проработки приемов как императивного, так и объектно-ориентированного программирования на языке C++. Издание содержит задачи для обучения студентов решению задач разветвляющихся, циклических алгоритмов, использованию функций, решению задач с использованием простейших численных методов, а также обработке массивов, применению принципов объектно-ориентированного программирования, наследованию. Достаточно большой раздел посвящен описанию основ работы с STL: рассмотрены некоторые контейнеры, правила создания и основы работы с функторами. В каждой теме приводятся примеры решения типовых задач и варианты индивидуальные заданий. Издание ориентировано в первую очередь на тех, кто не имеет опыта практического программирования на языке C++ и адресуется студентам, а также всем, кто хотел бы научиться приемам программирования при решении стандартных задач.

ОГЛАВЛЕНИЕ

СОЗДАНИЕ И ИСПОЛЬЗОВАНИЕ ФУНКЦИЙ

Общие требования к выполнению тематических заданий в данном разделе. Каждый алгоритм, решающий поставленную задачу, должен быть оформлен в виде функции. Если по каким-то причинам студент ***не*** выполняет этого требования, но демонстрирует работоспособную программу, написанную целиком в `main()`, то оценка в табеле по соответствующему тематическому заданию снижается на `20%`.

Программирование разветвляющихся алгоритмов

Задание по теме состоит из ***трех*** частей:

- первая часть посвящена применению функций для проверки условия в операторе `if()`, расположенного в `main()`;
- вторая часть посвящена использованию `if()` в теле функции;
- третья часть посвящена использованию `enum` для формирования значения, возвращаемого функцией, а также ее применения при проверке условия оператора выбора `switch()`.

Часть 1. Разработка функции, возвращающей булево значение

Обобщенная формулировка задания части 1. Выдать соответствующее сообщение о принадлежности заданной точки замкнутой области (Таблица 1).

Выполнение задания заключается во внесении измененийй в предлагаемый пример:

- если необходимо следует использовать дополнительные глобальные константы, соответствующих определенной области;
- изменить названия функции в соответствии с заданием;
- изменить выражение в операторе `return` в соответствии с заданной в индивидуальном задании геометрией области.

Пример задания и его выполнения

Формулировка примера задания. Определить, попадает ли точка в замкнутый круг заданного радиуса (глобальная константа `RADIUS`), если центр круга лежит в начале координат, а точка задаются своими координатами (Рисунок 1).

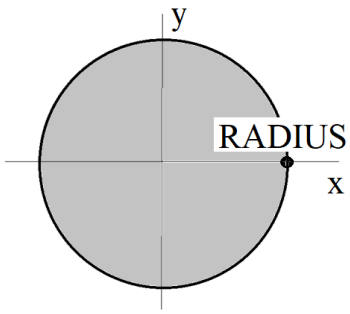


Рисунок 1 – Заданная область

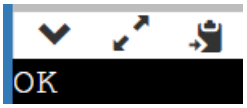
Программа.

main.cpp

```
#include <iostream>
#include <cmath>
using namespace std;

//объявление и инициализация глобальной константы
const float RADIUS = 1.5;

//прототип (объявление) функции
bool belongsCircle(float, float);

int main() {
    // инициализация значений координат точки
    float xPoint = 1, yPoint = 1;
    // проверка условия попадания точки в область и
    // вывод на экран результатов проверки
    if ( belongsCircle(xPoint, yPoint) ) {
        cout<< "OK" <<endl;
    }
    else {
        cout<<"Missing"<< endl;
    }
    return 0;
}
```

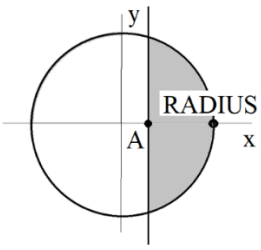
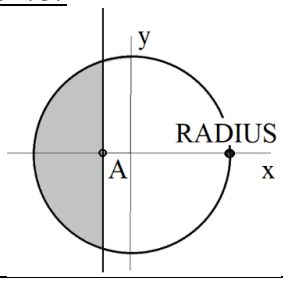
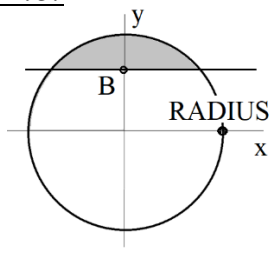
```
24
25  // определение функции
26  bool belongsCircle(float x,float y) {
27      return x * x + y * y <= pow(RADIUS,2);
28  }
```

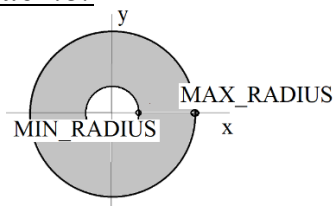
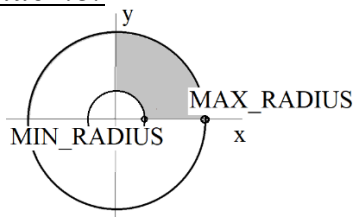
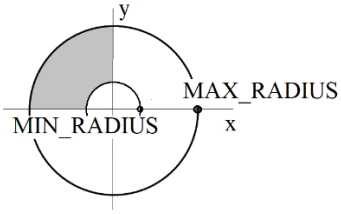
Результаты работы программы:

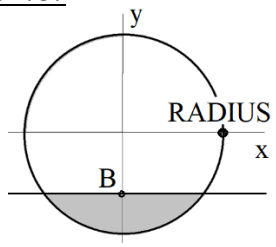
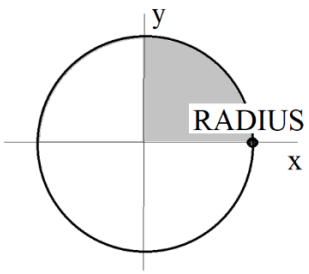
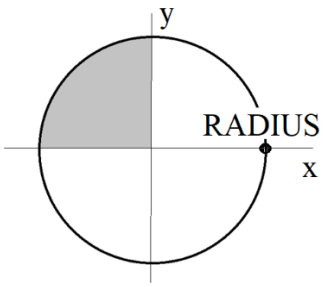
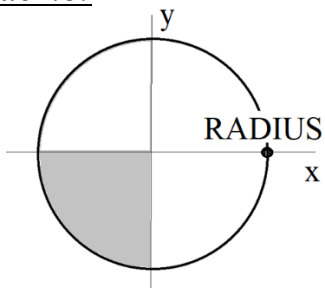


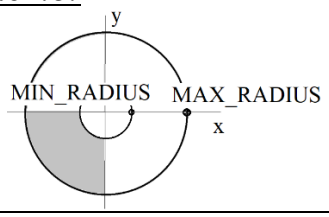
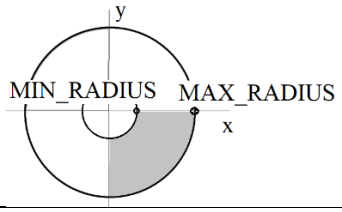
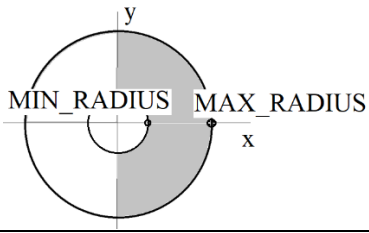
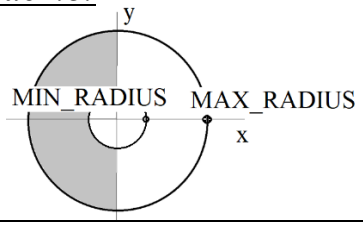
Варианты индивидуальных заданий к части 1.

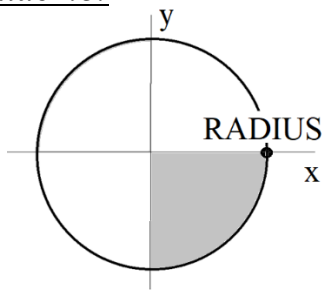
Таблица 1 – Круговые области для индивидуальных заданий

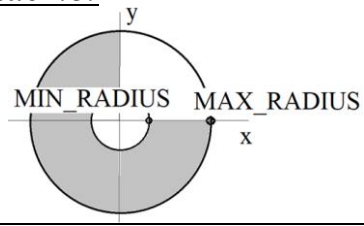
№	Задание
1.	Задать две константы `A` и `RADIUS`. *Область:* 
2.	Задать две константы `A` и `RADIUS`. *Область:* 
3.	Задать две константы `B` и `RADIUS`. *Область:* 

№	Задание
4.	Задать две константы `MIN_RADIUS` и `MAX_RADIUS`. *Область:* 
5.	Задать две константы `MIN_RADIUS` и `MAX_RADIUS`. *Область:* 
6.	Задать две константы `MIN_RADIUS` и `MAX_RADIUS`. *Область:* 

№	Задание
7.	Задать две константы `B` и `RADIUS`. *Область:* 
8.	Задать константу `RADIUS`. *Область:* 
9.	Задать константу `RADIUS`. *Область:* 
10.	Задать константу `RADIUS`. *Область:* 

№	Задание
11.	Задать две константы `MIN_RADIUS` и `MAX_RADIUS`. *Область:* 
12.	Задать две константы `MIN_RADIUS` и `MAX_RADIUS`. *Область:* 
13.	Задать две константы `MIN_RADIUS` и `MAX_RADIUS`. *Область:* 
14.	Задать две константы `MIN_RADIUS` и `MAX_RADIUS`. *Область:* 

№	Задание
15.	Задать константу RADIUS. *Область:* 

№	Задание
16.	Задать две константы MIN_RADIUS и MAX_RADIUS. *Область:* 

Часть 2. Программирование функций, содержащих оператор if () в своем теле

Пример задания и его выполнения

Формулировка примера задания. Ввести число x, выяснить что больше: целая часть числа x, или его дробная часть, умноженная на 10. Если первое, то число разделить на 10, если второе, то дробную часть от него отнять.

Программа.

main.cpp

```
#include <iostream>
using namespace std;

double Operation(double x);

int main() {
    double x ;
    cout << "Input x " << endl;
    cin >> x;
    cout << "\nResult of calculation = "
         << Operation(x)
         << endl;
    return 0;
}
```

```
15
16  double Operation(double x) {
17      double fractPart, result;
18      int intPart;
19      // определеяем целую часть
20      intPart = (int) x;
21      // дробная часть
22      fractPart = x - (int) x;
23      // проверка условия
24      if ((double)intPart > fractPart *10) {
25          //целая часть больше
26          result = x / 10;
27      }
28      else {
29          //больше умноженная на 10 дробная часть
30          result = x - fractPart;
31      }
32      return result;
33  }
```

Результаты работы программы:

```
Input x
4.7

Result of calculation = 4
```

Варианты индивидуальных заданий к части 2.

Указание: *при определении делимости использовать операцию определения остатка от целочисленного деления (%).*

1. Ввести три целые числа a, b, c. Вывести сумму чисел $a + b + c$, определить нечетная ли она, если нечетная, то найти значение произведения $a \cdot b \cdot c$.
2. Ввести целые числа a, b. Определить делится ли целое число a на целое число b без остатка. Если делится, вывести на экран их произведение $a \cdot b$.

3. Ввести целое число a получить число x равное:
   - половине a, если a – нечетное;
   - утроенному значению a, если a – четное.
4. Ввести целые числа a и b и действительное число z. Преобразовать число z по формуле $z \cdot x$, если a делиться на b без остатка и z/x в противном случае (где x – остаток от деления a на b).
5. Ввести натуральное m. Если 2^m больше, чем $8 \cdot m$, то m увеличить на 20, в противном случае m уменьшить в три раза.
6. Ввести неравные целые числа $m,\ n$. Определить какая из дробей m/n или n/m ближе к числу π . В первом случае число m умножить на 10, во втором удвоить число n. (*использовать определенную пользователем константу* `const double PI = 3.141592653589793;`)
7. Определить делится ли на 7 введенное натуральное число a . Если делится, то другому числу y присвоить 200, в противном случае y сделать равным 500.
8. Ввести целые числа $m,\ n,\ k$. Для двух дробей m/n и n/k выяснить равны ли их дробные части. Если дробные части равны, то вывести это значение, если нет, то ограничиться выводом соответствующего сообщения.
9. Ввести целые числа $m,\ n,\ l,\ k$. Для двух дробей m/n и и l/k выяснить равны ли их дробные части. Если части равны, то вывести один раз значение дробной части, если не равны, то вывести произведение обеих дробных частей, умноженное на 2.
10. Ввести целые числа $m,\ n$. Для дроби m/n определить превышает ли дробная часть числа 0.7 . Если превышает, то число m разделить на 2 , иначе удвоить число n .
11. Ввести целые числа $m,\ n$. Определить превышает ли умноженная на 20 дробная часть числа m/n целую часть m/n. Если превышает, то вывести значение на сколько.
12. Ввести целые числа $m,\ n$. Определить является ли целая часть числа m/n нечетной. Если она нечетная, то m возвести в квадрат, иначе n утроить.
13. Ввести целые числа $m,\ n$. Возвести во вторую степень число m/n, если его целая часть больше числа k , где k остаток от деления m на 8.
14. Ввести три числа x , y , z , определить их сумму. Если сумма больше или равна 20, то оставить числа без изменения, в противном случае определить число s - сколько не хватает в сумме, чтобы она была равна 120.
15. Ввести число x. Выяснить превышает ли модуль числа $cos(x^2)$ значение $\sqrt{2}\ /\ 2$. Если превышает, то x уменьшить втрое, в противном случае x удвоить.

Часть 3. Использованию `enum` для формирования значения, возвращаемого функцией, а также его применения при проверке условия оператором выбора `switch()`

Пример задания и его выполнения

Формулировка примера задания. Даны два вещественных числа a и x. Используя функцию определяющую принадлежность переменной x к определенному интервалу, вывести значение y, вычисленное по формуле:

$$y = \begin{cases} \sin(a \cdot x), -1 \leq a \leq 1 \text{ и } -1 \leq x \leq 1, \\ \cos(a \cdot x), a < -1 \text{ и } 1 < a \text{ и } x < -1 \text{ и } 1 < x. \end{cases}$$

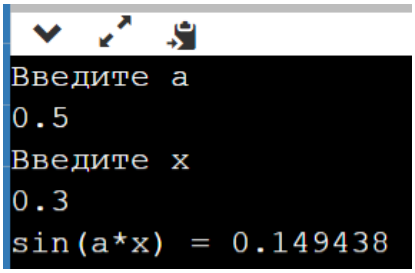
Программа.

main.cpp

```
#include <iostream>
#include <cmath>
using namespace std;

enum set {
    NOTHING,
    FIRST_SET,
    SECOND_SET
};

set detectorSet(double aValue, double xValue) {
    if (-1<=aValue && aValue<=1 && -1<=xValue && xValue<=
                                    return FIRST_SET;
    if (aValue<-1 && 1<aValue && xValue<-1 && 1<xValue)
                                    return SECOND_SET;
    return NOTHING;
}

int main() {
    cout << "Введите a"<< endl;
    double a;
    cin >> a;
    cout << "Введите x"<< endl;
```

```
24      double x;
25      cin >> x;
26      switch( detectorSet(a, x) ) {
27          case FIRST_SET: cout<<"sin(a*x) = "<< sin(a * x)
28                                  << endl;
29                          break;
30          case SECOND_SET: cout<<"cos(a*x) = "<< cos(a * x)
31                                  << endl;
32          default: cout<< "ошибка ввода"<< endl;
33      }
34      return 0;
35  }
```

Результаты работы программы:



Варианты индивидуальных заданий к части 3

С клавиатуры вводятся значения a и x. Составить программу для вычисления y:

1. $y = \begin{cases} a \cdot sin^2(x) \text{ при } -5 \le a \cdot x \le 0 \text{ или } 3 \le a \cdot x \le 5, \\ a \cdot cos(x^2) \text{ при } 0 < a \cdot x < 3, \\ ln(1 + |a \cdot x|) \text{ при } a \cdot x < -5 \text{ или } a \cdot x > 5. \end{cases}$

2. $y = \begin{cases} \sqrt{a \cdot cos^2(x)} \text{ при } -4 \le a \cdot x \le -2 \text{ или } 0 \le a \cdot x \le 4, \\ a^2 \cdot cos(x^2) + 1 \text{ при } -2 < a \cdot x < 0, \\ ln(1 + |a^2 + x|) \text{ при } a \cdot x < -4 \text{ или } a \cdot x > 4. \end{cases}$

3. $y = \begin{cases} \sqrt{|a + tg(a^2 \cdot x)|} \text{ при } -10 \le a \cdot x \le -5 \text{ или } 0 \le a \cdot x \le 5, \\ 1 - a \cdot cos(x^3) \text{ при } -5 < a \cdot x < 0, \\ a + ln^4(1 + |x|) \text{ при } a \cdot x < -10 \text{ или } a \cdot x > 5. \end{cases}$

4. $y = \begin{cases} \sqrt{|a^2 + ctg(a \cdot x)|} \text{ при } -10 \le a \cdot x \le -7 \text{ или } 0 \le a \cdot x \le 7, \\ 1 + a \cdot \sqrt{|cos(x^3)|} \text{ при } -7 < a \cdot x < 0, \\ a + tg^4(|a \cdot x|) \text{ при } a \cdot x < -7 \text{ или } a \cdot x > 7. \end{cases}$

5. $y = \begin{cases} a + sin^2(x) \text{ при } -7 \le a \cdot x \le 0 \text{ или } 3 \le a \cdot x \le 7, \\ a \cdot \sqrt{|cos(x^2)|} \text{ при } 0 < a \cdot x < 3, \\ ln^4(1 + |a \cdot x|) \text{ при } a \cdot x < -7 \text{ или } a \cdot x > 7. \end{cases}$

6. $y = \begin{cases} \sqrt{a + |tg(x^2)|} \text{ при } -5 \le a \cdot x \le 0 \text{ или } 3 \le a \cdot x \le 5, \\ ln(|a + x|) \text{ при } 0 < a \cdot x < 3, \\ a^2 + x^3 - 5 \text{ при } a \cdot x < -5 \text{ или } a \cdot x > 5. \end{cases}$

7. $y = \begin{cases} \sqrt{a^2 \cdot |cos(x)|} \text{ при } -8 \le a \cdot x \le -3 \text{ или } 0 \le a \cdot x \le 4, \\ a^2 \cdot cos(x^2) + 1 \text{ при } -3 < a \cdot x < 0, \\ ln(|a^2 + x|) \text{ при } a \cdot x < -8 \text{ или } a \cdot x > 4. \end{cases}$

8. $y = \begin{cases} \sqrt{|a + sin(a^2 \cdot x)|} \text{ при } -15 \le a \cdot x \le -5 \text{ или } 0 \le a \cdot x \le 15, \\ 1 + a \cdot ctg(x^3) \text{ при } -5 < a \cdot x < 0, \\ a^3 + ln(1 + |x|) \text{ при } a \cdot x < -15 \text{ или } a \cdot x > 15. \end{cases}$

9. $y = \begin{cases} \sqrt{|a^3 + x|} \text{ при } -6 \le a \cdot x \le -3 \text{ или } 0 \le a \cdot x \le 12, \\ a^2 \cdot tg(x^2) + 1 \text{ при } -3 < a \cdot x < 0, \\ ln(|a^2 + x|) \text{ при } a \cdot x < -4 \text{ или } a \cdot x > 12. \end{cases}$

10. $y = \begin{cases} \sqrt{|a^3|} + tg(a \cdot x) \text{ при } -6.8 \le a \cdot x \le -1.3 \text{ или } 0 \le a \cdot x \le 5.4\,, \\ a^2 + cos^2(x^2) + 1 \text{ при } -1.3 < a \cdot x < 0, \\ ln(|a^2 + x^3|) \text{ при } a \cdot x < -6.8 \text{ или } a \cdot x > 5.4\,. \end{cases}$

11. $y = \begin{cases} \sqrt{|sin(a^2 \cdot x)|} \text{ при } -10 \le a \cdot x \le -7 \text{ или } 0 \le a \cdot x \le 5.4\,, \\ a \cdot \sqrt{|cos(x^3)|} \text{ при } -7 < a \cdot x < 0, \\ a + ln(|a \cdot x|) \text{ при } a \cdot x < -10 \text{ или } a \cdot x > 5.4\,. \end{cases}$

12. $y = \begin{cases} \sqrt{|a^2 + ln(a \cdot x)|} \text{ при } -15 \le a \cdot x \le -7 \text{ или } 0 \le a \cdot x \le 10\,, \\ x + \sqrt{|sin(a^3)|} \text{ при } -7 < a \cdot x < 0, \\ a + tg^4(a \cdot x) \text{ при } a \cdot x < -15 \text{ или } a \cdot x > 10\,. \end{cases}$

$$13.y = \begin{cases} \sqrt{|ln(a^2 + x^3)|} \text{ при } -2 \le a \cdot x \le 2 \text{ или } 4 \le a \cdot x \le 6\,, \\ x^3 \cdot exp(a \cdot x) \text{ при } 2 < a \cdot x < 4, \\ \sqrt{|a \cdot (1 - x)|} \text{ при } a \cdot x < -2 \text{ или } a \cdot x > 6\,. \end{cases}$$

$$14.y = \begin{cases} \sqrt{x^2 + |cos(a \cdot x)|} \text{ при } -11 \le a \cdot x \le -5 \text{ или } 0 \le a \cdot x \le 5\,, \\ 1 + a \cdot \sqrt{|ctg(x^3)|} \text{ при } -5 < a \cdot x < 0, \\ a + ln|a \cdot x| \text{ при } a \cdot x < -11 \text{ или } a \cdot x > 5\,. \end{cases}$$

$$15.y = \begin{cases} \sqrt{1 + |a \cdot tg(x^2)|} \text{ при } -15 \le a \cdot x \le 0 \text{ или } 3 \le a \cdot x \le 5\,, \\ ln|a + x| \text{ при } 0 < a \cdot x < 3, \\ a^2 + x^2 - 5 \text{ при } a \cdot x < -15 \text{ или } a \cdot x > 5\,. \end{cases}$$

Циклические алгоритмы

Задание по теме состоит из ***двух*** частей:

- первая часть посвящена вызову функций внутри операторов циклов, расположенных в `main()`;
- вторая часть посвящена использованию операторов циклов в теле функций.

Часть 1. Вызов функции в операторах циклов

Пример задания и его выполнения

Формулировка примера задания. Создать функцию $f(x)$ в соответствии с заданием (например, $f(x) = (x + 1)^2$). В `main()` протабулировать ее значения на интервале $[0, 3]$ с шагом 0.2, используя три различных оператора цикла `for` (на интервале $[0, 1[$), `while` (на интервале $[1, 2[$), `do while` (на интервале $[2, 3]$), указав при этом значения функции с нечетной и четной целой частью.

Для вывода строки с информацией использовать дополнительную функцию `PrintValue()`.

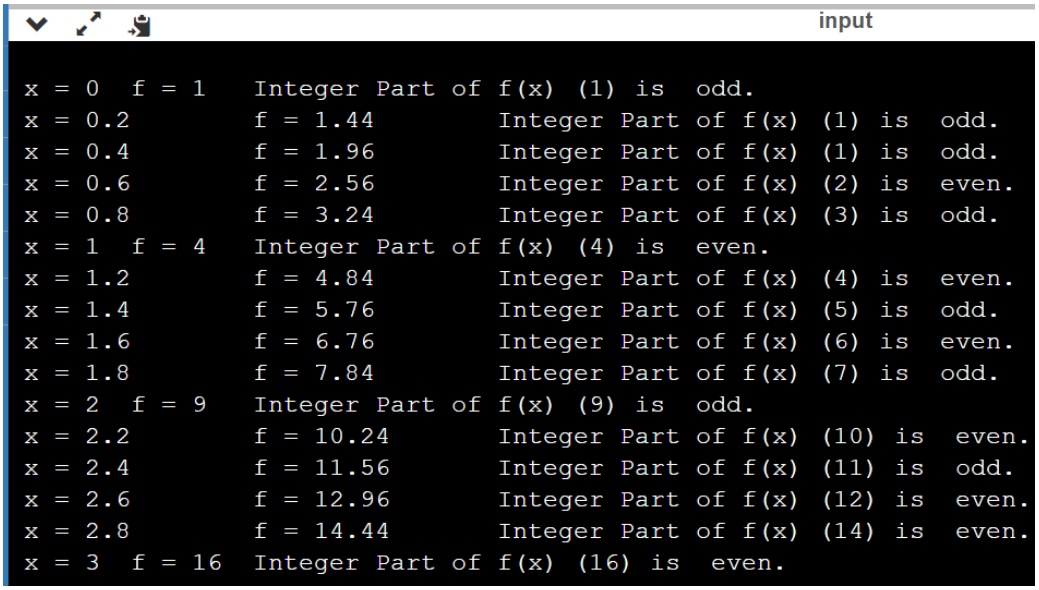
Программа.

main.cpp

```
1  #include <iostream>
```

```cpp
#include <cmath>
using namespace std;

double userFunction (double x) {
    return (x + 1) * (x + 1);
}

void printValue(double x) {
    cout << "\n x = "<<x<<"\tf = "
         << userFunction(x)
         << "\tInteger Part of f(x) ("
         << (int) userFunction(x) <<") is ";
    // определение четности или нечетности
    // целой части
    if ( ((int) userFunction(x)) % 2 != 0 ) {
        cout<<" odd.";
    }
    else {
        cout<<" even.";
    }
}

int main() {
    const double tabStep = 0.2;
    double x;
    for(x = 0; x < 1; x += tabStep) {
        printValue(x);
    }
    while( x < 2 ) {
        printValue(x);
        x += tabStep;
    }
    do {
        printValue(x);
        x += tabStep;
    }
    while( x < 3 + tabStep/2);
    return 0;
}
```

Результат работы программы:



Варианты индивидуальных заданий

Используя три различных цикла, протабулировать функции (Таблица 2).

Таблица 2 – Функции для табулирования

N	$f(x)$
1.	$\frac{2 \cdot x + 1}{x^3 + 2 \cdot x^2 + x + 2}$
2.	$\sqrt{3^{-2 \cdot x^2} + 2^{4 \cdot x}}$
3.	$4 \cdot \frac{sin(x^3 + 2 \cdot x^2 + x + 2)}{x^2 + 2}$
4.	$\begin{cases} (x+4)^2, \text{при } \lvert sin(x)\rvert > 0.5 \\ x, \quad \text{при } \lvert sin(x)\rvert \le 0.5 \end{cases}$
5.	$\begin{cases} \sqrt{\lvert x\rvert} + 4, \text{при } \lvert cos(x)\rvert > 0.5 \\ x - 1, \quad \text{при } \lvert cos(x)\rvert \le 0.5 \end{cases}$
6.	$sin\big(4 \cdot tg(x)\big)$
7.	$sin^2\big(ctg(\lvert x\rvert + 2)\big)$
8.	$tg(\lvert x^5\rvert + 2)$

N	$f(x)$
9.	$\sqrt{\lvert 2^{3 \cdot x} + 5 \cdot x^2 + x - 1\rvert};$
10.	$\lvert sin(x^2 + 3)\rvert$
11.	$e^{3 \cdot x^2 + \lvert sin(x)\rvert};$
12.	$sin\left(\frac{x^4 + x + 7}{x + 5}\right)$
13.	$\sqrt{e^{x^2} + e^{-x^2}}$
14.	$\sqrt[4]{\left\lvert \frac{2}{(1+x)^3} \right\rvert}$
15.	$\frac{3^{sin(x)}}{\lvert x^3\rvert + 1}$
16.	$ln\lvert sin(x) + 2 \cdot x\rvert$

Часть 2. Подсчет в теле функции количества значений, удовлетворяющих признаку

Пример задания и его выполнения

Формулировка примера задания. Ввести два натуральных числа `a` и `b` (`0 < a < b`). Трижды определить сколько патрульных из диапазона `[a, b]` являются четвертой степенью какого-либо натурального числа с помощью трех различных функций, использующих по одному оператору цикла (`for()`, `while()`, `do while()`).

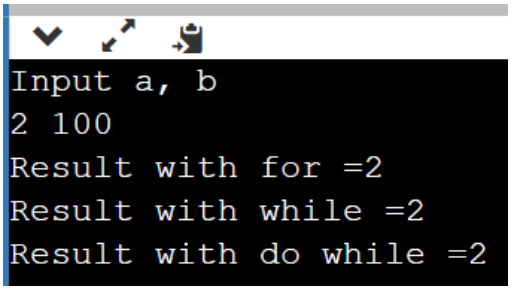
Программа.

main.cpp

```cpp
#include <iostream>
#include <cmath>
using namespace std;

//прототипы
int forNumber(int, int);
int whileNumber(int, int);
int doNumber(int, int);

int main() {
    int a, b;

    while(true) {
        cout << "Input a, b"<<endl;
        cin >> a >> b;
        if (0 < a && a < b)break;
        cout << "\nIncorrect input values "
             << endl;
    }
    cout << "Result with for ="
         << forNumber(a, b) << endl
         << "Result with while ="
         << whileNumber(a, b) << endl
         << "Result with do while ="
         << doNumber(a, b) << endl;
```

```
26      return 0;
27  }
28
29  //описания функций
30  int forNumber(int a, int b) {
31      int n = 0;
32      double x;
33      for (int i = a; i <= b; i++) {
34          x = pow(i, 0.25);
35          n = n + ((x - (int) x) == 0);
36      }
37      return n;
38  }
39
40  int whileNumber(int a, int b) {
41      int n = 0;
42      double x;
43      int i = a;
44      while(i <=b ) {
45          x = pow(i, 0.25);
46          n = n + ((x - (int) x) == 0);
47          i++;
48      }
49      return n;
50  }
51
52  int doNumber(int a, int b) {
53      int n = 0;
54      double x;
55      int i = a;
56      do {
57          x = pow(i, 0.25);
58          n = n + ((x - (int) x) == 0);
59          i++;
60      }
61      while(i <= b);
62      return n;
63  }
```

Результат работы программы:



Варианты индивидуальных заданий

Ввести два положительных целых числа `a` и `b` (`0 < a < b`). Трижды (различными функциями) определить для диапазона `[a, b]` количество:

1. ***четных*** целых чисел;
2. целых чисел, являющихся ***квадратом*** натурального числа;
3. ***нечетных*** целых чисел;
4. натуральных чисел, которые ***нацело делят*** целое число $b - a$;
5. натуральных чисел, являющихся ***квадратом нечетного*** целого числа;
6. натуральных чисел, ***больших*** $\sqrt{b-a}$;
7. целых чисел, являющихся ***кубом нечетного*** целого числа;
8. целых чисел, попадающих в диапазон $\left[0, \frac{1}{2} ln(b-a)\right]$;
9. натуральных чисел, ***меньших*** $\sqrt{b-a}$;
10. натуральных чисел, являющихся ***квадратом четного*** целого числа;
11. натуральных чисел $i \in [a, b]$ $(0 < a,\ 0 < b)$ таких что $\left|ln\left(sin\left(\frac{1}{i}\right)\right)\right| \in [a, b]$;
12. целых чисел, являющихся ***кубом*** целого числа;
13. целых чисел, ***меньших*** $\sqrt[3]{b-a}$;
14. целых чисел, являющихся ***кубом четного*** целого числа;
15. целых чисел, ***больших*** $\sqrt[3]{b-a}$;

Вычисление сумм

Пример задания и его выполнения

Формулировка примера задания. Ввести ***положительные*** числа x (с плавающей точкой) и n (натуральное). Выполнить проверку корректности

введенных параметров (их неотрицательности). Трижды определить значение выражения:

$$S(x,n) = \sum_{i=1}^{n} \frac{i^2}{\sqrt{x+i}},$$

используя в трех различных функциях по одному оператору цикла (`for, while, do while`).

Программа.

main.cpp

```
#include <iostream>
#include <cmath>
using namespace std;

//прототипы
double forSum(double, int);
double whileSum(double, int);
double doSum(double, int);

int main() {
    double x;
    int n;
    while(true) {
        cout << "Input x, n"<<endl;
        cin >> x >> n;
        if (0 < x && 0 < n)break;
        cout <<"\nIncorrect input values "
             << endl;
    }
    cout << "Result with for ="
         << forSum(x, n) << endl
         << "Result with while ="
         << whileSum(x, n) << endl
         << "Result with do while ="
         << doSum(x, n) << endl;
        return 0;
}
```

```
28
29  //описания функций
30  double forSum(double x, int n) {
31      double s = 0;
32      for (int i=1; i<=n; i++) {
33          s = s + i * i / sqrt(i + x);
34      }
35      return s;
36  }
37  double whileSum(double x, int n) {
38      double s=0;
39      int i=1;
40      while(i <= n) {
41      s = s + i * i / sqrt(i + x);
42      i++;
43      }
44      return s;
45  }
46  double doSum(double x, int n) {
47      double s = 0;
48      int i = 1;
49      do {
50          s = s + i * i / sqrt(i + x);
51          i++;
52      }
53      while(i <= n);
54      return s;
55  }
```

Результат работы программы:

```
Input x, n
0.5 5
Result with for =26.3595
Result with while =26.3595
Result with do while =26.3595
```

Варианты индивидуальных заданий

Часть 1

С помощью трех функций, каждая из которых использует один оператор цикла, определить значение суммы (Таблица 3) для ***не***отрицательных чисел `x` (`double`) и `n` (`int`).

Таблица 3 – Индивидуальные задания по вычислению сумм

N	Формула
1.	$S(n) = \sum_{i=2}^{n} ln(i);$
2.	$S(x,n) = \sum_{i=1}^{n} (i^2 + x);$
3.	$S(n) = \sum_{i=1}^{n} (i+1)^2;$
4.	$S(x,n) = \sum_{i=1}^{n} (i + x);$
5.	$S(x,n) = \sum_{i=1}^{n} (i^2 + x);$
6.	$S(x,n) = \sum_{i=1}^{n} (i^3 + i \cdot x);$
7.	$S(x,n) = \sum_{i=1}^{n} \left(\frac{i}{2} + x\right);$
8.	$S(x,n) = \sum_{i=1}^{n} \left(\frac{1}{\sqrt{2 \cdot i}} + x\right);$

N	Формула
9.	$S(x,n) = \sum_{i=1}^{n} (x \cdot i - 1)^3;$
10.	$S(x,n) = \sum_{i=1}^{n} \left(2 \cdot x + \sqrt{i}\right);$
11.	$S(x,n) = \sum_{i=1}^{n} \left(i + \sqrt{2 \cdot \lvert x \rvert}\right);$
12.	$S(x,n) = \sum_{i=1}^{n} \frac{i+x}{(i+1)^2};$
13.	$S(x,n) = \sum_{i=1}^{n} \left(x + \sqrt{\frac{1}{i}}\right);$
14.	$S(x,n) = \sum_{i=1}^{n} \frac{2}{(i+x)^3};$
15.	$S(x,n) = \sum_{i=1}^{n} ln\lvert i + 2 \cdot x \rvert.$
16.	$S(n) = \sum_{i=1}^{n} \frac{3^{i+1}}{i^3}$

Часть 2

Дано натуральное n. Используя три различных цикла с помощью трех функций, вычислить значение суммы S:

1. $S = 1 + 2 + \cdots + n;$
2. $S = \frac{1}{1} + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n};$
3. $S = 1^2 + 2^2 + \cdots + n^2;$
4. $S = \frac{1}{1^3} + \frac{1}{2^3} + \frac{1}{3^3} + \cdots + \frac{1}{n^3}$
5. $S = 1 + 3 + 5 + \cdots + n,$ где n – ***нечетное*** число;
6. $S = \frac{1}{1^3} + \frac{1}{3^3} + \frac{1}{5^3} + \cdots + \frac{1}{n^3},$ где n – ***нечетное*** число;
7. $S = 2 + 4 + 6 + \cdots + n,$ где n – ***четное*** число;
8. $S = \frac{1}{2^5} + \frac{1}{4^5} + \frac{1}{6^5} + \cdots + \frac{1}{n^5},$ где n – ***четное*** число;

9. $S = 1 \cdot 2 + 2 \cdot 3 + 3 \cdot 4 + \cdots + n \cdot (n+1)$;

10. $S = \frac{1}{1 \cdot 2} + \frac{1}{2 \cdot 3} + \frac{1}{3 \cdot 4} + \cdots + \frac{1}{n \cdot (n+1)}$;

11. $S = 1 \cdot 3 + 3 \cdot 5 + 5 \cdot 7 + \cdots + n \cdot (n+2)$, где n – ***нечетное*** число;
12. $S = \frac{1}{1 \cdot 3} + \frac{1}{3 \cdot 5} + \frac{1}{5 \cdot 7} + \cdots + \frac{1}{n \cdot (n+2)}$, где n – ***нечетное*** число;
13. $S = 2 \cdot 4 + 4 \cdot 6 + 6 \cdot 8 + \cdots + n \cdot (n+2)$, где n – ***четное*** число;
14. $S = \frac{1}{2 \cdot 4} + \frac{1}{4 \cdot 6} + \frac{1}{6 \cdot 8} + \cdots + \frac{1}{n \cdot (n+2)}$, где n – ***четное*** число;

15. $S = sin(1) + sin(2) + \cdots + sin(2 \cdot n + 1)$;
16. $S = \frac{1}{sin(1)} + \frac{1}{sin(2)} + \frac{1}{sin(3)} + \cdots + \frac{1}{sin(n)}$;

Вычисление произведений

Пример задания и его выполнения

Формулировка примера задания. Интерактивно задать положительные числа x (с плавающей точкой) и n (натуральное). Выполнить проверку корректности введенных параметров (их неотрицательности). Трижды определить значение выражения: $(x+1)^n$ используя его в трех различных функциях по одному оператору цикла (`for, while, do while`).

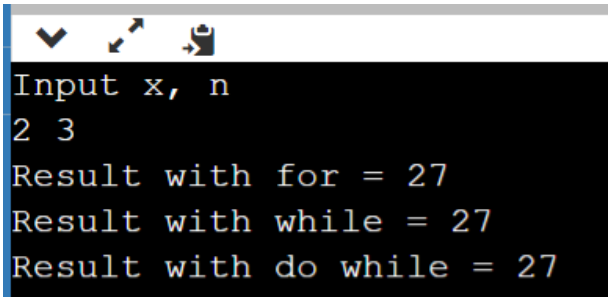
Программа.

main.cpp

```cpp
#include <iostream>
#include <cmath>
using namespace std;

//прототипы
double forProduct(double x, int n);
double whileProduct(double x, int n);
double doProduct(double x, int n);

int main() {
    double x;
    int n;
```

```
    while(true) {
        cout << "Input x, n"<<endl;
        cin >> x >> n;
        if (0 < x && 0 < n) break;
        cout<<"\nIncorrect input values "<< endl;
    }
    cout<< "Result with for = "<< forProduct(x, n)
        <<endl
        << "Result with while = "
        << whileProduct(x, n)
        << endl
        << "Result with do while = "
        << doProduct(x, n);
    return 0;
}

//описания функций
double forProduct(double x, int n) {
    double product = 1;
    for (int i=1; i<=n; i++) {
        product = product * (x + 1);
    }
    return product;
}
double whileProduct(double x, int n) {
    double product = 1;
    int i = 1;
    while( i <= n ) {
        product = product * (x + 1);
        i++;
    }
    return product;
}

double doProduct(double x, int n) {
    double product = 1;
    int i = 1;
    do {
        product = product * (x + 1);
        i++;
    }
    while( i <= n );
    return product;
}
```

Результаты работы программы:



Варианты индивидуальных заданий

Часть 1

1. написать функцию, вычисляющую (возвращающую значение) факториала натурального n ($n! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$), где n передается в функцию как параметр;
2. написать функцию, вычисляющую (возвращающую) x^n, где x – число с плавающей точкой, n – натуральное число. Функция получает x и n как параметр;
3. написать функцию, вычисляющую $(-1)^n \cdot n!$, где натуральное n передается в функцию как параметр;
4. написать функцию, вычисляющую $2^{2 \cdot n+1}$, где натуральное n передается в функцию как параметр;
5. написать функцию, вычисляющую (возвращающую значение) двойного факториала натурального n (если n нечетное, то $n!! = 1 \cdot 3 \cdot 5 \cdot \ldots \cdot n$, если n четное, то $n!! = 1 \cdot 2 \cdot 4 \cdot \ldots \cdot n$), где n передается в функцию как параметр;
6. написать функцию, вычисляющую (возвращающую) $\left(sin(x)\right)^{2 \cdot n}$, где x – число с плавающей точкой, n – натуральное число. Функция получает x и n как параметр;
7. написать функцию, вычисляющую $(-1)^n x^n$, где натуральное n и вещественное x передается в функцию как параметр;
8. написать функцию, вычисляющую $3^{2 \cdot n}$, где натуральное n передается в функцию как параметр;
9. написать функцию, вычисляющую (возвращающую значение) произведения $\left(1 + \frac{1}{1^2}\right) \cdot \left(1 + \frac{1}{2^2}\right) \cdot \ldots \cdot \left(1 + \frac{1}{n^2}\right)$ для заданного натурального n, передаваемого в функцию как параметр;

10. написать функцию, вычисляющую (возвращающую) $x^{3 \cdot n}$, где x – число с плавающей точкой, n – натуральное число. Функция получает x и n как параметр;
11. написать функцию, вычисляющую $(-1)^n x^{2 \cdot n}$, где натуральное n и вещественное x передается в функцию как параметр;
12. написать функцию, вычисляющую $7^{3 \cdot n}$, где натуральное n передается в функцию как параметр;
13. написать функцию, вычисляющую (возвращающую значение) факториал натурального $(2 \cdot n + 1)$ $((2 \cdot n + 1)! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (2 \cdot n + 1))$, где n передается в функцию как параметр;
14. написать функцию, вычисляющую (возвращающую) $(x + 5)^{2n+1}$, где x – число с плавающей точкой, n – натуральное число. Функция получает x и n как параметр;
15. написать функцию, вычисляющую $(-1)^n x^{2 \cdot n+1}$, где натуральное n и вещественное x передается в функцию как параметр;
16. написать функцию, вычисляющую $(2 \cdot n)^n$.

Часть 2

Используя три различных цикла, определить значение произведения (Таблица 4) для ***не***отрицательных чисел `x` (`double`) и `n` (`int`).

Таблица 4 – Индивидуальные названия по вычислению произведений

N	Формула
1.	$P(x,n) = \prod_{i=1}^{n}(i^2 + x);$
2.	$P(x,n) = \prod_{i=1}^{n}(i^3 + i \cdot x);$
3.	$P(x,n) = \prod_{i=1}^{n}\left(\frac{i}{2} + x\right);$
4.	$P(x,n) = \prod_{i=1}^{n}\left(\frac{1}{\sqrt{2 \cdot i}} + x\right);$
5.	$P(n) = \prod_{i=2}^{n} ln(i);$
6.	$P(x,n) = \prod_{i=1}^{n}(i^2 + x);$
7.	$P(n) = \prod_{i=1}^{n}(i + 1)^2;$
8.	$P(x,n) = \prod_{i=1}^{n}(i + x);$

N	Формула
9.	$P(x,n) = \prod_{i=1}^{n}\left(x + \sqrt{\frac{1}{i}}\right);$
10.	$P(x,n) = \prod_{i=1}^{n}\frac{2}{(i+x)^3};$
11.	$P(x,n) = \prod_{i=1}^{n} ln\lvert i + 2 \cdot x\rvert.$
12.	$P(n) = \prod_{i=1}^{n}\frac{3^{i+1}}{i^3}$
13.	$P(x,n) = \prod_{i=1}^{n}(x \cdot i - 1)^3;$
14.	$P(x,n) = \prod_{i=1}^{n}\left(2 \cdot x + \sqrt{i}\right);$
15.	$P(x,n) = \prod_{i=1}^{n}\left(i + \sqrt{2 \cdot \lvert x\rvert}\right);$
16.	$P(x,n) = \prod_{i=1}^{n}\frac{i+x}{(i+1)^2};$

Итерационные вычисления

Задание по теме состоит из ***двух*** частей:

- первая часть посвящена организации вычислений значений отрезка степенного ряда с заданной точностью;
- вторая часть посвящена вычисления с заданной точностью предела последовательности.

Часть 1. Вычисление с заданной точностью приближенного значения функции, представленной отрезком ряда

Пример задания и его выполнения

Формулировка примера задания. Даны действительные числа x, ε (где $\varepsilon > 0$ – точность). Вычислить с заданной точностью ε приближенное значение бесконечной суммы и сравнить его со значением, вычисляемым встроенной функцией.

Рассмотрим пример итерационного вычисления с помощью разложения в бесконечный ряд. Функции e^x на интервале $x \in (-1,1)$ с точностью ε:

$$1+\frac{x}{1!}+\frac{x^2}{2!}+\ldots+\frac{x^n}{n!}+\ldots$$

Критерием остановки суммирования является условие малости очередного прибавляемого члена ряда по отношению к введенному значению точности:

$$\left|\frac{x^n}{n!}\right| \leq \varepsilon$$

Программа.

main.cpp

```
1  #include <iostream>
2  #include <cmath>
3  using namespace std;
4
5  //прототип функции
```

```
6  double expUser( double, double);
7
8  int main(void) {
9      double x, epsilon;
10
11     while(true) {
12         cout<< "Введите x и точность" << endl;
13         cin>>x>>epsilon;
14         if ( fabs(x) < 1 && fabs(epsilon) < 0.05) break;
15         cout<< "Параметры введены некорректно. "
16             <<" Попробуйте еще раз!"<< endl;
17     }
18
19     cout <<"Значение итерационно вычисленной функции = "
20          << expUser(x, epsilon)
21          << endl
22          << "Значение встроенной функции = " << exp(x);
23     return 0;
24 }
25
26 double expUser(double x,double epsilon) {
27     double Sum = 1, Term = 1;
28     for (int i = 1; fabs(Term) > epsilon; i++) {
29         Term = Term * x / i;
30         Sum = Sum + Term;
31     }
32     return Sum;
33 }
```

Результаты работы программы:

```
Введите x и точность
0.7 0.00005
Значение итерационно вычисленной функции = 2.01375
Значение встроенной функции = 2.01375
```

Варианты индивидуальных заданий

Используя любой из операторов цикла написать программу вычисляющую приближенную сумму с заданной точностью и сравнить полученное значение со значением соответствующей функции из заголовочного файла `cmath`:

1. бесконечный ряд: $1-\frac{x^2}{2!}+\frac{x^4}{4!}-\ldots+(-1)^m\frac{x^{2m}}{(2m)!}+\ldots$, точное значение для сравнения: $\cos(x)$ ($x\in R$);

2. бесконечный ряд: $1+\frac{1}{2}x-\frac{1\cdot1}{2\cdot2}x^2+\frac{1\cdot1\cdot3}{2\cdot4\cdot6}x^3-\frac{1\cdot1\cdot3\cdot5}{2\cdot4\cdot6\cdot8}x^4+\ldots$, точное значение для сравнения: $(1+x)^{1/2}$ ($x\in\left]-1;1\right[$);

3. бесконечный ряд: $x+\frac{x^3}{3!}+\frac{x^5}{5!}+\ldots+\frac{x^{2m-1}}{(2m-1)!}+\ldots$, точное значение для сравнения: $\mathrm{sh}(x)$ ($x\in R$);

4. бесконечный ряд: $1-\frac{3}{2}x+\frac{3\cdot5}{2\cdot4}x^2-\frac{3\cdot5\cdot7}{2\cdot4\cdot6}x^3+\frac{3\cdot5\cdot7\cdot9}{2\cdot4\cdot6\cdot8}x^4-\ldots$, точное значение для сравнения: $(1+x)^{-3/2}$ ($x\in\left]-1;1\right[$);

5. бесконечный ряд: $1+\frac{x^2}{2!}+\frac{x^4}{4!}+\ldots+\frac{x^{2m}}{(2m)!}+\ldots$, точное значение для сравнения: $\mathrm{ch}(x)$ ($x\in R$);

6. бесконечный ряд: $1-\frac{5}{2}x+\frac{5\cdot7}{2\cdot4}x^2-\frac{5\cdot7\cdot9}{2\cdot4\cdot6}x^3+\frac{5\cdot7\cdot9\cdot11}{2\cdot4\cdot6\cdot8}x^4-\ldots$, точное значение для сравнения: $(1+x)^{-5/2}$ ($x\in\left]-1;1\right[$);

7. бесконечный ряд: $x-\frac{x^2}{2}+\frac{x^3}{3}+\ldots+(-1)^{n-1}\frac{x^n}{n}+\ldots$, точное значение для сравнения: $\ln(1+x)$ ($x\in\left]-1;1\right]$);

8. бесконечный ряд: $1+\frac{1}{4}x-\frac{1\cdot3}{4\cdot8}x^2+\frac{1\cdot3\cdot7}{4\cdot8\cdot12}x^3-\frac{1\cdot3\cdot7\cdot11}{4\cdot8\cdot12\cdot16}x^4+\ldots$, точное значение для сравнения: $(1+x)^{1/4}$ ($x\in\left]-1;1\right[$);

9. бесконечный ряд: $x-\frac{x^3}{3}+\frac{x^5}{5}+\ldots+(-1)^{m-1}\frac{x^{2m-1}}{2m-1}+\ldots$, точное значение для сравнения: $\mathrm{arctg}(x)$ ($x\in\left[-1;1\right]$);

10. бесконечный ряд: $-x-\frac{x^2}{2}-\frac{x^3}{3}-\ldots-\frac{x^n}{n}-\ldots$, точное значение для сравнения: $\ln(1-x)$ $(x\in[-1;1[)$;

11. бесконечный ряд: $1-x+x^2-\ldots+(-1)^n x^n+\ldots$, точное значение для сравнения: $\frac{1}{1+x}$ $(x\in]-1;1[)$;

12. бесконечный ряд: $1+x+x^2-\ldots+x^n+\ldots$, точное значение для сравнения: $\frac{1}{1-x}$ $(x\in]-1;1[)$;

13. бесконечный ряд: $1-2\cdot x+3\cdot x^2-4\cdot x^3+5\cdot x^4-\ldots$, точное значение для сравнения: $(1+x)^{-2}$ $(x\in]-1;1[)$;

14. бесконечный ряд: $2\left(x+\frac{x^3}{3}+\frac{x^5}{5}+\ldots+\frac{x^{2n-1}}{2n-1}+\ldots\right)$, точное значение для сравнения: $\ln\frac{1+x}{1-x}$ $(x\in]-1;1[)$;

15. бесконечный ряд: $1-\frac{1}{2}x+\frac{3}{8}x^2-\frac{5}{16}x^3+\ldots+(-1)^{n-1}\frac{(2n-3)!!}{(2n)!!}x^n+\ldots$, точное значение для сравнения: $\frac{1}{\sqrt{1+x}}$ $(x\in[-1;1])$.

Часть 2. Вычисление с заданной точностью приближенного значения предела последовательности

Пример задания и его выполнения

Формулировка примера задания. Вычислить предел сходящейся последовательности $\{a_n\}_{n=1}^{\infty}$. Вычисления прекратить, если выполнено условие $|a_{n+1}-a_n|<\varepsilon$. После окончания вычислений в качестве значения предела вывести на экран значение a_n.

Рассмотрим пример вычисления пердела последовательности $a_n = \frac{n}{\sqrt{n^2+1}+\sqrt{n^2-1}}$ (т.е. вычислить приближенное значение $\lim_{n\to\infty} \frac{n}{\sqrt{n^2+1}+\sqrt{n^2-1}}$).

Программа.

main.cpp

```cpp
#include <iostream>
#include <cmath>
using namespace std;

//прототипы функций
double accuracy(double);
double sequence(unsigned);

int main(void) {
    double epsilon;
    while(true) {
        cout << "Введите точность" << endl;
        cin >> epsilon;
        if ( fabs(epsilon) < 0.05) break;
        cout<< "Точность введена некорректно. "
            <<" Попробуйте еще раз!"<< endl;
    }
    cout << "Значение итерационно вычисленного"
         << " предела = "
         << accuracy(epsilon)
         << endl;
    return 0;
}

double accuracy(double epsilon) {
    double current, next;
    int i = 0;
    do {
        current = sequence(i);
        next = sequence(i + 1);
        i++;
    }
```

```
33      while (fabs(current- next)> epsilon);
34      return current;
35  }
36
37  double sequence(unsigned n) {
38      return n / (sqrt(n * n + 1) + sqrt(n * n - 1));
39  }
```

Результат работы программы:

```
Введите точность
0.0005
Значение итерационно вычисленного предела = 0.500245
```

Варианты индивидуальных заданий

Вычислить предел последовательности $\lim\limits_{n\to\infty} a_n$:

1. $a_n = \frac{3\cdot n^2-7\cdot n+1}{2-5\cdot n-6\cdot n^2}\ (n = 0,2,\ldots)$;
2. $a_n = \frac{2}{n}\ (n = 1,2,\ldots)$;
3. $a_n = \frac{n-1}{n^2}\ (n = 1,2,\ldots)$;
4. $a_n = \frac{(-1)^n}{n+1}\ (n = 0,2,\ldots)$;
5. $a_n = \frac{1}{3^n}\ (n = 0,2,\ldots)$;
6. $a_n = \frac{1}{\sqrt{n+5}}\ (n = 0,2,\ldots)$;
7. $a_n = \frac{1}{\sqrt[n]{2}}\ (n = 1,2,\ldots)$;
8. $a_n = \frac{1}{\sqrt[3]{n^2+4n-2}}\ (n = 1,2,\ldots)$;
9. $a_n = \sin\left(\frac{1}{n+3}\right)\ (n = 0,2,\ldots)$.

Рекурсия

Пример задания и его выполнения

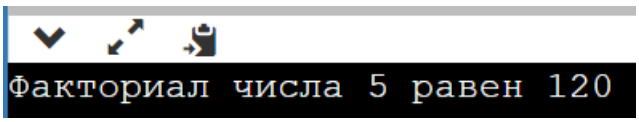
Формулировка примера задания. Написать функцию на языке C++, рекурсивно вычисляющую факториал натурального числа $n = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot n$ $(n!)$ с использованием тернарной операции.

Пример выполнения задания.

main.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  long factorial(int n) {
5      return n == 1 ? 1 : n * factorial(n - 1);
6  }
7
8  int main() {
9      cout<<"Факториал числа 5 равен "<< factorial(5);
10     return 0;
11 }
```

Результат работы программы:



Варианты индивидуальных заданий

Часть 1

Индивидуальные задания находятся в таблице темы «Вычисление сумм» (Таблица 3).

Часть 2

Написать рекурсивную функцию, вычисляющую:

1. $(-1)^n \cdot n!$, где натуральное n передается в функцию как параметр;
2. $2^{2 \cdot n+1}$, где натуральное n передается в функцию как параметр;
3. значение двойного факториала натурального n (если n нечетное, то $n!! = 1 \cdot 3 \cdot 5 \cdot \ldots \cdot n$, если n четное, то $n!! = 1 \cdot 2 \cdot 4 \cdot \ldots \cdot n$), где n передается в функцию как параметр;
4. $\big(sin(x)\big)^{2 \cdot n}$, где натуральное n и вещественное x передается в функцию как параметр;
5. $(-1)^n x^n$, где натуральное n и вещественное x передается в функцию как параметр;
6. $3^{2 \cdot n}$, где натуральное n передается в функцию как параметр;
7. произведение $\left(1 + \frac{1}{1^2}\right) \cdot \left(1 + \frac{1}{2^2}\right) \cdot \ldots \cdot \left(1 + \frac{1}{n^2}\right)$, где натуральное n передается в функцию как параметр;
8. $x^{3 \cdot n}$, где натуральное n и вещественное x передается в функцию как параметр;
9. $(-1)^n x^{2 \cdot n}$, где натуральное n и вещественное x передается в функцию как параметр;
10. $7^{3 \cdot n}$, где натуральное n передается в функцию как параметр;
11. значение факториал натурального $(2 \cdot n + 1)$ ($(2 \cdot n + 1)! = 1 \cdot 2 \cdot 3 \cdot \ldots \cdot (2 \cdot n + 1)$), где n передается в функцию как параметр;
12. $(x + 5)^{2n+1}$, где натуральное n и вещественное x передается в функцию как параметр;
13. $(-1)^n x^{2 \cdot n+1}$, где натуральное n и вещественное x передается в функцию как параметр;
14. $(2 \cdot n)^n$, где натуральное n передается в функцию как параметр;
15. x^n, где натуральное n и вещественное x передается в функцию как параметр.

Обработка одномерных массивов. Алгоритмы, требующие возвращения функцией значения

Часть 1. Передача в функцию массива автоматической памяти по ссылке

Пример задания и его выполнения

Формулировка примера задания. В одномерном ***автоматическом*** массиве из `n (n < N = 100)` чисел с плавающей точкой с помощью функции найти минимальный, используя ***ссылки*** в качестве инструмента передачи массива в функцию. Инициализацию массива выполнить случайными целыми числами.

Программа.

main.cpp

```
#include <iostream>
#include <cmath>
using namespace std;

const int N = 100;

//прототипы функций без имен формальных
//параметров
void initArray(float (&)[N], int);
void printArray(float (&)[N], int);
float minArray(float (&)[N], int);

int main() {
    int n;
    float array[N];

    cout << "Input dimension" << endl;
    //ввод активной размерн.
    cin >> n;
```

```
20
21      initArray(array, n);
22
23      cout << "Initialized array:" << endl;
24      printArray(array, n);
25
26      cout << "Minimum of array "
27           << minArray(array, n);
28      return 0;
29  }
30
31  void initArray(float (&a)[N], int n) {
32      srand(time(0));
33      for(int i = 0; i < n; i++) {
34          a[i] = rand() % 10;
35      }
36  }
37
38  void printArray(float (&a)[N], int n) {
39      for(int i = 0; i < n; i++) {
40          cout << a[i] << " ";
41      }
42      cout << endl;
43  }
44
45  float minArray(float (&a)[N], int n) {
46      float min = a[0];
47      for(int i = 1; i < n; i++) {
48          if (min > a[i]) min = a[i];
49      }
50      return min;
51  }
```

Результаты работы программы:

```
Input dimension
7
Initialized array:
6 0 0 4 7 4 1
Minimum of array 0
```

Варианты индивидуальных заданий

В соответствии с заданием инициализировать массив (или ***несколько*** массивов) из `n (n < N = 100)` чисел случайными значениями. Используя любой из операторов цикла с помощью функций, решить поставленную задачу:

1. найти разность между наибольшим и наименьшим значением в массиве;
2. найти среднее арифметическое наибольшего и наименьшего значения в массиве;
3. понимая, что элементы массивов $x[i]$ и $y[i]$ обозначают координаты точек на плоскости, определить минимальный радиус круга, в который попадают все эти точки;
4. определить сколько раз в массиве встретится максимум;
5. ввести число z, выяснить сколько элементов из массива превышает число z;
6. найти максимальную сумму двух соседних элементов массива;
7. найти количество элементов массива, которые меньше своего левого соседа;
8. проверить образуют ли массив возрастающую или убывающую последовательность;
9. понимая, что элементы массивов $a[i]$, $b[i]$ и $c[i]$ обозначают длины ребер параллелепипедов, определить параллелепипед с максимальным объемом;
10. понимая, что элементы массивов $x[i]$ и $y[i]$ обозначают координаты точек на плоскости, определить среднюю длину радиус-векторов точек;
11. понимая, что элементы массивов $a[i]$, $b[i]$ обозначают длины сторон прямоугольников, определить прямоугольник с минимальной площадью;
12. найти среднее арифметическое и вернуть в качестве возвращаемого значения количество элементов, превышающих среднее арифметическое;
13. понимая, что элементы массивов $a[i]$, $b[i]$ обозначают длины катетов прямоугольных треугольников, определить треугольник с самой длинной гипотенузой;
14. с помощью функции определить скалярное произведение двух векторов a, b.
15. понимая, что элементы массивов $x[i]$, $y[i]$ обозначают координаты точек на плоскости, определить максимальную длину радиус-векторов точек.

Часть 2. Передача массива, выделенного в куче, в функцию по указателю

Пример задания и его выполнения

Формулировка примера задания. В одномерном массиве из `n` чисел с плавающей точкой, расположенном в куче, с помощью функции найти минимальный, используя ***указатель*** в качестве инструмента передачи массива в функцию. Инициализацию массива выполнить с помощью ввода чисел с клавиатуры.

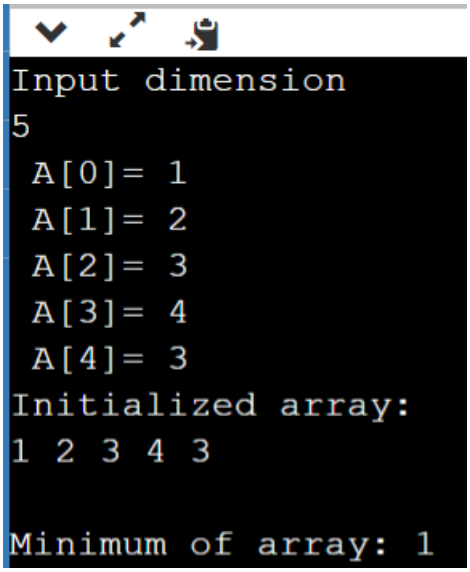
Программа.

main.cpp

```cpp
#include <iostream>
using namespace std;

void newInitArray(float *, int);
void printArray(float *, int);
float minArray(float *, int);

int main() {
    int size;
    cout << "Input dimension" << endl;
    //ввод размерности
    cin >> size;
    //выдел. памяти в куче
    float * array = new float [size];
    newInitArray(array, size);
    cout << "Initialized array:" << endl;
    printArray (array, size);
    cout << endl <<"Minimum of array: "
         << minArray(array, size)
         << endl;
    delete []array;
    return 0;
}
```

```
24
25 void newInitArray(float *a, int n) {
26     for(int i = 0; i < n; i++) {
27         cout << " a["<< i <<"]= ";
28         cin >> a[i];
29     }
30 }
31 void printArray(float *a, int n) {
32     for(int i = 0; i < n; i++) {
33         cout << a[i] << " ";
34     }
35     cout<<endl;
36 }
37 float minArray(float *a, int n) {
38     float min = a[0];
39     for(int i = 1; i < n; i++) {
40         if (min > a[i]) min = a[i];
41     }
42     return min;
43 }
```

Результаты работы программы:



Варианты индивидуальных заданий

Ввести одномерный целочисленный массив a из n элементов. Используя любой из операторов цикла с помощью функций решить поставленную задачу:

1. определить с помощью функции в нем количество элементов кратных четырем;
2. определить в нем произведение элементов, значения которых лежат вне диапазона $[a;\ b]$;
3. определить в нем сумму остатков от деления на 4 тех элементов, которые не кратны трем;
4. определить в нем среднее геометрическое элементов, стоящих на четных позициях;
5. определить среднее арифметическое элементов, стоящих на позициях кратных трем;
6. определить произведение элементов чье значение без остатка делится на 3 и не делится на 2;
7. определить в нем сумму элементов, стоящих на позициях, чей номер больше записанного в них значения;
8. определить в нем произведение нечетных значений элементов;
9. определить в нем количество элементов, квадрат которых больше введенного z;
10. определить произведение элементов, которые при делении на 2 дают такой же остаток, как и при делении на 3;
11. определить в нем среднее геометрическое четных элементов;
12. определить в нем количество тех элементов, которые при делении на 3 дают остаток 2;
13. определить в нем среднее геометрическое тех элементов, которые при делении на 4 дают остаток 1 или 3;
14. определить количество тех элементов, которые без остатка делятся на собственный индекс;
15. определить количество тех элементов, стоящих на четных позициях, которые сами четны;
16. определить среднее арифметическое квадратов элементов, стоящих на позициях, которые при делении на 4 дают остаток 2.

Перемещение по одномерным массивам с помощью указателей

Требования к выполнению заданий из ***обеих*** частей по данной теме: при выполнении задания ***нельзя*** обращаться к элементу массива ***по индексу***, т.е. с помощью ***операции индексирования***.

Часть 1. Передача в функцию массива автоматической памяти по указателю

Пример задания и его выполнения

Формулировка примера задания. В одномерном ***автоматическом*** массиве из `n (n < N = 100)` чисел с плавающей точкой с помощью функции найти минимальный, используя ***указатели*** в качестве инструмента как передачи массива в функцию, так и перемещения по нему.

Инициализацию массива выполнить случайными числами.

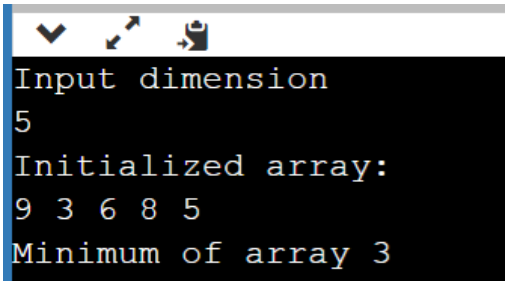
Программа.

main.cpp

```
#include <iostream>
#include <cmath>
using namespace std;

const int N = 100;

//прототипы функций без имен
//формальных параметров
void initArray(float*, float*);
void printArray(float *, float*);
float minArray(float*, float*);

int main() {
    int n;
    float array[N];

    cout << "Input dimension"<<endl;
```

```
18      //ввод активной размерности
19      cin >> n;
20
21      initArray(array, array + n);
22
23      cout << "Initialized array:"
24           << endl;
25      printArray(array, array + n);
26
27      cout << "Minimum of array "
28           << minArray(array, array + n);
29      return 0;
30  }
31
32  void initArray(float *begin, float *end) {
33      float *ptr;
34      srand(time(0));
35      for(ptr = begin; ptr < end; ptr++) {
36          *ptr = rand() % 10;
37      }
38  }
39
40  void printArray(float *begin, float *end) {
41      float *ptr;
42      for(ptr = begin; ptr < end; ptr++) {
43          cout << *ptr << " ";
44      }
45      cout<<endl;
46  }
47
48  float minArray(float *begin, float *end) {
49      float *ptr;
50      float min = *begin;
51      for(ptr = begin + 1; ptr < end; ptr++) {
52          if (min > *ptr) min = *ptr;
53      }
54      return min;
55  }
```

Результат работы программы:



Варианты индивидуальных заданий

Инициализировать массив/вектор (или массив***ы***/вектор***ы***) из `n (n < N = 100)` чисел случайными значениями. Используя любой из операторов цикла с помощью функций решить поставленную задачу:

Замечание.

В рамках выполнения заданий по данной теме нельзя использовать в качестве имен пользовательских функций `max()` и/или `min()`, которые являются синонимами функций, описанных в пространстве `std`.

1. понимая, что элементы массивов $a[i]$, $b[i]$ и $c[i]$ обозначают длины ребер параллелепипедов, определить параллелепипед с максимальным объемом;
2. понимая, что элементы массивов $x[i]$ и $y[i]$ обозначают координаты точек на плоскости, определить среднюю длину радиус-векторов точек;
3. понимая, что элементы массивов $a[i]$, $b[i]$ обозначают длины сторон прямоугольников, определить прямоугольник с минимальной площадью;
4. найти среднее арифметической и вернуть в качестве возвращаемого значения количество элементов, превышающих среднее арифметическое;
5. понимая, что элементы массивов $a[i]$, $b[i]$ обозначают длины катетов прямоугольных треугольников, определить треугольник с самой длинной гипотенузой;
6. с помощью функции определить скалярное произведение двух векторов a, b.
7. понимая, что элементы массивов $x[i]$, $y[i]$ обозначают координаты точек на плоскости, определить максимальную длину радиус-векторов точек;
8. найти разность между наибольшим и наименьшим значением в массиве;

9. найти среднее арифметическое наибольшего и наименьшего значения в массиве;
10. понимая, что элементы массивов $x[i]$ и $y[i]$ обозначают координаты точек на плоскости, определить минимальный радиус круга, в который попадают все эти точки;
11. определить сколько раз в массиве встретится максимум;
12. ввести число z, выяснить сколько элементов из массива превышает число z;
13. найти максимальную сумму двух соседних элементов массива;
14. найти количество элементов массива, которые меньше своего левого соседа;
15. проверить образуют ли массив возрастающую или убывающую последовательность.

Часть 2. Передача в функцию массива из кучи по указателю

Пример задания и его выполнения

Формулировка примера задания. В одномерном массиве из `n` чисел с плавающей точкой, выделенном в куче, с помощью функции найти максимальный, используя ***указатели*** в качестве инструмента обработки массива.

Инициализацию массива выполнить с помощью ввода чисел с клавиатуры.

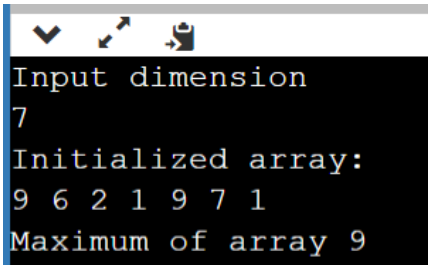
Программа.

main.cpp

```
1  #include <iostream>
2  #include <cmath>
3  using namespace std;
4
5  //прототипы функций без имен
6  //формальных параметров
7  void initArray(float*, float*);
8  void printArray(float*, float*);
9  float maxArray(float*, float*);
10
```

```
11  int main() {
12      int size;
13      cout << "Input dimension" << endl;
14      //ввод размерности
15      cin >> size;
16      float* array = new float[size];
17      initArray(array, array + size);
18
19      cout <<"Initialized array:" << endl;
20      printArray(array, array + size);
21
22      cout << endl << "Maximum of array "
23           << maxArray(array, array + size);
24      return 0;
25  }
26
27  void initArray(float* begin, float* end) {
28      srand(time(0));
29      float* ptr = begin;
30      for( ; ptr < end; ptr++) {
31          *ptr = rand() % 10;
32      }
33  }
34
35  void printArray(float *begin, float *end) {
36      float *ptr = begin;
37      for( ; ptr < end; ptr++) {
38          cout << (*ptr) << " ";
39      }
40  }
41
42  float maxArray(float *begin, float *end) {
43      float max = *begin;
44      float* ptr = begin + 1;
45      for( ; ptr < end; ptr++) {
46          if (max < *ptr) max = *ptr;
47      }
48      return max;
49  }
```

Результаты работы программы



Варианты индивидуальных заданий

Ввести одномерный целочисленный массив a из n элементов. Используя любой из операторов цикла с помощью функций решить поставленную задачу:

1. определить произведение элементов, которые при делении на 2 дают такой же остаток, как и при делении на 3;
2. определить в нем среднее геометрическое четных элементов;
3. определить в нем количество тех элементов, которые при делении на 3 дают остаток 2;
4. определить в нем среднее геометрическое тех элементов, которые при делении на 4 дают остаток 1 или 3;
5. определить количество тех элементов, которые без остатка делятся на собственный индекс;
6. определить количество тех элементов, стоящих на четных позициях, которые сами четны;
7. определить среднее арифметическое квадратов элементов, стоящих на позициях, которые при делении на 4 дают остаток 2;
8. определить с помощью функции в нем количество элементов кратных четырем;
9. определить в нем произведение элементов, значения которых лежат вне диапазона $[a;\ b]$;
10. определить в нем сумму остатков от деления на 4 тех элементов, которые не кратны трем;
11. определить в нем среднее геометрическое элементов, стоящих на четных позициях;
12. определить среднее арифметическое элементов, стоящих на позициях кратных трем;
13. определить произведение элементов чье значение без остатка делится на 3 и не делится на 2;
14. определить в нем сумму элементов, стоящих на позициях, чей номер больше записанного в них значения;

15. определить в нем произведение нечетных значений элементов;
16. определить в нем количество элементов, квадрат которых больше введенного z.

Перестановки и сортировки одномерных массивов

Пример задания и его выполнения

Формулировка примера задания для одномерного массива в стеке:

- написать ***две*** служебные функции:
  - заполняющую одномерный массив случайными числами с ***плавающей*** точкой в диапазоне `[-10; 10]`;
  - выводящую значения массива в строку на экран;
- написать ***три*** функции выполняющих:
  - реверс (зеркальное отображение массива в себя) и возвращать по ссылке результат преобразований;
  - циклический сдвиг влево;
  - сортировку по возрастанию.

Замечание.

Пример программы демонстрирует возможность использования различных механизмов обращения к массиву в рамках одной программы, т.е. и через ссылку, и через указатель.

Программа.

main.cpp

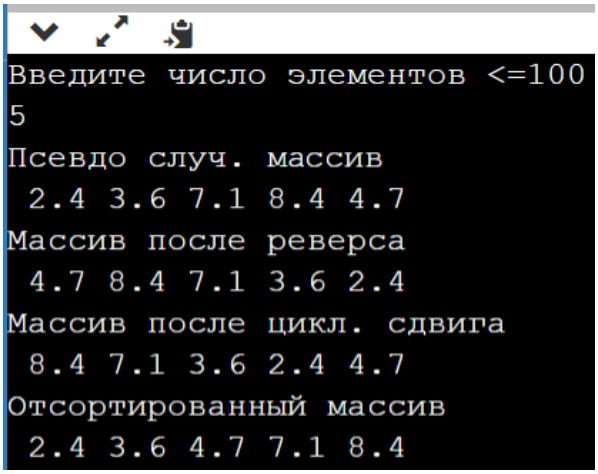
```
1  #include <iostream>
2  //для функций rand() и srand()
3  #include <cstdlib>
4  //для функции time
5  #include <ctime>
6  #define N 100
7  #define MAX_RAND 100
8  using namespace std;
9
10 void initArray(double (&)[N], int);
```

```
11  void printArray(double (&)[N], int);
12  void reverseArray(double (&)[N], int);
13
14  //в одной программе допускается обращаться к
15  //одному и томуже массиву используя и
16  //указатель, и ссылку
17  void cyclicLeftShift(double*, int);
18  void selectionSort(double*, int n);
19
20  int main() {
21      double array[N];
22      int n;
23
24      while(true) {
25          cout <<"Введите число элементов <="
26               << N << endl;
27          cin >> n;
28          if(0 < n && n <= N) break;
29      }
30
31      initArray(array, n);
32
33      cout << "Псевдо случ. массив" << endl;
34      printArray(array, n);
35
36      reverseArray(array, n);
37      cout << endl <<"Массив после реверса"
38           << endl;
39      printArray(array, n);
40
41      cyclicLeftShift(array, n);
42      cout << endl <<"Массив после цикл. сдвига"
43           << endl;
44      printArray(array, n);
45
46      selectionSort(array, n);
47      cout << endl <<"Отсортированный массив"
48           << endl;
49      printArray(array, n);
50      return 0;
51  }
```

```
52
53  void initArray(double (&a)[N], int n) {
54      srand(time(0));
55      for(int i = 0; i < n; i++) {
56          a[i] = (rand() % MAX_RAND) * 0.1;
57      }
58  }
59
60  void printArray(double (&a)[N], int n) {
61      for(int i = 0; i < n; i++) {
62          cout<<' '<< a[i];
63      }
64  }
65
66  void reverseArray(double (&a)[N], int n) {
67      double temp;
68      for(int i = 0; i < n/2; i++) {
69          temp = a[i];
70          a[i] = a[n - 1 - i];
71          a[n - 1 - i] = temp;
72      }
73  }
74
75  void cyclicLeftShift(double* a, int n) {
76      double temp = a[0];
77      for(int i = 0; i < n - 1; i++) {
78          a[i] = a[i + 1];
79      }
80      a[n - 1] = temp;
81  }
82
83  void selectionSort(double* a, int n) {
84      float temp;
85      for(int i = 0; i < n - 1; i++) {
86          int min = i;
87          for(int j = i + 1; j < n; j++) {
88              if (a [j] < a [min]) {
89                  min = j;
90              }
91          }
92          temp = a[i];
93              a[i] = a[min];
```

```
94                  a[min] = temp;
95              }
96          }
```

Результат работы программы:



Варианты индивидуальных заданий

Часть 1. Зеркальная перестановка

Инициализировать массив из `n  (n = 10)` чисел случайными значениями (***как положительными, так и отрицательными***). Используя любой из операторов цикла с помощью функций решить поставленную задачу о ***зеркальной перестановке*:**

1. второй половины массива;
2. первой половины массива;
3. элементов массива, начиная с последнего отрицательного, имеющего ***не***четный индекс (для гарантии работоспособности алгоритма, два последних элемента массива после заполнения случайными числами переопределить с помощью тернарной операции положительными значениями);
4. второй трети массива;
5. отрицательных элементов массива, имеющих четные индексы;
6. элементов массива, начиная с последнего отрицательного, имеющего четный индекс (для гарантии работоспособности алгоритма, два последних элемента массива после заполнения случайными числами переопределить с помощью тернарной операции положительными значениями);
7. первой трети массива;

8. отрицательных элементов массива, имеющих ***не***четные индексы;
9. третьей трети массива;
10. элементов массива, имеющие четные индексы;
11. отрицательных элементов массива;
12. элементов массива, имеющие ***не***четные индексы;
13. положительных элементов массива;
14. положительных элементов массива, имеющих четные индексы;
15. положительных элементов массива, имеющих ***не***четные индексы;

Часть 2. Циклический сдвиг влево

Дополнить программу, написанную в соответствии с индивидуальным заданием из первой части текущей темы функцией, решающей задачу о ***циклической перестановке*** на один символ сдвигом влево ***указанной части массива*****:**

1. второй трети массива;
2. отрицательных элементов массива, имеющих четные индексы;
3. элементов массива, начиная с последнего отрицательного, имеющего **не**четный индекс (для гарантии работоспособности алгоритма, два последних элемента массива после заполнения случайными числами переопределить с помощью тернарной операции положительными значениями);
4. первой трети массива;
5. отрицательных элементов массива, имеющих **не**четные индексы;
6. третьей трети массива;
7. элементов массива, имеющие четные индексы;
8. отрицательных элементов массива;
9. элементов массива, имеющие **не**четные индексы;
10. положительных элементов массива;
11. положительных элементов массива, имеющих четные индексы;
12. положительных элементов массива, имеющих **не**четные индексы;
13. второй половины массива;
14. первой половины массива;
15. элементов массива, начиная с последнего отрицательного, имеющего четный индекс (для гарантии работоспособности алгоритма, два последних элемента массива после заполнения случайными числами переопределить с помощью тернарной операции положительными значениями);

Часть 3. Сортировка

Дополнить программу, написанную в соответствии с индивидуальным заданием из первой и второй частей текущей темы, функцией, решающей задачу о ***сортировке выбором*** по возрастанию ***указанной части массива*:**

1. первой трети массива;
2. отрицательных элементов массива, имеющих **не**четные индексы;
3. отрицательных элементов массива;
4. третьей трети массива;
5. элементов массива, имеющие четные индексы;
6. элементов массива, имеющие **не**четные индексы;
7. положительных элементов массива;
8. положительных элементов массива, имеющих четные индексы;
9. положительных элементов массива, имеющих **не**четные индексы;
10. второй половины массива;
11. первой половины массива;
12. элементы, начиная с последнего отрицательного, имеющего четный индекс (для гарантии работоспособности алгоритма, два последних элемента массива после заполнения случайными числами переопределить с помощью тернарной операции положительными значениями);
13. второй трети массива;
14. отрицательных элементов массива, имеющих четные индексы;
15. элементы, начиная с последнего отрицательного, имеющего **не**четный индекс (для гарантии работоспособности алгоритма, два последних элемента массива после заполнения случайными числами переопределить с помощью тернарной операции положительными значениями).

МНОГОФАЙЛОВЫЕ ПРОЕКТЫ В ИМПЕРАТИВНОМ ПРОГРАММИРОВАНИИ

Общие требования к выполнению заданий из этого раздела. Инструменты онлайн IDE OnlineGDB beta для создания многофайловых проектов

Критерии оценки выполнения задания

Несмотря на то, что в примерах выполнения заданий будет приводится код проекта состоящего, как и ранее из одного файла, но для получения оценки 10 «десять» баллов за выполняемое задание студенты должны ***самостоятельно*** создать многофайловый проект в соответствии с материалами лекций и разобранным в них примером ***перевода*** строки, стожащей символы-цифры, в число. Пример проекта доступен по ссылке URL: https://www.onlinegdb.com/edit/UydJ5EbHG (дата доступа 02.02.2023).

Если в соответствии с индивидуальными заданиями этой части будет создана ***однофайловая*** рабочая программа:

- с разделением на функции, то это будет оценено как 8 «восемь»;
- без использования функции – как 6 «шесть».

Сведения о многофайловых проектах

В многофайловых проектах императивного программирования в языке C++ используется определенная иерархия файлов:

- все прототипы функций, макроопределения и глобальные константы выносятся в заголовочные файлы (расширение .h);
- все описания функций выносятся в файлы с расширением .cpp;
- следует (но не обязательно) использовать одинаковые имена заголовочных файлов и соответствующих им cpp-файлов;

- программа (функция `main()`) в единственном числе остается в файле `main.cpp`;
- перед `main()` пишется набор необходимых директив препроцессора (в частности, с помощью `#include` подключаются необходимые заголовочные файлы).

При компиляции многофайлового проекта на первом шаге исполняются директивы препроцессора во всех `cpp`-файлах и формируется объединенный объектный код программы.

Использование многофайловой структуры позволяет несопоставимо облегчить работу коллектива программистов, каждый из которых разрабатывает и тестирует свой набор функций (в рамках собственного `cpp`-модуля), а также пишет для своего `cpp`-файла (`cpp`-модуля) еще и заголовочный файл (`.h`).

В этом случае сборка проекта осуществляется простым копированием отдельных файлов (модулей) в общий проект, дописыванием директив `#include` в файл, содержащий функцию `main()` а также последовательных вызовов новых функций в `main()`.

Использование многофайловых проектов дает возможность:

- доработки каждого из файлов в отдельности;
- подключения дополнительных файлов (расширения функциональности).

При этом сохраняется персональная ответственность каждого из разработчиков за качество работы функций, собранных в персонально разрабатываемых модулях.

Добавление файла в безымянный проект онлайн IDE `OnlineGDB beta`

Файл в проект «по умолчанию» (или безымянный проект) онлайн среды `OnlineGDB beta` добавляется нажатием на кнопку `New File (Ctrl+M)` (Рисунок 2).

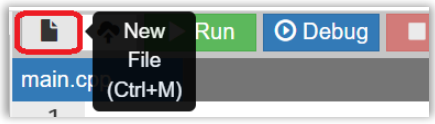


Рисунок 2 – Кнопка `New File (Ctrl+M)`

Далее появляется окно `New File` (Рисунок 3), в поле которого (выделено красным) следует написать имя файла и его расширение. Напомним, что в проекте могут использоваться файлы с расширением `.h` и `.cpp`.

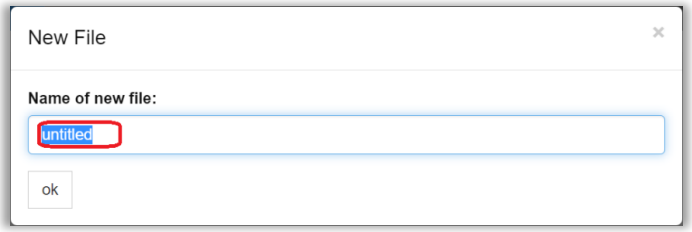


Рисунок 3 – Окно `New File`

По завершению ввода имени файла и расширения следует нажать на кнопку `ok` (или клавишу `Enter` на клавиатуре). В проекте появиться новая закладка, соответствующая имени созданного файла. Например, если в поле ввода окна `New File` (Рисунок 3) ввести название `example.cpp`, то результат операции будет иметь вид, указанный на рисунке ниже (Рисунок 4).

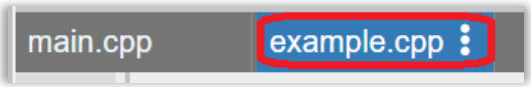


Рисунок 4 – Созданная закладка в безымянном проекте с введенным именем файла

Двумерные массивы. Алгоритмы, требующие возвращения функцией значения

Часть 1. Передача в функцию двумерного массива по ссылке

Пример задания и его выполнения

Формулировка примера задания. В двумерном автоматическом массиве размерностью `n x m (n, m < N = 100)` целых чисел с помощью функции найти максимальный. Инициализацию массива выполнить случайными целыми числами (в том числе отрицательными).

Программа.

main.cpp

```
#include <iostream>
#include <cmath>
using namespace std;

const int N = 100;

//прототипы функций без имен формальных
//параметров
void initMatrix(int (&)[N][N], int, int);
void printMatrix(int (&)[N][N], int, int);
int maxMatrix(int (&)[N][N], int, int);

int main() {
    int n, m;
    int matrix[N][N];

    cout << "Input dimensions n and m"
         << endl;
    //ввод активной размерн.
    cin >> n >> m;

    initMatrix(matrix, n, m);

    cout << "Initialized Matrix:" << endl;
    printMatrix(matrix, n, m);

    cout << "Maximum of Matrix "
         << maxMatrix(matrix, n, m);
    return 0;
}

void initMatrix(int (&a)[N][N], int n, int m) {
    srand(time(0));
    for(int i = 0; i < n; i++) {
        for(int j = 0; j < m; j++) {
            a[i][j] = 50 - rand() % 100;
        }
    }
}
```

```
40
41  void printMatrix(int (&a)[N][N], int n, int m) {
42      for(int i = 0; i < n; i++) {
43          for(int j = 0; j < m; j++) {
44              cout << a[i][j] << " ";
45          }
46      cout << endl;
47      }
48  }
49
50  int maxMatrix(int (&a)[N][N], int n, int m) {
51      int max = a[0][0];
52      for(int i = 0; i < n; i++) {
53          for(int j = 0; j < m; j++) {
54              if (max < a[i][j]) max = a[i][j];
55          }
56      }
57      return max;
58  }
```

Результат работы программы:

```
Input dimensions n and m
3
3
Initialized Matrix:
30 5 -34
-7 40 5
16 -19 -37
Maximum of Matrix 40
```

Варианты индивидуальных заданий

Используя любой из операторов цикла с помощью функций решить инициализировать массив из `n x m (n, m < N = 100)` чисел случайными ***целыми*** значениями. Далее определить в нем:

1. среднее арифметическое элементов, значения которых лежат в диапазоне `[2; 20]`;
2. среднее арифметическое элементов, стоящих на позициях с нечетной суммой индексов;

3. сумму элементов, чье значение без остатка делится на 2 и не делится на 3;
4. сумму элементов, чье значение без остатка делится на 3 и не делится на 2;
5. сумму элементов, стоящих на позициях, чьи индексы в сумме больше записанного в них значения;
6. произведение элементов, значения которых лежат вне диапазона [-10; 25];
7. количество элементов, квадрат которых больше 16;
8. произведение элементов, квадрат которых меньше 25;
9. сумму модулей положительных элементов;
10. среднее арифметическое модулей отрицательных элементов;
11. среднее геометрическое элементов, у которых оба индекса четные;
12. среднее геометрическое квадратов четных элементов;
13. количество тех элементов, которые при делении на 4 дают остаток 2;
14. произведение тех элементов, чей модуль лежит в диапазоне [15; 50];
15. количество тех элементов, которые без остатка делятся на собственный индекс (строки или столбца);
16. среднее арифметическое значение из минимального значения с обоими четными индексами и максимального значения с обоими нечетными индексами.

Часть 2. Передача в функцию двумерного массива по указателю на указатель

Пример задания и его выполнения

Формулировка примера задания. В двумерном массиве n x m целых чисел, расположенном в куче, с помощью функции найти максимальный, используя ***указатель на указатель*** в качестве инструмента передачи массива в функцию. Инициализацию массива выполнить с помощью ввода чисел с клавиатуры.

Программа.

main.cpp

```
1  #include <iostream>
2  using namespace std;
3
```

```
 4  void newInitMatrix(int**, int, int);
 5  void printMatrix(int**, int, int);
 6  int maxMatrix(int**, int, int);
 7  void deleteMatrix(int**, int);
 8
 9  int main() {
10      int n, m;
11
12      cout << "Input dimensions n and m"
13           << endl;
14      //ввод размерности
15      cin >> n >> m;
16
17      //выдел. дин. памяти
18      int ** matrix = new int* [n];
19      for(int i = 0; i < n; i++)
20          matrix[i] = new int [m];
21
22      newInitMatrix(matrix, n, m);
23
24      cout << "Initialized matrix:" << endl;
25      printMatrix(matrix, n, m);
26
27      cout << endl << "Maximum of Matrix: "
28           << maxMatrix(matrix, n, m)
29           << endl;
30      deleteMatrix(matrix, n);
31      return 0;
32  }
33
34  void newInitMatrix(int** a, int n, int m) {
35      for(int i = 0; i < n; i++) {
36          for(int j = 0; j < m; j++) {
37              cout << " a[" << i
38                   << "][" << j << "]= ";
39              cin >> a[i][j];
40          }
41      }
42  }
43
```

```
44 void printMatrix(int** a, int n, int m) {
45     for(int i = 0; i < n; i++) {
46         for(int j = 0; j < m; j++) {
47             cout << a[i][j] << " ";
48         }
49         cout << endl;
50     }
51 }
52
53 int maxMatrix(int** a, int n, int m) {
54     int max = a[0][0];
55     for(int i = 0; i < n; i++) {
56         for(int j = 0; j < m; j++) {
57             if (max < a[i][j]) max = a[i][j];
58         }
59     }
60     return max;
61 }
62
63 void deleteMatrix(int** a, int n) {
64     for(int i = 0; i < n; i++)
65         delete []a[i];
66     delete []a;
67 }
```

Результат работы программы:

```
Input dimensions n and m
2 2
 a[0][0]= 1
 a[0][1]= 2
 a[1][0]= 3
 a[1][1]= 4
Initialized matrix:
1 2
3 4

Maximum of Matrix: 4
```

Варианты индивидуальных заданий

Используя любой из операторов цикла с помощью функций ввести двумерный массив из `n x m` целых чисел с клавиатуры и решить поставленную задачу (массив должен быть создан в куче):

1. определить в нем количество тех элементов, стоящих на позициях с нечетной суммой индексов, которые сами четны;
2. определить в нем сумму остатков от деления на `2` тех элементов, которые не кратны `2`;
3. определить в нем произведение остатков от деления на `4` тех элементов, которые не кратны `4`;
4. определить в нем среднее геометрическое тех элементов, которые при делении на `4` дают остаток `1` или `3`;
5. определить в нем произведение тех элементов, чей модуль лежит вне диапазона `[10; 25]`;
6. найти суммы индексов строк и столбцов нечетных элементов массива.
7. определить произведение модулей отрицательных элементов;
8. определить сумму элементов, которые при делении на `2` дают такой же остаток, как и при делении на `3`;
9. определить сумму элементов, которые при умножении на `7` дают значение большее, чем при возведении в квадрат;
10. определить среднее геометрическое квадратов элементов, стоящих на позициях, у которых хотя бы один индекс кратен `2`;
11. определить среднее геометрическое квадратов элементов, стоящих на позициях, у которых как минимум один из индексов при делении на `3` дает остаток `2`;
12. определить в нем количество элементов кратных `4`;
13. определить в нем произведение нечетных элементов;
14. найти произведение индексов строк и сумму индексов столбцов для элементов массива больших `8`;
15. определить сумму квадратов индексов (как строк, так и столбцов) элементов, делящихся на `4` без остатка;
16. определить сумму индексов всех (их может быть несколько) минимальных по величине элементов двумерного массива.

Формирование двумерного массива специального вида

Часть 1. Передача в функцию двумерного массива по ссылке

Пример задания и его выполнения

Формулировка примера задания. В квадратном двумерном автоматическом массиве размерностью `n x n (n < N = 100)` разместить целые числа на главной или побочной диагоналях согласно представленному шаблону:

$$\begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & & 1 & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 1 & & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}$$

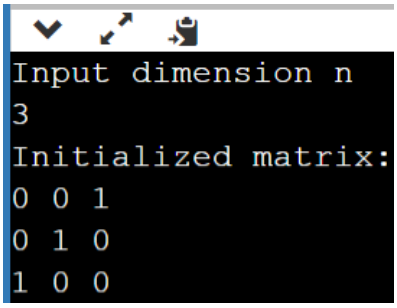
Программа.

main.cpp

```cpp
#include <iostream>
#include <cmath>
using namespace std;

const int N = 100;

//прототипы функций без имен формальных
//параметров
void initMatrix(int (&)[N][N], int);
void printMatrix(int (&)[N][N], int);

int main() {
    int n;
    int matrix[N][N];

    cout << "Input dimension n" << endl;
```

```
17      //ввод активной размерн.
18      cin >> n;
19
20      initMatrix(matrix, n);
21
22      cout << "Initialized matrix:"
23           << endl;
24      printMatrix(matrix, n);
25      return 0;
26  }
27
28  void initMatrix(int (&a)[N][N], int n) {
29      for(int i = 0; i < n; i++) {
30          a[i][n - 1 - i] = 1;
31      }
32  }
33
34  void printMatrix(int (&a)[N][N], int n) {
35      for(int i = 0; i < n; i++) {
36          for(int j = 0; j < n; j++) {
37              cout << a[i][j] << " ";
38          }
39          cout << endl;
40      }
41  }
```

Результат работы программы:



Варианты индивидуальных заданий

С помощью функции, разместить на одной из диагоналей двумерного массива заданных значений (Таблица 5).

Таблица 5 – Индивидуальные задания по размещению на диагоналях двумерного массива чисел

N	Шаблон
1	$\begin{pmatrix} 0 & 0 & \cdots & 0 & n \\ 0 & 0 & & n-1 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 2 & \cdots & 0 & 0 \\ 1 & 0 & & 0 & 0 \end{pmatrix}$
2	$\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 2 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & n-1 & 0 \\ 0 & 0 & & 0 & n \end{pmatrix}$
3	$\begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & & 2 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & n-1 & \cdots & 0 & 0 \\ n & 0 & & 0 & 0 \end{pmatrix}$
4	$\begin{pmatrix} n & 0 & \cdots & 0 & 0 \\ 0 & n-1 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & 2 & 0 \\ 0 & 0 & & 0 & 1 \end{pmatrix}$
5	$\begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & & 2 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 2 & \cdots & 0 & 0 \\ 1 & 0 & & 0 & 0 \end{pmatrix}$

N	Шаблон
6	$\begin{pmatrix} 0 & 0 & \cdots & 0 & 1^2 \\ 0 & 0 & & 2^2 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 2^2 & \cdots & 0 & 0 \\ 1^2 & 0 & & 0 & 0 \end{pmatrix}$
7	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \cdots \quad a_{n-1})$ Заполнить по шаблону: $\begin{pmatrix} 0 & 0 & \cdots & 0 & a_{n-1} \\ 0 & 0 & & a_{n-2} & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & a_1 & \cdots & 0 & 0 \\ a_0 & 0 & & 0 & 0 \end{pmatrix}$
8	$\begin{pmatrix} 0 & 0 & \cdots & 0 & n^2 \\ 0 & 0 & & (n-1)^2 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 2^2 & \cdots & 0 & 0 \\ 1^2 & 0 & & 0 & 0 \end{pmatrix}$
9	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \cdots \quad a_{n-1})$ Заполнить по шаблону: $\begin{pmatrix} 0 & 0 & \cdots & 0 & a_0 \\ 0 & 0 & & a_1 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & a_{n-2} & \cdots & 0 & 0 \\ a_{n-1} & 0 & & 0 & 0 \end{pmatrix}$
10	$\begin{pmatrix} n^2 & 0 & \cdots & 0 & 0 \\ 0 & (n-1)^2 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & 2^2 & 0 \\ 0 & 0 & & 0 & 1^2 \end{pmatrix}$

N	Шаблон
11	$\begin{pmatrix} 1^2 & 0 & \cdots & 0 & 0 \\ 0 & 2^2 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & (n-1)^2 & 0 \\ 0 & 0 & & 0 & n^2 \end{pmatrix}$
12	Заполнить единицами только **нечетные** строки на побочной диагонали. Шаблон для нечетного n имеет вид: $\begin{pmatrix} 0 & 0 & 0 & & 0 & 0 & \mathbf{0} \\ 0 & 0 & 0 & \cdots & 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 & & \mathbf{0} & 0 & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & \mathbf{0} & & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & \cdots & 0 & 0 & 0 \\ \mathbf{0} & 0 & 0 & & 0 & 0 & 0 \end{pmatrix}$
13	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \cdots \quad a_{n-1})$ Заполнить по шаблону: $\begin{pmatrix} a_0 & 0 & \cdots & 0 & 0 \\ 0 & a_1 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & a_{n-2} & 0 \\ 0 & 0 & & 0 & a_{n-1} \end{pmatrix}$

N	Шаблон
14	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \cdots \quad a_{n-1})$ Заполнить по шаблону: $\begin{pmatrix} a_{n-1} & 0 & \cdots & 0 & 0 \\ 0 & a_{n-2} & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & a_1 & 0 \\ 0 & 0 & & 0 & a_0 \end{pmatrix}$
15	Заполнить единицами только **четные** строки на главной диагонали. Шаблон для нечетного n имеет вид: $\begin{pmatrix} \mathbf{1} & 0 & 0 & & 0 & 0 & 0 \\ 0 & \mathbf{0} & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & & 0 & 0 & 0 \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & 0 & & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & \mathbf{0} & 0 \\ 0 & 0 & 0 & & 0 & 0 & \mathbf{1} \end{pmatrix}$
16	$\begin{pmatrix} 1^2 & 0 & \cdots & 0 & 0 \\ 0 & 2^2 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & 2^2 & 0 \\ 0 & 0 & & 0 & 1^2 \end{pmatrix}$

Часть 2. Передача в функцию двумерного массива по указателю на указатель

Пример задания и его выполнения

Формулировка примера задания. В двумерном квадратном массиве, расположенном в куче, размерностью `n x n` с помощью функции разместить целые числа согласно представленному шаблону:

$$\begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & & 1 & 1 \\ \vdots & & \ddots & \vdots & \\ 0 & 1 & & 1 & 1 \\ 1 & 1 & \cdots & 1 & 1 \end{pmatrix}$$

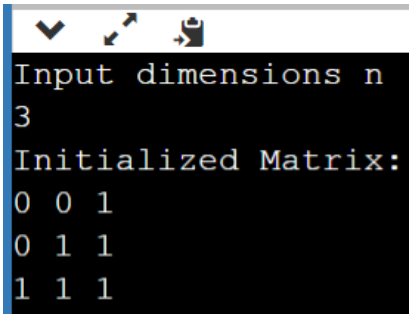
Программа.

main.cpp

```
1  #include <iostream>
2  using namespace std;
3
4  void newInitMatrix(int**, int);
5  void printMatrix(int**, int);
6  void deleteMatrix(int**, int);
7
8  int main() {
9      int n;
10
11     cout << "Input dimension n"
12          << endl;
13     //ввод размерности
14     cin >> n;
15
16     //выдел. дин. памяти
17     int** matrix = new int* [n];
18     for(int i = 0; i < n; i++)
19         matrix[i] = new int [n];
20
```

```
21      newInitMatrix(matrix, n);
22
23      cout << "Initialized matrix:"
24           << endl;
25      printMatrix(matrix, n);
26
27      deleteMatrix(matrix, n);
28      return 0;
29  }
30
31  void newInitMatrix(int** a, int n) {
32      for(int i = 0; i < n; i++) {
33          for(int j = 0; j < n; j++) {
34              if (j >= n - 1 - i) {
35                  a[i][j] = 1;
36              }
37              else {
38                  a[i][j] = 0;
39              }
40          }
41      }
42  }
43
44  void printMatrix(int** a, int n) {
45      for(int i = 0; i < n; i++) {
46          for(int j = 0; j < n; j++) {
47              cout << a[i][j] << " ";
48          }
49          cout << endl;
50      }
51  }
52
53  void deleteMatrix(int** a, int n) {
54      for(int i = 0; i < n; i++)
55          delete []a[i];
56      delete []a;
57  }
```

Результат работы программы:



Варианты индивидуальных заданий

С помощью функции, разместить в указанном треугольнике двумерного массива заданные значения (Таблица 6).

Таблица 6 – Треугольный шаблон заполнения двумерного массива

N	Шаблон
1	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \dots \quad a_{n-1})$ Заполнить по шаблону по строкам до главной диагонали: $\begin{pmatrix} a_0 & 0 & \dots & 0 & 0 \\ a_1 & a_0 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ a_{n-2} & a_{n-3} & \dots & a_0 & 0 \\ a_{n-1} & a_{n-2} & & a_1 & a_0 \end{pmatrix}$
2	По строкам: $\begin{pmatrix} 1 & 2 & \dots & n-1 & n \\ 0 & 1 & & n-2 & n-1 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \dots & 1 & 2 \\ 0 & 0 & & 0 & 1 \end{pmatrix}$
3	$\begin{pmatrix} 1 & 1 & \dots & 1 & 1 \\ 0 & 1 & & 1 & 1 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \dots & 1 & 1 \\ 0 & 0 & & 0 & 1 \end{pmatrix}$
4	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \dots \quad a_{n-1})$ Заполнить по шаблону (по строкам): $\begin{pmatrix} a_{n-1} & a_{n-2} & \dots & a_1 & a_0 \\ a_{n-2} & a_{n-3} & & a_0 & 0 \\ \vdots & & \ddots & \vdots & \\ a_1 & a_0 & \dots & 0 & 0 \\ a_0 & 0 & & 0 & 0 \end{pmatrix}$

N	Шаблон
5	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \dots \quad a_{n-1})$ Заполнить по шаблону (по строкам): $\begin{pmatrix} a_0 & a_1 & \cdots & a_{n-2} & a_{n-1} \\ 0 & a_0 & & a_{n-3} & a_{n-2} \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & & a_0 & a_1 \\ 0 & 0 & \cdots & 0 & a_0 \end{pmatrix}$
6	$\begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & 1 & & 1 & 0 \\ \vdots & & \ddots & \vdots & \\ 1 & 1 & & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}$
7	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \dots \quad a_{n-1})$ Заполнить по шаблону (по строкам): $\begin{pmatrix} 0 & 0 & \cdots & 0 & a_0 \\ 0 & 0 & & a_0 & a_1 \\ \vdots & & \ddots & \vdots & \\ 0 & a_0 & & a_{n-3} & a_{n-2} \\ a_0 & a_1 & \cdots & a_{n-2} & a_{n-1} \end{pmatrix}$

N	Шаблон
8	$\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 1 & 1 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ 1 & 1 & & 1 & 0 \\ 1 & 1 & \cdots & 1 & 1 \end{pmatrix}$
9	Заполнить единицами только **четные *строки и*** **четные *столбцы*** от главной диагонали. Шаблон для нечетного n имеет вид: $\begin{pmatrix} \mathbf{1} & \mathbf{0} & \mathbf{1} & & \mathbf{1} & \mathbf{0} & \mathbf{1} \\ 0 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 0 & \mathbf{1} & & \mathbf{1} & \mathbf{0} & \mathbf{1} \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & 0 & & \mathbf{1} & \mathbf{0} & \mathbf{1} \\ 0 & 0 & 0 & \cdots & 0 & \mathbf{0} & \mathbf{0} \\ 0 & 0 & 0 & & 0 & 0 & \mathbf{1} \end{pmatrix}$
10	По столбцам: $\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 2 & 1 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ n-2 & n-3 & & 1 & 0 \\ n-1 & n-2 & \cdots & 2 & 1 \end{pmatrix}$

N	Шаблон
11	Заполнить единицами только **нечетные** ***столбцы*** от побочной диагонали. Шаблон для нечетного n имеет вид: $\begin{pmatrix} 0 & 0 & 0 & & 0 & 0 & \mathbf{0} \\ 0 & 0 & 0 & \cdots & 0 & \mathbf{1} & \mathbf{0} \\ 0 & 0 & 0 & & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ & \vdots & & \ddots & & \vdots & \\ 0 & 0 & \mathbf{0} & & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ 0 & \mathbf{1} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & & \mathbf{0} & \mathbf{1} & \mathbf{0} \end{pmatrix}$
12	По диагонали: $\begin{pmatrix} 1 & 2 & \cdots & n-2 & n-1 \\ 0 & 1 & & n-3 & n-2 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & 1 & 2 \\ 0 & 0 & & 0 & 1 \end{pmatrix}$
13	По строкам: $\begin{pmatrix} 1 & 2 & \cdots & n-2 & n-1 \\ 1 & 2 & & n-2 & n-1 \\ \vdots & & \ddots & \vdots & \\ 1 & 2 & \cdots & n-2 & n-1 \\ 1 & 2 & & n-2 & n-1 \end{pmatrix}$

N	Шаблон
14	По столбцам: $\begin{pmatrix} 1 & 1 & \cdots & 1 & 1 \\ 0 & 2 & & 2 & 2 \\ \vdots & & \ddots & \vdots & \\ 0 & 0 & \cdots & n-2 & n-2 \\ 0 & 0 & & 0 & n-1 \end{pmatrix}$
15	По диагонали: $\begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 2 & 1 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ n-2 & n-3 & \cdots & 1 & 0 \\ n-1 & n-2 & & 2 & 1 \end{pmatrix}$
16	Предварительно задан одномерный массив: $(a_0 \quad a_1 \quad \cdots \quad a_{n-1})$ Заполнить по шаблону (по диагонали): $\begin{pmatrix} a_0 & 0 & \cdots & 0 & 0 \\ a_1 & a_0 & & 0 & 0 \\ \vdots & & \ddots & \vdots & \\ a_{n-2} & a_{n-3} & \cdots & a_0 & 0 \\ a_{n-1} & a_{n-2} & & a_1 & a_0 \end{pmatrix}$

Обработка строк

Замечание.

В части 2 текущей темы для получения оценки `10 «десять»` требуется ***расширить*** *функциональность многофайлового проекта, созданного в части 1. Это значит к созданному многофайловому проекту при выполнении части 1 необходимо добавить дополнительные заголовочный*

файл и файл-библиотеку в соответствии с заданием части 2, а также вызвать дополнительные функции из «расширения» в `main()`.

Если оба задания будут выполнены в рамках одного многофайлового проекта без добавления указанных файлов, то такой проект будет оценен как 9 «девять», а рабочая ***однофайловая*** *программа:*

- *с разделением на функции будет оценен как* `7 «семь»`*;*
- *без разделения на функции –* `5 «пять»`*.*

Часть 1. Функции, возвращающие значения

Пример задания и его выполнения

Формулировка примера задания. Ввести строку с клавиатуры, посчитать в ней количество букв, цифр, и пробелов. Результаты подсчетов вывести на экран.

Программа.

main.cpp

```
#include <iostream>
using namespace std;

int leterNumber(char*);
int digitNumber(char*);
int spaceNumber(char*);

int main() {
    const int STR_LEN = 250;
    char* str = new char[STR_LEN];

    cout << "Input string" << endl;
    cin.getline(str, STR_LEN);

    cout << "Number of letters: "
         << leterNumber(str) << endl
         << "Number of digits: "
         << digitNumber (str)
         << endl
         << "Number of spapces: "
```

```
21          << spaceNumber(str)
22          << endl;
23      delete []str;
24      return 0;
25  }
26
27  int leterNumber(char* s) {
28      int Number = 0;
29      for(int i = 0; s[i]!='\0'; i++) {
30          if( (s[i]>= 'A' && s[i]<= 'Z')
31              || (s[i]>= 'a' && s[i]<= 'z') ) {
32                  Number++;
33          }
34      }
35      return Number;
36  }
37
38  int digitNumber(char* str) {
39      int Number = 0;
40      for(int i = 0; str[i]!='\0'; i++) {
41          if(str[i]>= '0' && str[i]<= '9') {
42              Number++;
43          }
44      }
45      return Number;
46  }
47
48  int spaceNumber(char* string) {
49      int Number = 0;
50      for(int i = 0; string[i]!='\0'; i++) {
51          if(string[i]==' ') Number++;
52      }
53      return Number;
54  }
```

Результаты работы программы:

```
Input string
asd 123 34dgf
Number of letters: 6
Number of digits: 5
Number of spapces: 2
```

Варианты индивидуальных заданий

Дана строка ***латинских*** символов, цифр и разделителей (пробелы и знаки препинания). С помощью функций:

1. в строке определить символ с наименьшим кодом в стандартной кодовой таблице;
2. посчитать количество разделителей (пробелы и знаки препинания), если необходимо, то создать вспомогательный массив из разделителей;
3. определить номер по порядку первого слова, начинающегося с гласной буквы, если необходимо, то создать вспомогательный массив из гласных букв и разделителей;
4. посчитать количество заглавных согласных, если необходимо, то создать вспомогательный массив из заглавных согласных букв;
5. определить номер последнего слова, начинающегося с согласной, если необходимо, то создать вспомогательный массив из согласных букв и разделителей;
6. посчитать количество заглавных гласных в строке, если необходимо, то создать вспомогательный массив из заглавных гласных букв;
7. определить длину первого слова, начинающегося с согласной, если необходимо, то создать вспомогательный массив из согласных букв и разделителей;
8. посчитать количество слов в строке, если необходимо, то создать вспомогательный массив из разделителей;
9. в строке определить индекс символа с наибольшим кодом;
10. посчитать наибольшее количество подряд идущих гласных в строке, если необходимо, то создать вспомогательный массив из гласных;
11. определить наибольший суммарный код (из стандартной кодовой таблицы) слова в строке, если необходимо, то создать вспомогательный массив из разделителей;
12. в строке посчитать наибольшее количество групп знаков из подряд идущих разделителей, если необходимо, то создать вспомогательный массив из разделителей;
13. в строке посчитать количество строчных согласных, если необходимо, то создать вспомогательный массив из строчных согласных;
14. в строке посчитать сколько вхождений в строку имеет комбинация знаков «согласная буква-разделитель-согласная буква, если необходимо, то создать вспомогательный массив из согласных букв и разделителей;
15. посчитать количество строчных гласных, если необходимо, то создать вспомогательный массив из строчных гласных;

16. посчитать сколько вхождений в строку имеет комбинация букв «гласная буква-разделитель-гласная буква», если необходимо, то создать вспомогательный массив из гласных и разделителей;
17. определить среднюю длину слов в строке, если необходимо, то создать вспомогательный массив из разделителей;
18. определить количество слов длина, которых больше/меньше заданного числа, если необходимо, то создать вспомогательный массив из разделителей;
19. определить количество слов начинающихся и заканчивающихся одной и той же буквой, если необходимо, то создать вспомогательный массив из разделителей;
20. определить длину самого короткого/длинного слова в строке, если необходимо, то создать вспомогательный массив из разделителей;
21. определить номер по порядку самого длинного/короткого слова в строке, если необходимо, то создать вспомогательный массив из разделителей;
22. посчитать наибольшее количество подряд идущих согласных в строке, если необходимо, то создать вспомогательный массив из согласных.

Часть 2. Функции, не возвращающие значения

Пример задания и его выполнения

Формулировка примера задания. Ввести строку с клавиатуры зеркально переставить символы в строке (реверс) после этого сдвинуть циклически на один символ. Результаты вывести на экран.

Программа.

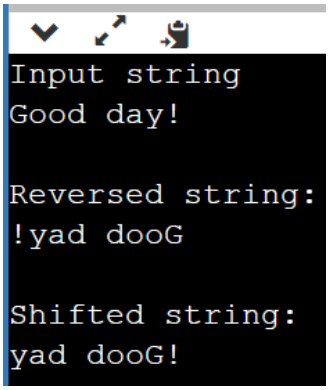
main.cpp

```cpp
#include <iostream>
using namespace std;

int strLength(char*);
void strRevers(char*);
void strCyclicLeftShift(char*);

int main() {
    const int STR_LEN = 25;
    char *str = new char[STR_LEN];
```

```
11
12      cout << "Input string" << endl;
13      cin.getline(str, STR_LEN);
14
15      strRevers(str);
16      cout << endl << "Reversed string:"
17          << endl << str << endl;
18
19      strCyclicLeftShift(str);
20      cout << endl<< "Shifted string:"
21          << endl << str << endl;
22
23      delete []str;
24      return 0;
25  }
26
27  int strLength(char* str) {
28      int n;
29      for(n = 0; str[n] != '\0'; n++);
30      return n;
31  }
32
33  void strRevers(char* str) {
34      int n = strLength(str);
35      char buff;
36      for(int i = 0; i < n / 2; i++) {
37          buff = str[i];
38          str[i] = str[n - 1 - i];
39          str[n - 1 - i] = buff;
40      }
41  }
42
43  void strCyclicLeftShift(char* str) {
44      int n = strLength(str);
45      char buff = str[0];
46      for(int i = 0; i < n - 1; i++) {
47          str[i] = str[i + 1];
48      }
```

```
49      str[n-1] = buff;
50  }
```

Результаты работы программы:



Варианты индивидуальных заданий

Дана строка символов. Слова в строке отделяются друг от друга разделителями (пробелами и знаками препинания).

Удаление символов в строке сдвигом влево

При выполнении задания сСимвол конца `'\0'` строки также сдвигать влево при удалении каждого заданного символа. Удалить из строки:

1. все пробелы, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
2. все разделители между словами, первое из которых заканчивается на гласную, а следующее начинается на согласную;
3. все запятые, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
4. все символы, имеющие наименьший код в стандартной кодовой таблице;
5. все точки, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
6. все символы, имеющие наибольший код в стандартной кодовой таблице;
7. все восклицательные знаки, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
8. все повторяющиеся гласные (первый в последовательности повторяющихся гласных оставить);

9. все повторяющиеся согласные (первый в последовательности повторяющихся согласных оставлять);
10. все двоеточия, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
11. все символы из строки между символами с наибольшим и наименьшим кодом в стандартной кодовой таблице; если в строке таких символов несколько, то удалить наибольшую из возможных подстрок;
12. все точки с запятой, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
13. все символы из строки между вторым и последним пробелами. Пробел между словами может остаться только один;
14. все символы между первой и последней запятыми (остается только одна запятая);
15. все знаки между второй и предпоследней гласной буквой (остается только начальная гласная, определяющая строку для удаления).
16. первое слово в строке;
17. все самые длинные слова;
18. все самые короткие слова.

Циклически переставить на один символ сдвигом вправо

Понимая, что подстрока – это участок строки ***циклически*** переставить:

1. вторую половину строки;
2. символы, имеющие четные индексы;
3. символы первой половины строки;
4. подстроку из трех подряд слов, последним из которых является предпоследнее слово строки;
5. k-ое слово в строке;
6. подстроку из двух подряд слов, начинающейся с третьего по порядку слова;
7. первое слово в строке;
8. четвертое с конца слово, начинающееся с гласной буквы;
9. последнее слово в строке;
10. второе с конца слово, начинающееся с согласной буквы;
11. второе слово, начинающееся с гласной буквы;
12. третье слово, начинающееся с согласной буквы;
13. подстроку (участок строки), заключенный между символами с максимальным и минимальным кодом из стандартной таблицы;
14. первую по порядку подстроку из подряд идущих цифр (предполагается, что группа из подряд идущих цифр содержит не менее двух цифр, а таких групп не менее одной);

15. символы подстроки, заключенные между третьей гласной и предпоследней согласной;
16. символы, имеющие нечетные индексы.

Зеркально переставить

Понимая, что подстрока – это участок строки ***зеркально*** переставить:

1. подстроку (участок строки), заключенный между символами с максимальным и минимальным кодом;
2. первую по порядку подстроку из подряд идущих цифр (предполагается, что группа из подряд идущих цифр содержит не менее двух цифр, а таких групп не менее одной);
3. символы подстроки, заключенные между третьей гласной и предпоследней согласной;
4. символы, имеющие нечетные индексы;
5. вторую половину строки;
6. символы, имеющие четные индексы;
7. символы первой половины строки;
8. подстроку из трех подряд слов, последним из которых является предпоследнее слово строки;
9. k-ое слово в строке (k вводится с клавиатуры);
10. подстроку из двух подряд слов, начинающейся с третьего по порядку слова;
11. первое слово в строке;
12. четвертое с конца слово, начинающееся с гласной буквы;
13. последнее слово в строке;
14. второе с конца слово, начинающееся с согласной буквы;
15. второе слово, начинающееся с гласной буквы;
16. третье слово, начинающееся с согласной буквы.

Приближенное вычисление определенных интегралов

Обобщенная формулировка задания. Найти приближенное значение определенного интеграла $\int_a^b f(x)dx$ с заданной точностью ε, используя 2 из предложенных квадратурных формул.

Для оценки достижения заданной точности использовать двойной пересчет: вычислить интеграл вначале для n разбиений отрезка, затем – для

$2n$; сравнить полученные результаты: если $|I_n - I_{2n}| < \varepsilon$, то $I \approx I_{2n}$, в противоположном случае вычислить интеграл для $4n$ и т.д.

Функцию $f(x)$ и предлагаемую квадратурную формулу оформить в виде внешней функций. Значения a, b, ε и начальное разбиение отрезка вести с клавиатуры.

Формулы для вычисления интегралов:

1. ***Составная формула трапеций:***

$$\int_a^b f(x)dx \approx \frac{b-a}{2n}\left[f(a) + 2f(a+h) + \ldots + 2f(a+(n-1)h) + f(b)\right],\ h = \frac{b-a}{n}.$$

2. ***Формула левых прямоугольников:***

$$\int_a^b f(x)dx \approx h\sum_{i=0}^{n-1} f(x_i),\ h = \frac{b-a}{n},\ x_i = a + ih,\ i = \overline{o, n}.$$

3. ***Формула правых прямоугольников:***

$$\int_a^b f(x)dx \approx h\sum_{i=1}^{n} f(x_i),\ h = \frac{b-a}{n},\ x_i = a + ih,\ i = \overline{o, n}.$$

4. ***Формула средних прямоугольников:***

$$\int_a^b f(x)dx \approx h\sum_{i=0}^{n-1} f\left(x_i + \frac{h}{2}\right),\ h = \frac{b-a}{n},\ x_i = a + ih,\ i = \overline{o, n-1}.$$

5. ***Нижняя сумма Дарбу:***

$$\int_a^b f(x)\,dx \approx h \cdot \sum_{i=0}^{n-1} min\{f(x_i), f(x_{i+1})\},$$
$$h = \frac{b-a}{n}, x_i = a + ih, i = \overline{0, n-1}.$$

6. ***Верхняя сумма Дарбу:***

$$\int_a^b f(x)\,dx \approx h \cdot \sum_{i=0}^{n-1} max\{f(x_i), f(x_{i+1})\},$$
$$h = \frac{b-a}{n}, x_i = a + ih, i = \overline{0, n-1}.$$

Пример задания и его выполнения

Формулировка примера задания. Используя квадратурную формулу левых прямоугольников (2), вычислить приближенное значение интеграла $\int_a^b sin(x) \cdot e^x dx$ с точностью ε.

Описание используемых функций. Функция `leftRectangle` с прототипом:

```
double leftRectangle (double LowerLimit,
                      double UpperLimit,
                      unsigned k);
```

возвращает значение интеграла, посчитанное по формуле левых прямоугольников для `k` разбиений отрезка интегрирования.

Функция `sinExp` с прототипом `double sinExp(double x);` возвращает значение подынтегральной функции $sin(x) \cdot e^x$ при заданном значении x.

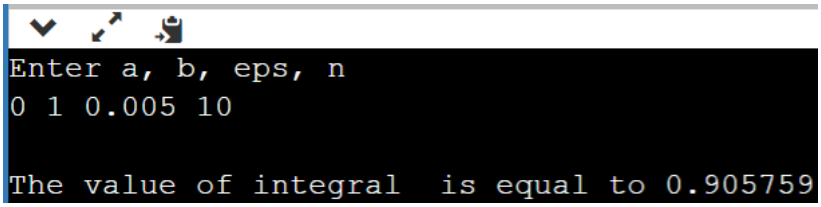
Программа.

main.cpp

```
1  #include <iostream>
2  #include <cmath>
3  using namespace std;
4
5  double leftRectangle(double, double, int);
6  double sinExp(double);
7
8  int main(void) {
9      double a, b, previous, next = 1, eps;
10     int n;
```

```
11
12      while(true) {
13          cout << "Enter a, b, eps, n"
14               << endl;
15          cin >> a >> b >> eps >> n;
16          if ( (a < b) && (n > 0)
17               && (eps < 1) && (eps > 0) ) {
18              break;
19          }
20          cout << endl
21               <<"Parameters are incorrect!!!"
22               << endl << "Try again!!!"
23               << endl;
24      }
25
26      while (fabs(previous - next) > eps) {
27          previous = leftRectangle (a, b, n);
28          next = leftRectangle (a, b, 2 * n);
29          n = 2 * n ;
30      }
31
32      cout << endl << "The value of integral "
33           << " is equal to " << previous;
34      return 0;
35  }
36
37  double leftRectangle (double a,
38                        double b, int k) {
39      double step = (b - a) / k;
40
41      double integral = 0;
42      double t = a;
43      while ( t < b) {
44          integral = integral + sinExp(t);
45          t = t + step;
46      }
47      return step * integral;
48  }
49
50  double sinExp(double x) {
51      return sin(x) * exp(x);
52  }
```

Результат работы программы:



Варианты индивидуальных заданий

С помощью функции и ***двух*** вариантов квадратурных формул вычислить определенных интеграл с заданной точностью (Таблица 7).

Таблица 7 – Варианты подынтегральных функций и квадратурных формул

№	$f(x)$	Квадратурные формулы
1	$\sqrt{x}$	(1), (2)
2	e^{x^2}	(2), (3)
3	$1/(1+x)$	(3), (4)
4	$1/(1+x^3)$	(4), (5)
5	e^{-x^2}	(1), (2)
6	$e^{1/x}$	(2), (3)
7	$\sin x/x$	(3), (4)
8	$\sin x/x$	(4), (5)

№	$f(x)$	Квадратурные формулы
9	$\sin x^2$	(1), (2)
10	$\cos x^2$	(2), (3)
11	$1/1-x+x^2$	(3), (4)
12	$1/\sqrt{x^4}$	(4), (5)
13	$1/x$	(1), (2)
14	$\ln x$	(2), (3)
15	$\sqrt{x}\cos x$	(3), (4)
16	$\sin(x)/(1+x^2)$	(4), (5)

Решение нелинейных уравнений

Обобщенная формулировка задания. Решить нелинейное уравнение вида $f(a)=0$ с параметром с заданной точностью ε, используя предлагаемые итерационные методы. Предварительно провести графическое отделение одного из корней, т.е. найти отрезок, на котором существует единственный корень. Требуемую точность и начальное приближение ввести с клавиатуры.

Использовать один из следующих итерационных алгоритмов нахождения решения на отрезке $[a\,;b\,]$:

1. ***Метод деления отрезка пополам*** (метод дихотомии). Краткое описание метода для отрезка $[a\,;b\,]$, содержащего единственный корень (пусть для определенности $f(a)<0,\ f(b)>0$):
   a. в качестве начального приближения корня $\tilde{x}$ принимается середина этого отрезка, т.е. $x_0=(a+b)/\,2$;
   b. далее исследуем значение функции $f(x)$ на концах отрезков $[a\,;x_0\,]$ и $[\,x_0;b\,]$:
      i. тот из отрезков, на концах которого $f(x)$ принимает значения разных знаков, содержит искомый корень;
      ii. выбранный на предыдущем шаге отрезок принимаем в качестве нового отрезка для продолжения деления, а вторую половину исходного отрезка $[a;\ b]$ отбрасываем;
   c. в качестве первой итерации нахождения корня принимаем середину нового отрезка и т. д.;
   d. если длина полученного отрезка становится меньше наперед заданной погрешности, т.е. $|\,b-a\,|<\varepsilon$, то вычисления прекращаются.
2. ***Метод простых итераций***:

$$x_n=x_{n-1}-\frac{f(x_{n-1})}{M},\text{ где } M>\max_{x\in[a,b]}|f'(x)|, n=\overline{0,\infty}, x_0=a.$$

3. Метод Ньютона:

$$x_{n+1}=x_n-\frac{f(x_n)}{f'(x_n)}, n=0,1,\ldots,\ x_0\in[a,b].$$

4. ***Модифицированный метод Ньютона***:

$$x_{n+1}=x_n-\frac{f(x_n)}{f'(x_0)}, n=0,1,\ldots,\ x_0\in[a,b].$$

5. ***Метод секущих***:

$$x_{n+1}=x_n-\frac{(x_n-x_{n-1})f(x_n)}{f(x_n)-f(x_{n-1})}, n=0,1,\ldots\ x_0,x_1\in[a,b].$$

Пример задания и его выполнения

Формулировка примера задания. Решить методом Ньютона с точностью ε нелинейное уравнение $3\cdot x-cos(x)-1$, предварительно отделив один корень.

Описание используемых функций. Функция `newtonMethod` с прототипом:

```
double newtonMethod(double x0, double eps);
```

возвращает решение нелинейного уравнения для начального приближения `x0`, посчитанного по формуле Ньютона с точностью `eps`.

Функция `function` с прототипом: `double function(double);` возвращает значение функции $f(x) = 3 \cdot x - cos(x) - 1$ при заданном значении x.

Функция `derivative` с прототипом `double derivative (double);` возвращает значение производной функции $3 \cdot x - cos(x) - 1$ при заданном значении x.

Программа.

main.cpp

```
#include <iostream>
#include <cmath>
using namespace std;

//прототипы функций
double newtonMethod (double, double);
double function(double);
double derivative(double);

int main(void) {
    double x0, epsilon;

    while(true) {
        cout << "Введите начальное прибл. и точн."
             << endl;
        cin >> x0 >> epsilon;
        if ( (epsilon < 0.5) && (epsilon > 0) ) {
            break;
        }
        cout << "Точность не верна."
             << "Попробуйте еще раз!!!" << endl;
    }
```

```
23      cout << "Корень равен = "
24           << newtonMethod(x0, epsilon);
25      return 0;
26  }
27
28  double newtonMethod (double x0, double eps) {
29      double prev = x0;
30      double next = prev -
31                    function(prev)/derivative(prev);
32      while (fabs(prev - next) >= eps) {
33          prev = next;
34          next = prev -
35                 function(prev)/derivative(prev);
36      }
37      return prev;
38  }
39
40  double function(double x) {
41      return 3 * x - cos(x) - 1;
42  }
43
44  double derivative (double x) {
45      return 3 + sin(x);
46  }
```

Результат работы программы:

```
Введите начальное прибл. и точн.
1
0.005
Корень равен = 0.607121
```

Варианты индивидуальных заданий

С помощью функции и ***двух*** вариантов итерационных методов вычислить с заданной точностью корни для серии уравнений с переменным параметром α, изменяющемся на заданном интервале (Таблица 8).

Таблица 8 – Варианты нелинейных уравнений и итерационных методов

№	$f(x)$	Интервал для α	Шаг изменения α	Итерационный метод
1	$x^2\cos(2x)+1+\alpha$	-1:1	0.1	(1), (3)
2	$2x-\alpha\cos(x)$	0:1	0.1	(2), (4)
3	$(x-1+\alpha)^3+0.5x$	0:1	0.1	(1), (5)
4	$\alpha x-\cos(x)-1$	3:4	0.1	(2), (3)
5	$x^4+\alpha x-1$	1:5	1	(1), (4)
6	$x^4+\cos(x+\alpha)-\alpha$	3:7	0.5	(2), (5)
7	$3x-\sin(\alpha x)$	1:5	1	(1), (3)
8	$\cos(\alpha x)-1-\alpha x^3$	1:5	1	(2), (4)
9	$\alpha x-\ln(x)-5$	2:4	0.5	(1), (5)
10	$\sin(x+\alpha)-x+1$	1:5	1	(2), (3)
11	$\cos(1/x)-(x+1)^2+\alpha$	5:10	1	(1), (4)
12	$\alpha x^3-\sin(x+\alpha)$	0:1	0.1	(2), (5)
13	$\sin(x)-6x-\alpha$	3:10	1	(1), (3)
14	$x^2+\cos(x^2+\alpha)-\alpha$	3:7	0.5	(2), (4)
15	$x-\sin(x)-\alpha$	2:3	0.1	(1), (5)
16	$\alpha x+(\cos(x))^2$	1:5	1	(2), (3)

Указатели на функции

Обобщенная формулировка задания. Выполнить задания обоих предыдущих тем: «Приближенное вычисление определенных интегралов» и «Решение нелинейных уравнений», используя в функциях в качестве параметров указатели на функции:

- при вычислении интегралов – указатели на подынтегральную функцию и функцию квадратурной формулы передаются как параметры в функцию двойного пересчета, т.е. должно использоваться ***два*** указателя на функцию (в соответствии с примером);
- при решении нелинейных уравнений – ***один*** указатель на функцию решаемого нелинейного уравнения передается как параметр в функцию итерационного метода.

Замечание.

Как и заданиях по темам «Приближенное вычисление определенных интегралов» и «Решение нелинейных уравнений» проект должен содержать ***по два метода*** *из обеих указанных тем.*

Пример задания и его выполнения

Формулировка примера задания. Используя квадратурную формулу левых прямоугольников, вычислить приближенное значение интеграла $\int_a^b \sin x \cdot e^x dx$ с точностью ε.

С помощью `typedef` определены вспомогательные псевдонимы типов указателей на функцию:

```
typedef double(*integralFunction)(double);
typedef double(*quadratureFormula)(double ,
                                   double ,
                                   int ,
                                   integralFunction);
```

Описание используемых функций. Функция `leftRectangle` с прототипом

```
double leftRectangle(double a, double b,
                     unsigned n, IntegralFunction f);
```

возвращает значение интеграла функции `f` на отрезке от `a` до `b`, посчитанное по формуле левых прямоугольников для `n` разбиений отрезка интегрирования.

Функция `doubleCounting` с прототипом:

```
double doubleCounting (double a, double b,
                       double eps, int n,
                       integralFunction f,
                       quadratureFormula q);
```

возвращает значение интеграла функции `f` на отрезке от `a` до `b`, посчитанное по квадратурной формуле `Q`, начиная с `Number` разбиений отрезка интегрирования, с точностью `eps` по формуле двойного пересчета.

Функция `sinExp()` с прототипом `double sinExp(double x);` возвращает значение подынтегральной функции $\sin x \cdot e^x$ при заданном значении `x`.

Программа.

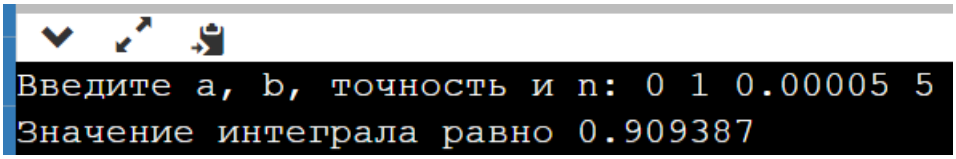
main.cpp

```
#include <iostream>
#include <cmath>
using namespace std;

//определение псевдонимов типов
typedef double(*integralFunction)(double);
typedef double(*quadratureFormula)(double,
                                   double,
                                   int,
                                integralFunction);

//прототипы
double sinExp(double);
double leftRectangle(double,
                     double,
                     int ,
                     integralFunction);

double doubleCounting(double,
                      double,
                      double,
                      int,
                      integralFunction,
                      quadratureFormula);

int main(void) {
    double a, b, eps;
    int n;

    while(true) {
        cout << "Введите a, b, точность и n: ";
        cin >> a >> b >> eps >> n;
        if ( (a < b) && (n > 0)
             && (eps < 0.5) && (eps > 0)) {
```

```
35              break;
36          }
37          cout << "Параметры не корректны."
38               << " Попробуйте еще раз!!!"
39               << endl;
40      }
41      cout<< "Значение интеграла равно "
42          << doubleCounting(a, b, eps, n, sinExp,
43                                    leftRectangle)
44          << endl;
45
46      return 0;
47  }
48
49  double doubleCounting(double a,
50                        double b,
51                        double eps,
52                        int n,
53                        integralFunction f,
54                        quadratureFormula formula) {
55      double prev, next;
56      do {
57          prev = formula(a, b, n, f);
58          next = formula(a, b, 2* n, f);
59          n = 2 * n ;
60      }
61      while ( fabs(prev - next) > eps);
62      return prev;
63  }
64
65  double leftRectangle(double a,
66                       double b,
67                       int n,
68                       integralFunction f) {
69      double result = 0;
70      double h = (b - a) / n;
71
72      for(double x = a; x < b; x += h) {
73          result = result + f(x);
74      }
75      return h * result;
76  }
```

```
77
78  double sinExp(double x) {
79      return sin(x)*exp(x);
80  }
```

Результат работы программы:



Варианты индивидуальных заданий

Переделать задания обоих предыдущих тем: «Приближенное вычисление определенных интегралов» и «Решение нелинейных уравнений» на использование указателей на функции.

Требования к выполнению:

- в переделанной работе по приближенному вычислению интегралов должно использоваться ***два*** указателя на функцию (в соответствии с примером);
- в работе по итерационным методам решения нелинейных уравнений – только ***один*** указатель на функцию.

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Общие требования к выполнению заданий этого раздела

Несмотря на то, что в примерах выполнения заданий приводится код проекта состоящего, как и ранее из ***одного*** файла, для получения оценки `10 «десять»` баллов за выполняемое задание студенты должны ***самостоятельно*** создать ***многофайловый*** проект в соответствии с материалами лекций и разобранным в них примером по использованию конструктора копирования. Пример проекта доступен по ссылке URL: https://www.onlinegdb.com/edit/LLc0F8Vo9 (дата доступа 02.02.2023).

Если в соответствии с индивидуальными заданиями этой части будет создан ***однофайловый*** проект, то это будет оценено как `8 «восемь»`.

Замечание.

*Для демонстрации выполнения задания структуры и классы из индивидуальных заданий должны содержать не менее **шести** полей.*

Массивы структур

Обобщенная формулировка задания. Выбрать предметную область для базы данных и предложить структуру для описания отдельных записей базы данных. Выбранная структура должна иметь не менее пяти полей (элементов) двух и более типов. Для выбранной базы данных написать ***три*** функции:

1. функцию формирования в куче одномерного массива структур заранее заданной размерности, значения которых вводятся с клавиатуры;
2. функцию просмотра содержимого массива структур;
3. функцию поиска и вывода на экран структуры (структур) с заданным значением элемента.

Пример выполнения задания

Формулировка примера задания. На основе предметной области «Человек» (фамилия, имя, отчество, пол, возраст) создать массив структур в куче. Вывести на экран содержимое массива. Сделать выборку в исходном массиве согласно полу и возрастному диапазону.

Программа.

main.cpp

```
#include <iostream>
#include <cstring>
using namespace std;

const int STR_LEN = 25;

struct Person {
    char firstName[STR_LEN],
         secondName[STR_LEN],
         lastName[STR_LEN];
    int age;
    char gender [STR_LEN];
};

// прототипы
Person initPerson();
void initArray(Person *, int );
void displayArray(Person* , int );
void displayChoise(Person*, int,
                   char*, int, int);
void displayPerson(Person );

int main() {
    int n;
    char genderTag[STR_LEN];
    int lowAge, upperAge;

    cout << "\nEnter the number of persons:";
    cin >> n;
    Person* arrayStruct = new Person[n];

    initArray(arrayStruct, n);

```

```
34      cout<<"\nThe list of persons: \n";
35      displayArray(arrayStruct, n);
36
37      cout<<"\nEnter the gender-tag: ";
38      cin.getline(genderTag, STR_LEN);
39
40      cout << "\nEnter the boundary of age: ";
41      cin >> lowAge >> upperAge;
42
43      cout<<"\n\nThe list of choise: \n";
44      displayChoise(arrayStruct, n,
45                    genderTag,
46                    lowAge, upperAge);
47
48      delete []arrayStruct;
49      return 0;
50  }
51
52  Person initPerson() {
53      Person man;
54
55      cout << "\nEnter first name:";
56      cin.getline(man.firstName, STR_LEN);
57
58      cout << "Enter second name:";
59      cin.getline(man.secondName, STR_LEN);
60
61      cout << "Enter last name:";
62      cin.getline(man.lastName, STR_LEN);
63
64      cout << "Enter age:";
65      cin >> man.age;
66
67      cout << "Enter Gender:";
68      cin.ignore(STR_LEN,'\n');
69      cin.getline(man.gender, STR_LEN);
70      return man;
71  }
72
73  void initArray(Person* array, int n) {
74      int i;
75      cin.ignore(STR_LEN,'\n');
76      for( i = 0; i < n; i++) {
77          cout << "\nEnter the information "
78               << "about " << i + 1
```

```
79              << "-th"<< endl;
80          array[i] = initPerson();
81      }
82  }
83
84  void displayArray (Person* array, int n) {
85      int i;
86      for( i = 0; i < n; i++ ) {
87          displayPerson(array[i]);
88      }
89  }
90
91  void displayChoise(Person* a, int n,
92                  char* tag, int lowAge,
93                  int upperAge) {
94      int i;
95      for( i = 0; i < n; i++ ) {
96          if ( !strcmp(a[i].gender, tag) &&
97               a[i].age <= upperAge &&
98               a[i].age >= lowAge) {
99              displayPerson(a[i]);
100         }
101     }
102 }
103
104 void displayPerson (Person man) {
105     cout << man.firstName << " "
106          << man.secondName << " "
107          << man.lastName<< " "
108          << man.age
109          << " " << man.gender<< endl;
110 }
```

Результат работы программы:

```
Введите число людей в массиве: 2

Введите информацию о 1-ом человеке

Введите имя: Ренато
Введите отчество: Августович
Введите фамилию: Буцо
Введите возраст: 50
```

```
Введите пол: м

Введите информацию о 2-ом человеке

Введите имя: Иван
Введите отчество: Иванович
Введите фамилию: Иванов
Введите возраст: 35
Введите пол: м

Массив людей:
Ренато Августович Буцо 50 м
Иван Иванович Иванов 35 м

Введите пол для выборки: м

введите границы возраста: 20
40

Выборка из массива:
Иван Иванович Иванов 35 м
```

Варианты индивидуальных заданий

Создать массив структур и выполнить алгоритмические действия перечисленные в обобщенной формулировке задания и приведенные в примере:

1. «Школьник»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); школа; класс.
2. «Студент»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); ВУЗ; курс; группа; средний бал; специальность.
3. «Покупатель»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); номер кредитной карточки; банковского счета.
4. «Пациент»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом,

квартира); номер больницы; отделение; номер медицинской карты; диагноз; группа крови.

5. «Владелец автомобиля»: фамилия; имя; отчество; номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира) марка автомобиля; номер автомобиля; номер техпаспорта.
6. «Военнослужащий»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); национальность; дата рождения (год, месяц число); должность; звание.
7. «Рабочий»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); национальность; дата рождения (год, месяц число); № цеха; табельный номер; образование; год поступления на работу.
8. «Владелец телефона»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); № телефона.
9. «Абитуриент»: фамилия; имя; отчество; пол; национальность; дата рождения (год, месяц число); домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); оценки по экзаменам; проходной балл.
10. «Государство»: название страны; столица; государственный язык; население; площадь территории; денежная единица; государственный строй; глава государства.
11. «Автомобиль»: марка; цвет; серийный номер; регистрационный номер; год выпуска; год техосмотра; цена.
12. «Товар»: наименование; стоимость; срок хранения; сорт; дата выпуска; срок годности.
13. «Кинолента»: название; режиссер (фамилия; имя); год выхода; страна; стоимость; доход; прибыль.
14. «Рейс»: марка автомобиля; номер автомобиля; пункт назначения; грузоподъемность (в тоннах); стоимость единицы груза; общая стоимость груза.
15. «Книга»: название; автор (фамилия; имя); год выхода; издательство; себестоимость; цена; прибыль.
16. «Здание»: адрес; тип здания; количество этажей; количество квартир; срок эксплуатации; срок до капитального ремонта (25 лет - срок эксплуатации).

Массивы объектов. Перегрузка ввода/вывода.

Обобщенная формулировка задания. Выбрать предметную область для базы данных и предложить класс для описания отдельных записей базы данных. Выбранный класс должен иметь не менее пяти полей (элементов) двух и более типов. Для выбранного класса в качестве методов создать обычный набор ***геттеров*** и ***сеттеров***.

Кроме того:

- перегрузить:
  - операцию ввода (`cin`) для заданного класса;
  - операцию вывода (`cout`) для заданного класса;
- написать следующие функции:
  - функцию формирования одномерного массива объектов в куче заданной с клавиатуры размерности (значения свойств объектов вводятся с клавиатуры);
  - функцию просмотра содержимого массива объектов;
  - функцию поиска и вывода на экран объектов с заданным значением элемента.

Пример выполнения задания

Формулировка примера задания. На основе предметной области «Человек» (фамилия, имя, отчество, пол, возраст) создать массив объектов в куче. Вывести на экран содержимое массива. Сделать выборку в исходном массиве согласно полу и возрастному диапазону.

Программа.

main.cpp

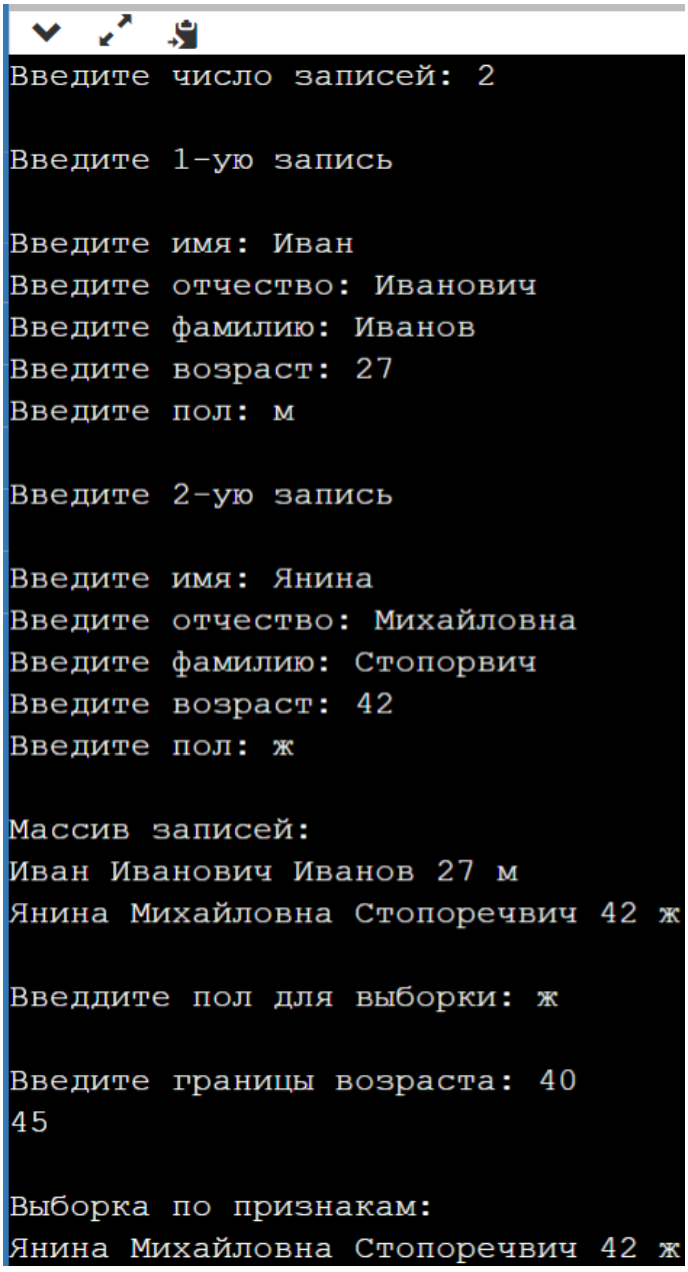
```
1  #include <iostream>
2  #include <cstring>
3  using namespace std;
4
5  const int STR_LEN = 25;
6
7  class Person {
8      char firstName[STR_LEN],
```

```
 9          secondName[STR_LEN],
10          lastName[STR_LEN];
11      int age;
12      char gender[STR_LEN];
13
14  public:
15      //геттеры
16      char * getFirstName() {return firstName;}
17      char * getSecondName() {return secondName;}
18      char * getLastName() {return lastName;}
19      int getAge() {return age;}
20      char * getGender() {return gender;}
21
22      //сеттеры
23      void setFirstName(char * firstName) {
24          strcpy(this->firstName, firstName);
25      }
26      void setSecondName(char * secondName) {
27          strcpy(this->secondName, secondName);
28      }
29      void setLastName(char * lastName) {
30          strcpy(this->lastName, lastName);
31      }
32      void setAge(int age) {
33          this->age = age;
34      }
35      void setGender(char * gender) {
36          strcpy(this->gender, gender);
37      }
38  };
39
40  //перегрузка cin и cout
41  istream &operator>>(istream &stream,
42                      Person &obj) {
43      char str[STR_LEN];
44
45      cout<<"\nВведите имя: ";
46      cin.getline(str, STR_LEN);
47      obj.setFirstName(str);
48
49      cout<<"Введите отчество: ";
50      cin.getline(str, STR_LEN);
51      obj.setSecondName(str);
52
```

```
53      cout<<"Введите фамилию: ";
54      cin.getline(str, STR_LEN);
55      obj.setLastName(str);
56
57      int n;
58      cout << "Введите возраст: ";
59      cin >> n;
60      obj.setAge(n);
61
62      cout<<"Введите пол: ";
63      cin.ignore(STR_LEN,'\n');
64      cin.getline(str, STR_LEN);
65      obj.setGender(str);
66      return stream;
67  }
68
69  ostream &operator<<(ostream &stream,
70                                  Person obj) {
71      stream << obj.getFirstName() << " "
72             << obj.getSecondName() << " "
73             << obj.getLastName() << " "
74             << obj.getAge() << " "
75             << obj.getGender() << endl;
76      return stream;
77  }
78
79  void initArray(Person*, int);
80  void displayArray (Person* , int);
81  void displayChoise (Person* , int,
82                              char*, int, int);
83
84  int main() {
85      int n;
86      char genderTag [STR_LEN];
87      int lowAge, upperAge;
88
89      cout << "Введите число записей: ";
90      cin >> n;
91      Person* array = new Person[n];
92
93      initArray(array, n);
94
95      cout<<"\nМассив записей: \n";
96      displayArray(array, n);
```

```
97
98      cout<<"\nВведдите пол для выборки: ";
99      cin.getline(genderTag, STR_LEN);
100
101     cout<<"\nВведите границы возраста: ";
102     cin >> lowAge >> upperAge;
103
104     cout<<"\nВыборка по признакам: \n";
105     displayChoise(array, n, genderTag,
106                           lowAge, upperAge);
107
108     delete []array;
109     return 0;
110 }
111
112 void initArray(Person* array, int n) {
113     int i;
114     cin.ignore(STR_LEN,'\n');
115     for( i = 0; i < n; i++) {
116         cout << flush
117              <<"\nВведите "
118              << i + 1 << "-ую запись\n";
119         cin >> array[i];
120     }
121 }
122
123 void displayArray (Person* array, int n) {
124     int i;
125     for( i = 0; i < n; i++ ) {
126         cout << array[i];
127     }
128 }
129
130 void displayChoise(Person* a, int n,
131         char* tag, int low, int upper) {
132     for(int i = 0; i < n; i++ ) {
133         if( !strcmp(a[i].getGender(),tag)
134             && a[i].getAge() <= upper
135             && a[i].getAge() >= low)
136         cout << a[i];
137     }
138 }
```

Результат работы программы:



Варианты индивидуальных заданий

Создать массив объектов заданного класса и выполнить алгоритмические действия перечисленные в обобщенной формулировке задания и приведенные в примере:

1. Класс «Учебная дисциплина» и его возможные свойства: название; продолжительность в часах лекционного курса; продолжительность в часах практических занятий; продолжительность в часах лабораторных занятий; количество тематических разделов, тип (общеобразовательная или дисциплина специализации).

2. Класс «Автобус» и его возможные свойства: фирма производитель, классификация по назначению, классификация по длине, классификация по компоновке, классификация по типу двигателя, параметры булевского типа, указывающие наличие: пневмоподвески, автоматических дверей, пандуса для инвалидов, ремней безопасности для крепления грузов.
3. Класс «Трамвай» и его возможные свойства: фирма производитель, разновидность по решаемым задачам, скорость передвижения, компоновка трамвая, схема управления двигателем, вместимость (количество пассажиров), требуемая инфраструктура.
4. Класс «Троллейбус» и его возможные свойства: фирма производитель, разновидность по решаемым задачам, компоновка шасси, тип тягового электродвигателя, система управления двигателем, наличие системы автономного хода, тип тормозной системы, тип вентиляции салона.
5. Класс «Мебель» и его возможные свойства: название, комплектность, фирма-производитель, год изготовления, назначение, материал изготовления, способ изготовления, стиль.
6. Класс «Подвижной состав автомобильного транспорта» и его возможные свойства: тип транспортного средства, фирма изготовитель, год выпуска, дата последнего технического осмотра, тип двигателя, назначение, проходимость, приспособленность к климатическим условиям, характер использования.
7. Класс «Проездной билет» и его возможные свойства: город, тип (разовый или многоразовый (абонементный)), носитель (БСК или бумажный), ограничение времени действия, вид транспорта, ограничение валидности по количеству поездок.
8. Класс «Рейс» и его возможные свойства: тип транспорта, класс обслуживания, пункт отправления, время отправления, пункт назначения, время прибытия, время в пути, количество остановок.
9. Класс «Дерево» и его возможные свойства: вид листьев, срок жизни листьев, род дерева, распространение, форма кроны, наличие плодов, характеристика корневой системы.
10. Класс «Медицинские расходные материалы» и его возможные свойства: вид, назначение, указание о стерильности, материал изготовления, фирма изготовитель, дата изготовления, срок годности.
11. Класс «Фрукт» и его возможные свойства: вид, вкус, размер, цвет кожуры, цвет мякоти, региональная принадлежность, сезонность.
12. Класс «Хлебобулочное изделие» и его возможные свойства, классифицирующие объект по: виду муки, рецептуре, способу выпечки, типу теста, по массе, группе изделия.

13. Класс «Электростанция» и его возможные свойства, классифицирующие объект в зависимости от: источника энергии (в частности, вида топлива), типа силовой установки, мобильности, степени применения (распространенности), мощности генерации, срока службы, ожидаемой стоимости сооружения.
14. Класс «Программист» и его возможные свойства: ФИО, ВУЗ по диплому, специальность по диплому, языки программирования, опыт работы, профессиональный уровень, перечисление проектов в работе над которыми принимал участие.
15. Класс «Животное» и его возможные свойства: название, срок жизни, тип, класс, отряд, семейство, род, вид.
16. Класс «Принтер» и его возможные свойства и его возможные свойства, классифицирующие объект по: названию, фирме-производителю, стране происхождения, возможности печати графической информации, конструктивному устройству и принципу формирования изображения, количеству выдаваемых цветов, типу интерфейса подключения.

Создание очереди объектов. Перегрузка преобразования типов

Обобщенная формулировка задания. В соответствии с индивидуальным заданием необходимо создать программу для связывания объектов в очередь.

Требования к выполнению задания:

- память под очередной элемент списка следует выделять в куче, для текстовых полей использовать тип `std::string`;
- для заданного класса перегрузить операцию преобразования информационной части узла очереди в тип `std::string`;
- написать следующие методы:
  - метод добавления записи в конец очереди;
  - метод извлечения информационной части и удаления узла из начала очереди.

Замечание.

При демонстрации работоспособности программы, очередь должна иметь не менее 5 узлов с проинициализированной информационной частью.

Пример выполнения задания

Формулировка примера задания. На основе предметной области «Человек» (фамилия, имя, отчество, пол, возраст) создать очередь (`queue`) объектов. Вывести на экран содержимое очереди.

Программа.

```
//файл main.cpp
```

main.cpp | node.h | node.cpp | userQueue.h | userQueue.cpp | per

```cpp
#include <iostream>
#include "node.h"
#include "userQueue.h"
using namespace std;

int main() {
    UserQueue queue;
    queue.push(new Person("Hopckins", "Antony", 1234));
    queue.push(new Person("Buzio", "Renato", 5678));
    queue.push(new Person("Vaskevich", "Andrey", 9102));
    std::cout << queue.frontNpop() << std::endl
              << queue.frontNpop() << std::endl
              << queue.frontNpop() << std::endl
              //демонстрация того, что очередь уже
              //пуста - на экран ничего не выводится
              << queue.frontNpop();
    return 0;
}
```

```
//файл node.h
```

main.cpp | node.h | node.cpp | user

```cpp
#ifndef NODE_h
#define NODE_h

#include "person.h"

class Node {
    //информационная часть будет
    //храниться в куче
    Person* inf;
    Node* next = nullptr;
```

```
11  public:
12      //публичные методы
13      Node(Person*);
14      Person getInf();
15      Node* getNext();
16      void setNext(Node*);
17  };
18
19  #endif //NODE_h
```

//файл node.cpp

main.cpp | node.h | node.cpp | us

```
1   #include "node.h"
2   #include "person.h"
3
4   Node::Node(Person* inf) {
5       this->inf = inf;
6   }
7   Person Node::getInf() {
8       return *inf;
9   }
10  Node* Node::getNext() {
11      return next;
12  }
13  void Node::setNext(Node* nx) {
14      next = nx;
15  }
```

//файл userQueue.h

main.cpp | node.h | node.cpp | userQueue.h

```
1   #ifndef USER_QUEUE_H
2   #define USER_QUEUE_H
3
4   #include "person.h"
5
6   class UserQueue {
7       //начало очереди
8       Node* begin = nullptr;
```

```
 9  public:
10      //прототип конструктора, создающего
11      //пустую очередь
12      UserQueue();
13      //прототип метода, добавляющего
14      //узел в конец
15      void push(Person* inf);
16      //прототип метода, извлекающего
17      //информационную часть с удалением
18      //начала
19      std::string frontNpop();
20      //прототип метода, проверяющего
21      //пуста ли очередь
22      bool empty();
23  };
24  #endif //USER_QUEUE_H
```

//файл userQueue.cpp

node.h | node.cpp | userQueue.h | userQueue.cpp | person.h

```
 1  #include <iostream>
 2  #include "node.h"
 3  #include "userQueue.h"
 4  #include "person.h"
 5
 6  //конструктор, создающий пустую очередь
 7  UserQueue::UserQueue() {};
 8
 9  //метод, добавляющий узел в конец
10  void UserQueue::push(Person* inf) {
11      //если очередь пуста
12      if ( empty() ) {
13          begin = new Node(inf);
14          return;
15      }
16      //если в очереди есть хотя бы один узел
17      Node* current = begin;
18      while(current->getNext() != nullptr) {
19          current = current->getNext();
20      }
21      current->setNext(new Node(inf));
22  }
```

```
23
24  //метод, извлекающий информационную часть
25  //с удалением начала очереди
26  std::string UserQueue::frontNpop() {
27      std::string result = ""; //можно "Очередь пуста"
28      if ( !empty() ) {
29          result = begin->getInf();
30          Node* current = begin;
31          begin = begin->getNext();
32          delete current;
33      }
34      return result;
35  }
36
37  //метод проверяющий пуста ли очередь
38  bool UserQueue::empty() {
39      //узлов в стеке нет
40      if (begin == nullptr) return true;
41      return false;
42  }
```

//файл person.h

userQueue.h | userQueue.cpp | person.h | person.c

```
1   #include <iostream>
2
3   #ifndef PERSON_H
4   #define PERSON_H
5
6   class Person {
7       std::string surname;
8       std::string name;
9       int passportID;
10  public:
11      //прототипы конструкторов
12      Person();
13      Person(std::string, std::string, int);
14      //прототипы геттеров
15      std::string getSurname();
16      std::string getName();
17      int getPassportID();
```

```
18      //приведение типа Person к строке
19      operator std::string() const;
20  };
21
22  #endif //PERSON_H
```

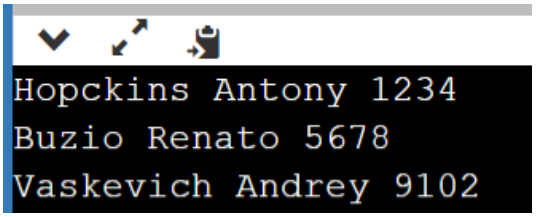
//файл person.cpp

userQueue.cpp | person.h | person.cpp

```
1   #include <iostream>
2   #include "person.h"
3   using namespace std;
4
5   //конструктор без параметров
6   Person::Person() {};
7
8   //конструктор с параметрами
9   Person::Person(string surname,
10                 string name,
11                 int id) {
12      this->surname = surname;
13      this->name = name;
14      passportID = id;
15  }
16  //геттеры
17  string Person::getSurname() {
18      return surname;
19  }
20  string Person::getName() {
21      return name;
22  }
23  int Person::getPassportID() {
24      return passportID;
25  }
26  //приведение типа Person к строке
27  Person::operator string() const {
28      string result = surname + " "
29                    + name + " "
```

```
30                  + to_string(passportID);
31      return result;
32  }
```

Результаты работы программы:



Замечание.

Пример многофайлового проекта по созданию очереди доступен по ссылке URL https://www.onlinegdb.com/edit/hZpf-iRP8 (дата доступа 05.03.2023). Однако он ***не полностью*** *соответствует заданию и приведенному выше примеру.*

Варианты индивидуальных заданий

1. «Покупатель»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); номер кредитной карточки; банковского счета.
2. «Пациент»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); номер больницы; отделение; номер медицинской карты; диагноз; группа крови.
3. «Владелец автомобиля»: фамилия; имя; отчество; номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира) марка автомобиля; номер автомобиля; номер техпаспорта.
4. «Военнослужащий»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); национальность; дата рождения (год, месяц число); должность; звание.
5. «Рабочий»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); национальность; дата рождения (год, месяц число); № цеха; табельный номер; образование; год поступления на работу.

6. «Владелец телефона»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); № телефона.
7. «Абитуриент»: фамилия; имя; отчество; пол; национальность; дата рождения (год, месяц число); домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); оценки по экзаменам; проходной балл.
8. «Государство»: название страны; столица; государственный язык; население; площадь территории; денежная единица; государственный строй; глава государства.
9. «Автомобиль»: марка; цвет; серийный номер; регистрационный номер; год выпуска; год техосмотра; цена.
10. «Товар»: наименование; стоимость; срок хранения; сорт; дата выпуска; срок годности.
11. «Кинолента»: название; режиссер (фамилия; имя); год выхода; страна; стоимость; доход; прибыль.
12. «Рейс»: марка автомобиля; номер автомобиля; пункт назначения; грузоподъемность (в тоннах); стоимость единицы груза; общая стоимость груза.
13. «Книга»: название; автор (фамилия; имя); год выхода; издательство; себестоимость; цена; прибыль.
14. «Здание»: адрес; тип здания; количество этажей; количество квартир; срок эксплуатации; срок до капитального ремонта (25 лет - срок эксплуатации).
15. «Школьник»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); школа; класс.
16. «Студент»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); ВУЗ; курс; группа; средний бал; специальность.

Абстрактные классы. Динамическая диспетчеризация методов

Пример выполнения задания

Формулировка примера задания. Создать абстрактный базовый класс (предметная область) `Base` и производные классы `FirstDerived`,

`SecondDerived`. Организовать массив в куче из четырех объектов классов наследников, проинициализировать его и вывести содержимое на экран.

Замечание.

- *поскольку в соответствии с индивидуальным заданием базовый класс будет иметь **три** наследника, то в отличие от примера при выполнении задания, необходимо создать и сохранить в массиве **шесть** объектов производных классов (по два объекта на один класс наследник) в произвольном порядке;*
- *каждый производный класс должен иметь суммарно не менее шести свойств;*
- *разделение по числу свойств, принадлежащих базовому классу и наследникам, не имеет значения;*
- *как показано в примере, базовый класс может вовсе не иметь свойств, а производные классы всегда должны иметь свойства.*

Программа.

main.cpp

```
#include<iostream>
using namespace std;

//в примере два насленика, поэтому
//объектов четыре, а в задании три
//наследника, следовательно объектов шесть
const int NUMBER_OBJECTS = 4;

//базовый абстрактный класс
class Base {
    public:
    virtual std::string get() = 0;
    virtual ~Base() {};
};

//первый производный класс
class FirstDerived : public Base {
    std::string name;
    int phone;
    public:
    FirstDerived(std::string name, int phone) {
        this->name = name;
```

```
23          this->phone = phone;
24      }
25      std::string get() {
26          return "Клиент: "
27                  + name + std::to_string(phone);
28      }
29  };
30
31  //второй производный класс
32  class SecondDerived : public Base {
33      string color;
34      int id;
35  public:
36      SecondDerived(std::string color, int id) {
37          this->color = color;
38          this->id = id;
39      }
40      std::string get() {
41          return "Цвет машины: "
42                  + color + " "
43                  + std::to_string (id);
44      }
45  };
46
47  //полиморфная (из-за get()) перегрузка вставки
48  //в поток
49  std::ostream& operator << (std::ostream& stream,
50                             Base* obj) {
51      //полиморфное обращение к методу get()
52      stream << obj->get();
53      return stream;
54  }
55
56  //инициализация массива указателями На наследников
57  Base** initPtrArray() {
58      Base** ptr = new Base*[NUMBER_OBJECTS];
59      ptr[0] = new FirstDerived("Ivan", 291111111);
60      ptr[1] = new FirstDerived("Serg", 297777777);
61      ptr[2] = new SecondDerived("Orange", 12);
62      ptr[3] = new SecondDerived("Black", 9);
63      return ptr;
64  }
```

```
65
66  //вывод массива из разных объектов на экран
67  void printPtrArray(Base** ptr) {
68      for(int i = 0; 1 < NUMBER_OBJECTS; i++) {
69          cout << ptr[i] << endl;
70      }
71  }
72
73  int main() {
74      //необходимо использовать указатель
75      //на указатель, т.к. конструктор абстрактного
76      //класса вызвать невозможно
77      Base** ptr = initPtrArray();
78      //вывод на экран элементов массива из объектов
79      //разных классов
80      printPtrArray (ptr);
81      //удаление объектов из кучи
82      for(int i = 0; 1 < NUMBER_OBJECTS; i++) {
83          delete ptr[i];
84      }
85      //удаление самого массива
86      delete []ptr;
87      return 0;
88  }
```

Результат работы программы:

```
Клиент: Ivan 291111111
Клиент: Serg 297777777
Цвет машины: Orange 12
Цвет машины: Black 9
```

Варианты индивидуальных заданий

Каждый производный класс должен иметь не менее шести полей строкового или целочисленного типа:

1. Создать абстрактный базовый класс (предметная область) «Транспортное средство» и производные классы «Автомобиль», «Велосипед», «Повозка».
2. Создать абстрактный базовый класс (предметная область) «Грузоперевозчик» и производные классы «Самолет», «Поезд», «Автомобиль».

3. Создать абстрактный базовый класс (предметная область) «Учащийся» и производные классы «Школьник», «Учащийся ГПТУ» и «Студент».
4. Создать абстрактный базовый класс (предметная область) «Музыкальный инструмент» и производные классы «Ударный», «Струнный», «Духовой».
5. Создать абстрактный базовый класс (предметная область) «Работник фирмы» и производные классы «Менеджер», «Администратор», «Программист».
6. Создать абстрактный базовый класс (предметная область) «Домашнее животное» и производные классы «Собака», «Кошка», «Попугай».
7. Создать абстрактный базовый класс (предметная область) «Садовое дерево» и производные классы «Яблоня», «Вишня», «Груша».
8. Создать абстрактный базовый класс (предметная область) «Единица хранения в библиотеке» и производные классы «Художественная книга», «Научная книга», «Журнал».
9. Создать абстрактный базовый класс (предметная область) «Автомобиль» и производные классы «Легковой автомобиль», «Грузовой автомобиль», «Вездеход».
10. Создать абстрактный базовый класс (предметная область) «Плавсредство» и производные классы «Корабль», «Баржа», «Яхта».
11. Создать абстрактный базовый класс (предметная область) «Электростанция» и производные классы «Теплоэлектростанция», «Гидроэлектростанция», «Атомная станция».
12. Создать абстрактный базовый класс (предметная область) «Городской транспорт» и производные классы «Троллейбус», «Автобус», «Трамвай».
13. Создать абстрактный базовый класс (предметная область) «Коммунальная техника» и производные классы «Уборочная техника», «Мусоровоз», «Автомобиль-разбрасыватель реагентов».
14. Создать абстрактный базовый класс (предметная область) «Торговая сеть» и производные классы «Гипермаркет», «Специализированные магазин», «Рынок».
15. Создать абстрактный базовый класс (предметная область) «Школа» и производные классы «Учитель», «Школьник», «Повар».
16. Создать абстрактный базовый класс (предметная область) «Рабочий» и производные классы «Слесарь», «Электрик», «Монтажник-высотник».

БИБЛИОТЕКА STL. ШАБЛОНЫ

Как и ранее несмотря на то, что в примерах выполнения заданий приводится код проекта состоящего из ***одного*** файла, для получения оценки `10 «десять»` баллов за выполняемое задание студенты должны ***самостоятельно*** создать ***многофайловый*** проект в соответствии с материалами лекций и разобранным в них примером по созданию очереди (URL: https://www.onlinegdb.com/edit/hZpf-iRP8, дата доступа 25.08.2023).

Если в соответствии с индивидуальными заданиями этой части будет создан ***однофайловый*** проект, то это будет оценено как `8 «восемь»`.

Замечание.

*Для демонстрации выполнения задания структуры и классы из индивидуальных заданий должны содержать не менее **шести** полей.*

Обработка строк типа `string`

Выполнение индивидуальных заданий предполагает использование исключительно методов класса `string`. Широкое использование операции ***индексирования*** при выполнении индивидуальных заданий приведет к ***снижению оценки*** на 20%.

Пример задания и его выполнения

Формулировка примера задания. С помощью методов класса `string` в заданной строке определить количество цифр, циклически сдвинуть строку вправо и лево. Результаты подсчета цифр и преобразований вывести на экран.

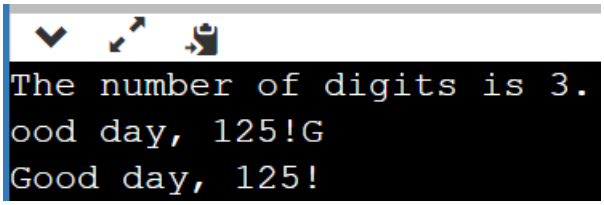
Программа.

main.cpp

```
1  #include <iostream>
2  #include <string>
3
4  //прототипы функций
5  int digitNumber(std::string& str);
```

```cpp
void shiftLeft(std::string&);
void shiftRight(std::string&);

int main() {
    std::string str = "Good day, 125!";

    std::cout << "The number of digits is "
              << digitNumber(str) << "."
              << std::endl;

    shiftLeft(str);
    std::cout << str << std::endl;

    shiftRight(str);
    std::cout << str << std::endl;
    return 0;
}

//подсчет количества цифр
int digitNumber(std::string& str) {
    int number = 0;
    for(char c : str) {
        number += isdigit(c);
    }
    return number;
}


//сдвиг вправо
void shiftLeft(std::string& s) {
    s = s + s[0];
    s.erase(0, 1);
}

//сдвиг влево
void shiftRight(std::string& s) {
    s = s.insert(0,
            std::string(1, s[s.size() - 1]));
    s.erase(s.size() - 1, 1);
}
```

Результаты работы программы:



Варианты индивидуальных заданий

Часть 1. Функции, возвращающие значения

Дана строка ***латинских*** символов, цифр и разделителей (пробелы и знаки препинания). С помощью функций:

1. в строке определить индекс символа с наибольшим кодом;
2. посчитать наибольшее количество подряд идущих гласных в строке, если необходимо, то создать вспомогательный массив из гласных;
3. определить наибольший суммарный код (из стандартной кодовой таблицы) слова в строке, если необходимо, то создать вспомогательный массив из разделителей;
4. в строке посчитать наибольшее количество групп знаков из подряд идущих разделителей, если необходимо, то создать вспомогательный массив из разделителей;
5. в строке посчитать количество строчных согласных, если необходимо, то создать вспомогательный массив из строчных согласных;
6. в строке посчитать сколько вхождений в строку имеет комбинация знаков «согласная буква-разделитель-согласная буква, если необходимо, то создать вспомогательный массив из согласных букв и разделителей;
7. посчитать количество строчных гласных, если необходимо, то создать вспомогательный массив из строчных гласных;
8. посчитать сколько вхождений в строку имеет комбинация букв «гласная буква-разделитель-гласная буква», если необходимо, то создать вспомогательный массив из гласных и разделителей;
9. определить среднюю длину слов в строке, если необходимо, то создать вспомогательный массив из разделителей;
10. определить количество слов длина, которых больше/меньше заданного числа, если необходимо, то создать вспомогательный массив из разделителей;

11. определить количество слов начинающихся и заканчивающихся одной и той же буквой, если необходимо, то создать вспомогательный массив из разделителей;
12. определить длину самого короткого/длинного слова в строке, если необходимо, то создать вспомогательный массив из разделителей;
13. определить номер по порядку самого длинного/короткого слова в строке, если необходимо, то создать вспомогательный массив из разделителей;
14. в строке определить символ с наименьшим кодом в стандартной кодовой таблице;
15. посчитать количество разделителей (пробелы и знаки препинания), если необходимо, то создать вспомогательный массив из разделителей;
16. определить номер по порядку первого слова, начинающегося с гласной буквы, если необходимо, то создать вспомогательный массив из гласных букв и разделителей;
17. посчитать количество заглавных согласных, если необходимо, то создать вспомогательный массив из заглавных согласных букв;
18. определить номер последнего слова, начинающегося с согласной, если необходимо, то создать вспомогательный массив из согласных букв и разделителей;
19. посчитать количество заглавных гласных в строке, если необходимо, то создать вспомогательный массив из заглавных гласных букв;
20. определить длину первого слова, начинающегося с согласной, если необходимо, то создать вспомогательный массив из согласных букв и разделителей;
21. посчитать количество слов в строке, если необходимо, то создать вспомогательный массив из разделителей;
22. посчитать наибольшее количество подряд идущих согласных в строке, если необходимо, то создать вспомогательный массив из согласных.

Часть 2. Удаление символов в строке сдвигом влево

При выполнении задания сСимвол конца `'\0'` строки также сдвигать влево при удалении каждого заданного символа. Удалить из строки:

1. все повторяющиеся гласные (первый в последовательности повторяющихся гласных оставить);
2. все повторяющиеся согласные (первый в последовательности повторяющихся согласных оставлять);
3. все двоеточия, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);

4. все символы из строки между символами с наибольшим и наименьшим кодом в стандартной кодовой таблице; если в строке таких символов несколько, то удалить наибольшую из возможных подстрок;
5. все точки с запятой, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
6. все символы из строки между вторым и последним пробелами. Пробел между словами может остаться только один;
7. все символы между первой и последней запятыми (остается только одна запятая);
8. все знаки между второй и предпоследней гласной буквой (остается только начальная гласная, определяющая строку для удаления).
9. первое слово в строке;
10. все самые длинные слова;
11. все пробелы, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
12. все разделители между словами, первое из которых заканчивается на гласную, а следующее начинается на согласную;
13. все запятые, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
14. все символы, имеющие наименьший код в стандартной кодовой таблице;
15. все точки, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
16. все символы, имеющие наибольший код в стандартной кодовой таблице;
17. все восклицательные знаки, сдвигая строку при удалении каждого заданного знака влево на один символ (забой или `BackSpace`);
18. все самые короткие слова.

Часть 3. Циклически переставить на один символ сдвигом вправо

Понимая, что подстрока – это участок строки ***циклически*** переставить:

1. первое слово в строке;
2. четвертое с конца слово, начинающееся с гласной буквы;
3. последнее слово в строке;
4. второе с конца слово, начинающееся с согласной буквы;
5. второе слово, начинающееся с гласной буквы;
6. третье слово, начинающееся с согласной буквы;
7. подстроку (участок строки), заключенный между символами с максимальным и минимальным кодом из стандартной таблицы;

8. первую по порядку подстроку из подряд идущих цифр (предполагается, что группа из подряд идущих цифр содержит не менее двух цифр, а таких групп не менее одной);
9. символы подстроки, заключенные между третьей гласной и предпоследней согласной;
10. вторую половину строки;
11. символы, имеющие четные индексы;
12. символы первой половины строки;
13. подстроку из трех подряд слов, последним из которых является предпоследнее слово строки;
14. k-ое слово в строке;
15. подстроку из двух подряд слов, начинающейся с третьего по порядку слова;
16. символы, имеющие нечетные индексы.

Часть 4. Зеркально переставить

Понимая, что подстрока – это участок строки ***зеркально*** переставить:

1. символы, имеющие четные индексы;
2. символы первой половины строки;
3. подстроку из трех подряд слов, последним из которых является предпоследнее слово строки;
4. k-ое слово в строке (k вводится с клавиатуры);
5. подстроку из двух подряд слов, начинающейся с третьего по порядку слова;
6. первое слово в строке;
7. четвертое с конца слово, начинающееся с гласной буквы;
8. последнее слово в строке;
9. второе с конца слово, начинающееся с согласной буквы;
10. второе слово, начинающееся с гласной буквы;
11. подстроку (участок строки), заключенный между символами с максимальным и минимальным кодом;
12. первую по порядку подстроку из подряд идущих цифр (предполагается, что группа из подряд идущих цифр содержит не менее двух цифр, а таких групп не менее одной);
13. символы подстроки, заключенные между третьей гласной и предпоследней согласной;
14. символы, имеющие нечетные индексы;
15. вторую половину строки;
16. третье слово, начинающееся с согласной буквы.

Использование контейнера queue для хранения и обработки списка объектов

Обобщенная формулировка задания. В соответствии с индивидуальным заданием необходимо создать программу для создания очереди `queue` из объектов заданного класса. Указанные в задании действия должны быть оформлены в виде отдельных функций.

Перегрузить:

- для выбранного класса операцию преобразования (операция « ( ) ») объекта класса к строке типа `string`;
- `cout` для очереди `queue` с указанным параметром `<ИмяКласса>`.

Написать следующие функции:

1. функцию формирования очереди `queue` заданной с клавиатуры длины (значения полей объекта также вводятся с клавиатуры);
2. функцию просмотра содержимого очереди `queue` из объектов заданного класса;
3. функцию поиска и вывода на экран объекта очереди с заданным значением ***двух*** полей в соответствии со ***второй*** частью индивидуального задания.

Пример задания и его выполнения

Формулировка примера задания. На основе предметной области «Человек» (фамилия, имя, идентификационный номер) создать очередь `queue` из объектов заданного класса. Вывести на экран содержимое очереди. Сделать выборку в исходной очереди согласно имени.

Замечание.

Пример содержит алгоритм чувствительный к регистру ввода строковых полей объекта. Для того чтобы сделать его не чувствительным к регистру следует преобразовать сравниваемые строки перед сравнением или в верхний, или в нижний регистры. Для латиницы перевод строки `str` в верхний регистр можно выполнить с помощью элементарного кода: `for (auto & c: str) c = (char)toupper(c);`*. Но для кириллицы это не работает.*

Программа.

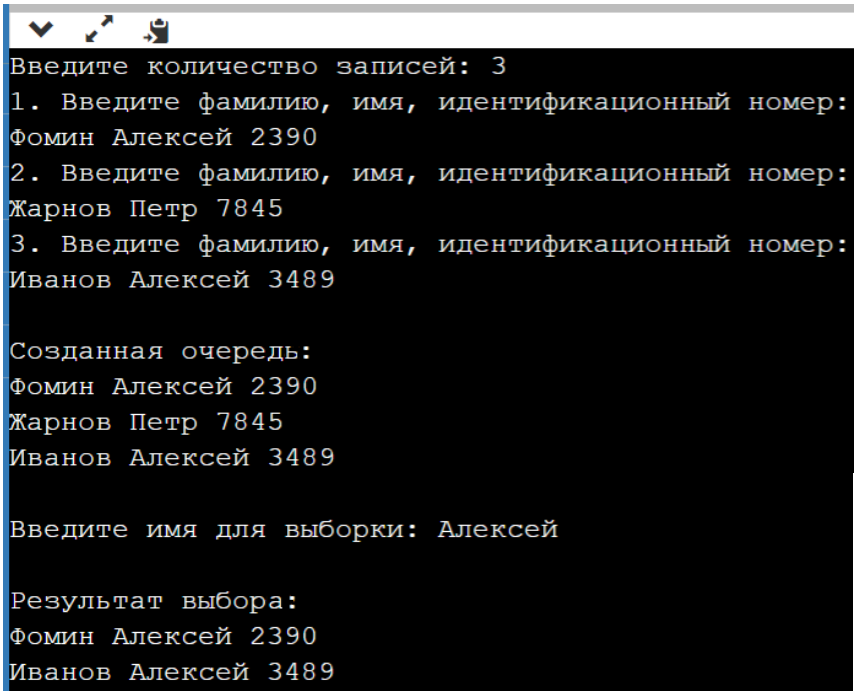
main.cpp

```cpp
#include <iostream>
#include <queue>
using namespace std;

class Person {
    string surname;
    string name;
    int id;
public:
    //прототипы конструкторов
    Person();
    Person(string, string, int);
    //прототипы геттеров
    string getSurname();
    string getName();
    int getId();
    //приведение типа Person к строке
    operator string();
};

//конструктор без параметров
Person::Person() {};

//конструктор с параметрами
Person::Person(string surname,
               string name, int id) {
    this->surname = surname;
    this->name = name;
    this->id = id;
}
//геттеры
string Person::getSurname() {
    return surname;
}
string Person::getName() {
    return name;
}
```

```
38 //приведение типа Person к строке
39 Person::operator string() {
40     string result = surname
41                   + " " + name
42                   + " " + to_string(id);
43     return result;
44 }
45
46 int Person::getId() {
47     return id;
48 }
49
50 //Функция инициализации очереди.
51 //Передача очереди по ссылке.
52 void initQueue(queue<Person>& q) {
53     int n, id;
54     string surname, name;
55     cout << "Введите количество записей: ";
56     cin >> n;
57     for(int i = 0; i < n; i++) {
58         cout << i + 1
59              << ". Введите фамилию, имя, "
60              << "идентификационный номер: "
61              << endl;
62         cin >> surname >> name >> id;
63         q.push(Person(surname, name, id));
64     }
65 }
66
67 //Функция выборки записей по признаку.
68 //Передача очереди по значению из-за pop().
69 void chooseQueue(queue<Person> q, string name) {
70     while( !q.empty() ) {
71         Person buf = q.front();
72         if (buf.getName() == name) {
73             cout << string(buf) << endl;
74         }
75         q.pop();
76     }
77 }
```

```
78
79  //Перегрузка cout для очереди из объектов Person.
80  //Передача очереди по значению из-за pop().
81  ostream& operator << (std::ostream &stream,
82                        queue<Person> q) {
83      while( !q.empty() ) {
84          //последовательность явных
85          //преобразований типов
86          stream << string( Person(q.front()))
87                 << endl;
88          q.pop();
89      }
90      return stream;
91  }
92
93  int main() {
94      //создаем пустую очередь
95      queue<Person> q;
96      //инициализируем очередь значениями
97      initQueue(q);
98      //выводи очередь на экран
99      cout << endl << "Созданная очередь: "
100         << endl;
101     cout << q;
102     //вводим имя для выборки
103     cout << endl
104          << "Введите имя для выборки: ";
105     string name;
106     cin >> name;
107     //выводи на экран результат выбора
108     cout << endl
109          << "Результат выбора: "
110          << endl;
111     chooseQueue(q, name);
112     return 0;
113 }
```

Результат работы программы:



Варианты индивидуальных заданий

Часть 1. Классы для создания объектов очереди

1. «Владелец телефона»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); № телефона.
2. «Абитуриент»: фамилия; имя; отчество; пол; национальность; дата рождения (год, месяц число); домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); оценки по экзаменам; проходной балл.
3. «Государство»: название страны; столица; государственный язык; население; площадь территории; денежная единица; государственный строй; глава государства.
4. «Автомобиль»: марка; цвет; серийный номер; регистрационный номер; год выпуска; год техосмотра; цена.
5. «Товар»: наименование; стоимость; срок хранения; сорт; дата выпуска; срок годности.
6. «Кинолента»: название; режиссер (фамилия; имя); год выхода; страна; стоимость; доход; прибыль.

7. «Рейс»: марка автомобиля; номер автомобиля; пункт назначения; грузоподъемность (в тоннах); стоимость единицы груза; общая стоимость груза.
8. «Книга»: название; автор (фамилия; имя); год выхода; издательство; себестоимость; цена; прибыль.
9. «Здание»: адрес; тип здания; количество этажей; количество квартир; срок эксплуатации; срок до капитального ремонта (25 лет - срок эксплуатации).
10. «Школьник»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); школа; класс.
11. «Студент»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); ВУЗ; курс; группа; средний бал; специальность.
12. «Покупатель»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); номер кредитной карточки; банковского счета.
13. «Пациент»: фамилия; имя; отчество; пол; национальность; рост; вес; дата рождения (год, месяц число); номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); номер больницы; отделение; номер медицинской карты; диагноз; группа крови.
14. «Владелец автомобиля»: фамилия; имя; отчество; номер телефона; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира) марка автомобиля; номер автомобиля; номер техпаспорта.
15. «Военнослужащий»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); национальность; дата рождения (год, месяц число); должность; звание.
16. «Рабочий»: фамилия; имя; отчество; домашний адрес (почтовый индекс, страна, область, район, город, улица, дом, квартира); национальность; дата рождения (год, месяц число); № цеха; табельный номер; образование; год поступления на работу.

Часть 2. Номера полей для организации выбора объектов из очереди

Выборка по значению:

1. ***первого*** и ***второго*** полей объектов очереди;

2. ***пятого*** и ***шестого*** полей объектов очереди.
3. ***второго*** и ***шестого*** полей объектов очереди;
4. ***первого*** и ***третьего*** полей объектов очереди;
5. ***четвертого*** и ***шестого*** полей объектов очереди;
6. ***второго*** и ***пятого*** полей объектов очереди;
7. ***третьего*** и ***четвертого*** полей объектов очереди;
8. ***первого*** и ***четвертого*** полей объектов очереди;
9. ***четвертого*** и ***пятого*** полей объектов очереди;
10. ***первого*** и ***пятого*** полей объектов очереди;
11. ***третьего*** и ***шестого*** полей объектов очереди;
12. ***первого*** и ***шестого*** полей объектов очереди;
13. ***второго*** и ***третьего*** полей объектов очереди;
14. ***третьего*** и ***пятого*** полей объектов очереди;
15. ***второго*** и ***четвертого*** полей объектов очереди;

Хранение наследников абстрактного класса в контейнере deque

Обобщенная формулировка задания. В соответствии с индивидуальным заданием необходимо создать программу для создания очереди `queue` из объектов заданного класса. Указанные в задании действия должны быть оформлены в виде отдельных функций.

Перегрузить:

- для выбранного класса операцию преобразования (операция « `()` ») объекта класса к строке типа `string`;
- `cout` для очереди `queue` с указанным параметром `<ИмяКласса>`.

Написать следующие функции:

4. функцию формирования очереди `queue` заданной с клавиатуры длины (значения полей объекта также вводятся с клавиатуры);
5. функцию просмотра содержимого очереди `queue` из объектов заданного класса;
6. функцию поиска и вывода на экран объекта очереди с заданным значением ***двух*** полей в соответствии со ***второй*** частью индивидуального задания.

Пример выполнения задания

Формулировка примера задания. Создать абстрактный базовый класс (предметная область) `Base` и производные классы `FirstDerived`, `SecondDerived`. Организовать очередь `deque` из указателей на четыре объекта класса наследника, расположенных в куче, проинициализировать очередь и вывести ее содержимое на экран.

Замечание.

- *поскольку в соответствии с индивидуальным заданием базовый класс будет иметь **три** наследника, то в отличие от примера при выполнении задания, необходимо создать и сохранить в очереди **шесть** объектов производных классов (по два объекта на один класс наследник) в произвольном порядке;*
- *каждый производный класс должен иметь суммарно с предком не менее **шести** свойств;*
- *разделение по числу свойств, принадлежащих базовому классу и наследникам, не имеет значения;*
- *как показано в примере, базовый класс может вовсе не иметь свойств, тогда как производные классы всегда должны иметь свойства.*

Программа.

main.cpp

```
1  #include<iostream>
2  #include<deque>
3  using namespace std;
4
5  //базовый абстрактный класс
6  class Base {
7      public:
8      virtual std::string get() = 0;
9      virtual ~Base() {};
10 };
11
12 //первый производный класс
13 class FirstDerived : public Base {
14     std::string name;
15     int phone;
16     public:
```

```
17      FirstDerived(std::string name, int phone) {
18          this->name = name;
19          this->phone = phone;
20      }
21      std::string get() {
22          return "Клиент: "
23                  + name + std::to_string(phone);
24      }
25  };
26
27  //второй производный класс
28  class SecondDerived : public Base {
29      string color;
30      int id;
31  public:
32      SecondDerived(std::string color, int id) {
33          this->color = color;
34          this->id = id;
35      }
36      std::string get() {
37          return "Цвет машины: "
38                  + color + " "
39                  + std::to_string (id);
40      }
41  };
42
43  //полиморфная (из-за get()) перегрузка вставки
44  //в поток наследников класса Base
45  std::ostream& operator << (std::ostream& stream,
46                                          Base* obj) {
47      //полиморфное обращение к методу get()
48      stream << obj->get();
49      return stream;
50  }
51
52  //инициализация двусторонней очереди указателями НА
53  //наследников
54  void initPtrDeque(deque<Base*>& ptrDeque) {
55      ptrDeque.push_back(new FirstDerived("Ivan",
56                                          291111111));
57      ptrDeque.push_back(new FirstDerived("Serg",
58                                          297777777));
```

```
    ptrDeque.push_back(new SecondDerived("Orange",
                                         12));
    ptrDeque.push_back(new SecondDerived("Black",
                                          9));
}

//перегрузка вставки в поток двусторонней очереди
std::ostream& operator << (std::ostream& stream,
                           deque<Base*>& ptrDeque) {
    for(auto element : ptrDeque) {
        stream << element << endl;
    }
    return stream;
}

void deleteObjAtHeap(deque<Base*>& ptrDeque) {
    for(auto element : ptrDeque) {
        delete element;
    }
}

int main() {
    //необходимо использовать указатель
    //на указатель, т.к. конструктор абстрактного
    //класса вызвать невозможно
    deque<Base*> ptrDeque;
    initPtrDeque(ptrDeque);
    //вывод на экран элементов массива из объектов
    //разных классов
    cout << ptrDeque;
    //удаление всех объектов-наследников из кучи
    deleteObjAtHeap(ptrDeque);
    return 0;
}
```

Результат работы программы совпадают с результатами работы примера из раздела «Абстрактные классы. Динамическая диспетчеризация методов»

Варианты индивидуальных заданий

Каждый производный класс должен иметь суммарно с предком не менее ***шести*** полей строкового и числового типа:

1. создать абстрактный базовый класс (предметная область) «Домашнее животное» и производные классы «Собака», «Кошка», «Попугай»;
2. создать абстрактный базовый класс (предметная область) «Садовое дерево» и производные классы «Яблоня», «Вишня», «Груша»;
3. создать абстрактный базовый класс (предметная область) «Единица хранения в библиотеке» и производные классы «Художественная книга», «Научная книга», «Журнал»;
4. создать абстрактный базовый класс (предметная область) «Автомобиль» и производные классы «Легковой автомобиль», «Грузовой автомобиль», «Вездеход»;
5. создать абстрактный базовый класс (предметная область) «Плавсредство» и производные классы «Корабль», «Баржа», «Яхта»;
6. создать абстрактный базовый класс (предметная область) «Электростанция» и производные классы «Теплоэлектростанция», «Гидроэлектростанция», «Атомная станция»;
7. создать абстрактный базовый класс (предметная область) «Городской транспорт» и производные классы «Троллейбус», «Автобус», «Трамвай»;
8. создать абстрактный базовый класс (предметная область) «Коммунальная техника» и производные классы «Уборочная техника», «Мусоровоз», «Автомобиль-разбрасыватель реагентов»;
9. создать абстрактный базовый класс (предметная область) «Торговая сеть» и производные классы «Гипермаркет», «Специализированные магазин», «Рынок»;
10. создать абстрактный базовый класс (предметная область) «Школа» и производные классы «Учитель», «Школьник», «Повар»;
11. создать абстрактный базовый класс (предметная область) «Рабочий» и производные классы «Слесарь», «Электрик», «Монтажник-высотник»;
12. создать абстрактный базовый класс (предметная область) «Транспортное средство» и производные классы «Автомобиль», «Велосипед», «Повозка»;
13. создать абстрактный базовый класс (предметная область) «Грузоперевозчик» и производные классы «Самолет», «Поезд», «Автомобиль»;
14. создать абстрактный базовый класс (предметная область) «Учащийся» и производные классы «Школьник», «Учащийся ГПТУ» и «Студент»;
15. создать абстрактный базовый класс (предметная область) «Музыкальный инструмент» и производные классы «Ударный», «Струнный», «Духовой»;

16. создать абстрактный базовый класс (предметная область) «Работник фирмы» и производные классы «Менеджер», «Администратор», «Программист».

Обработка контейнера `vector` с помощью шаблона функций

Часть 1. Использование в шаблонах функций операции индексирования и приведения типов

Пример задания и его выполнения

Формулировка примера задания. В одномерном контейнере `vector` из `n` чисел базового типа с помощью ***шаблонов*** функций найти минимальный и ***разделить весь вектор на минимальный***, используя операцию ***индексирования*** в качестве инструмента обращения к элементам контейнера. Инициализацию контейнера выполнить случайными целыми числами.

Программа.

main.cpp

```
#include <iostream>
#include <vector>
#include <cmath>
using namespace std;

//прототип функции
unsigned interactiveN();

//шаблоны функций
template<class T>
void initArray(vector<T>& a) {
    int n = interactiveN();
    srand(time(0));
    for(int i = 0; i < n; i++) {
```

```
15          a.push_back((rand() % 100) * 0.1);
16      }
17  }
18
19  template<class T>
20  T minArray(vector<T>& a) {
21      T min = a[0];
22      for(int i = 1; i < a.size(); i++) {
23          if (min > a[i]) min = a[i];
24      }
25      return min;
26  }
27
28  template<class T>
29  void arrayDivision(vector<T>& a) {
30      T min = minArray(a);
31      //исключение деления На нуль
32      min = min != 0 ? min : 1;
33      for(int i = 0; i < a.size(); i++) {
34          a[i] = a[i] / min;
35      }
36  }
37
38  template<class T>
39  ostream &operator <<(ostream &stream,
40                       vector<T> (&a)) {
41      int n = a.size();
42      for(int i = 0; i < n; i++) {
43          stream << a[i] << " ";
44      }
45      return stream;
46  }
47
48  int main() {
49      vector<float> array;
50      initArray(array);
51      cout << "Initialized array:"
52           << endl << array << endl;
```

```
53      cout << "Minimum of array "
54           << minArray(array) << endl;
55      arrayDivision(array);
56      cout << "Result array:"
57           << array << endl;
58      return 0;
59  }
```

Результат работы программы:

```
Input dimension
5
Initialized array:
5.8 1 5.1 0.3 9.5
Minimum of array 0.3
Result array:19.3333 3.33333 17 1 31.6667
```

Варианты индивидуальных заданий

Используя шаблоны функций инициализировать вектор из `n` чисел случайными числовыми значениями ***какого-либо числового*** базового типа, вывести вектор на консоль до и после заданного преобразования:

1. если целая часть значения элемента вектора по величине четная, то его нацело разделить на два;
2. каждый элемент вектора, целая часть которого дает в остатке двойку при делении на 4 увеличить на единицу;
3. каждый элемент вектора, чей модуль больше `7` обнулить;
4. каждый элемент вектора с четной целой частью и если он отрицателен, возвести в куб.
5. значение каждого элемента вектора с индексом кратным `3`, заменить его дробной частью;
6. каждый элемент вектора, стоящий на четной позиции заменить остатком от деления целой части этого элемента на `6`;
7. каждый элемент с нечетной целой частью вектора заменить остатком от деления целой части этого элемента на `3`;
8. каждый элемент вектора больший `5` умножить на результат деления целой части этого элемента на `5`;
9. каждый элемент вектора, чье значение лежит вне диапазона `[-2; 6]` увеличить на `12`;

10. каждый элемент вектора с индексом кратным `4` умножить на собственный индекс;
11. каждый элемент вектора, чье значение лежит вне диапазона `[-5; 6]` возвести в куб;
12. каждый элемент вектора, чье значение лежит в диапазоне `[-1;10]` умножить на `3`;
13. каждый второй элемент вектора, чье значение лежит в диапазоне `[-3; 5]` заменить `0`;
14. элементы вектора, чей квадрат меньше `16` увеличить втрое.
15. к элементам с нечетными целыми частями значений вектора прибавить значение собственного индекса;
16. элементы вектора целая часть, которых при делении нацело на собственный индекс дают нечетное значение увеличить на `2`.

Часть 2. Использование итератора в шаблонах функций

Замечание.
К сожалению, использовать параметризованный итератор ***непосредственно в заголовках шаблонов*** *функций* ***не позволяет*** *онлайн IDE OnlineGDB. Однако внутри* ***шаблонов*** *функций работа с итераторами ничем не отличается от* ***не****параметризованного случая.*

Пример задания и его выполнения

Формулировка примера задания. В одномерном контейнера `vector` из `n` чисел базового типа с помощью ***шаблонов*** функций найти минимальный и ***разделить весь вектор на минимальный***, используя ***итератор*** в качестве инструмента обращения к элементам контейнера. Инициализацию значений контейнера выполнить с помощью ввода чисел с клавиатуры.

Программа.

main.cpp

```cpp
#include <iostream>
#include <vector>
#include <cmath>
using namespace std;

//прототип функции
unsigned interactiveN();
```

```
 8
 9  //шаблоны функций
10  template<class T>
11  void initArray(vector<T>& a) {
12      int i = 0;
13      for(auto& element : a) {
14          cout << "a["<< i <<"] = ";
15          cin >> element;
16          i++;
17      }
18  }
19
20  template<class T>
21  T minArray(vector<T>& a) {
22      T min = *a.begin();
23      for(auto itr = a.begin() + 1; itr < a.end(); itr++) {
24          if (min > *itr) min = *itr;
25      }
26      return min;
27  }
28
29  template<class T>
30  void arrayDivision(vector<T>& a) {
31      T min = minArray(a);
32      //исключение деления На нуль
33      min = min != 0 ? min : 1;
34      for(auto& element : a) {
35          element = element / min;
36      }
37  }
38
39  template<class T>
40  ostream &operator <<(ostream &stream,
41                       vector<T> (&a)) {
42      int n = a.size();
43      for(int i = 0; i < n; i++) {
44          stream << a[i] << " ";
45      }
46      return stream;
47  }
48
49  int main() {
50      int n = interactiveN();
51      vector<float> array(n);
```

```
52      initArray(array);
53      cout << "Initialized array:"
54           << endl << array << endl;
55      cout << "Minimum of array "
56           << minArray(array) << endl;
57      arrayDivision(array);
58      cout << "Result array:"
59           << array << endl;
60      return 0;
61  }
62
63  unsigned interactiveN() {
64      int n;
65      cout << "Input dimension" << endl;
66      //ввод размерности вектора
67      cin >> n;
68      if (0 < n) return n;
69      exit(100);
70  }
```

Результат работы программы

```
Input dimension
7
a[0] = 6
a[1] = 9
a[2] = -10
a[3] = 2
a[4] = 3
a[5] = 45
a[6] = 61
Initialized array:
6 9 -10 2 3 45 61
Minimum of array -10
Result array:-0.6 -0.9 1 -0.2 -0.3 -4.5 -6.1
```

Варианты индивидуальных заданий

Ввести с консоли значения одномерного вектора `a` из `n` элементов ***любого базового числового*** типа. Используя шаблоны функций решить поставленную задачу:

1. найти произведение первых двух положительных элементов (предполагается, что они всегда существуют); произведением заменить все элементы с нечетной целой частью;
2. найти наименьший элемент среди тех, которые находятся на четных позициях; на полученное значение уменьшить элементы с четными индексами;
3. найти среднее арифметическое `sr` максимума и минимума вектора; далее возвести в куб все элементы меньшие чем `sr`;
4. найти произведение его элементов, принадлежащих интервалу `[min/2, max/2]`; значением этого произведения заменить второй и последний элементы вектора;
5. определить упорядочены ли элементы вектора по возрастанию; если не упорядочены, то поменять в векторе второй и последний элементы вектора, иначе эти элементы возвести в куб;
6. найти индекс первого положительного элемента; все отрицательные элементы, следующие за первым положительным увеличить на модуль суммы отрицательных всего вектора;
7. определить в векторе количество элементов с нечетной целой частью; далее максимальный элемент вектора умножить на найденное количество;
8. определить максимальный или минимальный элемент в векторе встречается раньше; если максимальный, то заменить минимумом второй элемент вектора, иначе заменить максимумом предпоследний элемент вектора;
9. определить упорядочены ли элементы вектора по возрастанию; если ***не*** упорядочены, то определить индекс первого элемента, нарушающего порядок, сам этот элемент увеличить на `2`, иначе поменять местами максимум и минимум в векторе;
10. найти индекс `firstEvenIndex` – первого четного значения элемента в векторе. Преобразовать последние `firstEvenIndex` элементов вектора путем их умножения на значение первого элемента с нечетной целой частью;
11. вычислить разность между суммой элементов, имеющих четные индексы, и суммой элементов, имеющих нечетные индексы; на полученную разность увеличить первую половину вектора;
12. определить количество смен знака (`numSignChange`) для его элементов; Если `numSignChange > 0` , то все элементы после `a[numSignChange]` заменить значением куба разности элемента с нулевым индексом и минимальным элементами вектора;
13. найти максимум среди элементов первой половины вектора и минимум среди второй половины вектора, после чего найденные элементы переставить;

14. выяснить, какое число в векторе встретится ранее – положительное или отрицательное (нули не рассматривать); если положительное – найти в векторе максимальный элемент и возвести его в куб, если отрицательное – возвести в квадрат минимальный элемент;
15. найти минимум среди элементов первой половины вектора и максимум среди второй половины; вычислить сумму найденных значений и заменить им элемент, стоящий перед найденным минимальным элементом (предполагается, что минимальный элемент никогда не будет иметь нулевой индекс);
16. вычислить наибольшее и наименьшее значения разности между соседними элементами; найденными значениями заменить, соответственно, второй и последний элементы вектора.

Применение параметризованных функторов. Шаблоны функций для контейнера `valarray`

Пример задания и его выполнения

Формулировка примера задания. В одномерном контейнере `valarray` из `n` чисел базового типа с помощью методов контейнера, ***шаблонов*** функций и ***функторов*** найти минимальный и ***разделить весь вектор на минимальный***, используя операцию ***индексирования*** в качестве инструмента обращения к элементам контейнера.

Программа.

main.cpp

```
1  #include <iostream>
2  #include <valarray>
3  #include <cmath>
4  using namespace std;
5
6  //прототип функции
7  unsigned interactiveN();
8
9  //шаблоны функций
```

```
10  template<class T>
11  void initArray(valarray<T>& a) {
12      srand(time(0));
13      for(int i = 0; i < a.size(); i++) {
14          a[i] = (rand() % 100) * 0.1;
15      }
16  }
17
18  //параметризованный функциональный тип
19  template<class T>
20  class UserFunctional {
21  public:
22      void operator() (valarray<T>& a) {
23          T min = a.min();
24          //исключение деления На нуль
25          min = min != 0 ? min : 1;
26          for(int i = 0; i < a.size(); i++) {
27              a[i] = a[i] / min;
28          }
29      }
30  };
31
32  template<class T>
33  ostream &operator <<(ostream &stream,
34                       valarray<T> (&a)) {
35      int n = a.size();
36      for(int i = 0; i < n; i++) {
37          stream << a[i] << " ";
38      }
39      return stream;
40  }
41
42  int main() {
43      int n = interactiveN();
44      valarray<float> array(n);
45      initArray(array);
46      cout << "Initialized array:"
47           << endl << array << endl;
48      cout << "Minimum of array "
49           << array.min() << endl;
```

```
50      //создание функтора (функционального
51      //объекта)
52      UserFunctional<float> f;
53      //применение функтора
54      f(array);
55      cout << "Result array:" << array
56           << endl;
57      return 0;
58  }
59
60  unsigned interactiveN() {
61      int n;
62      cout << "Input dimension" << endl;
63      //ввод размерности вектора
64      cin >> n;
65      if (0 < n) return n;
66      exit(100);
67  }
```

Результат работы программы полностью совпадает с результатом примера программы в теме «Обработка контейнера `vector` с помощью шаблона функций».

Варианты индивидуальных заданий

Часть 1

Используя шаблоны функций инициализировать вектор из `n` чисел случайными числовыми значениями ***какого-либо числового*** базового типа, вывести вектор на консоль до и после заданного преобразования:

1. каждый элемент вектора, стоящий на четной позиции заменить остатком от деления целой части этого элемента на `6`;
2. каждый элемент с нечетной целой частью вектора заменить остатком от деления целой части этого элемента на `3`;
3. каждый элемент вектора больший `5` умножить на результат деления целой части этого элемента на `5`;
4. каждый элемент вектора, чье значение лежит вне диапазона `[-2; 6]` увеличить на `12`;

5. каждый элемент вектора с индексом кратным `4` умножить на собственный индекс;
6. каждый элемент вектора, чье значение лежит вне диапазона `[-5; 6]` возвести в куб;
7. каждый элемент вектора, чье значение лежит в диапазоне `[-1;10]` умножить на `3`;
8. каждый второй элемент вектора, чье значение лежит в диапазоне `[-3; 5]` заменить `0`;
9. элементы вектора, чей квадрат меньше `16` увеличить втрое.
10. к элементам с нечетными целыми частями значений вектора прибавить значение собственного индекса;
11. элементы вектора целая часть, которых при делении нацело на собственный индекс дают нечетное значение увеличить на `2`;
12. если целая часть значения элемента вектора по величине четная, то его нацело разделить на два;
13. каждый элемент вектора, целая часть которого дает в остатке двойку при делении на 4 увеличить на единицу;
14. каждый элемент вектора, чей модуль больше `7` обнулить;
15. каждый элемент вектора с четной целой частью и если он отрицателен, возвести в куб.
16. значение каждого элемента вектора с индексом кратным `3`, заменить его дробной частью.

Часть 2

Ввести с консоли значения одномерного вектора `a` из `n` элементов ***любого базового числового*** типа. Используя шаблоны функций решить поставленную задачу:

1. найти произведение его элементов, принадлежащих интервалу `[min/2, max/2]`; значением этого произведения заменить второй и последний элементы вектора;
2. определить упорядочены ли элементы вектора по возрастанию; если не упорядочены, то поменять в векторе второй и последний элементы вектора, иначе эти элементы возвести в куб;
3. найти индекс первого положительного элемента; все отрицательные элементы, следующие за первым положительным увеличить на модуль суммы отрицательных всего вектора;
4. определить в векторе количество элементов с нечетной целой частью; далее максимальный элемент вектора умножить на найденное количество;
5. определить максимальный или минимальный элемент в векторе встречается раньше; если максимальный, то заменить минимумом

второй элемент вектора, иначе заменить максимумом предпоследний элемент вектора;

6. определить упорядочены ли элементы вектора по возрастанию; если ***не*** упорядочены, то определить индекс первого элемента, нарушающего порядок, сам этот элемент увеличить на `2`, иначе поменять местами максимум и минимум в векторе;
7. найти индекс `firstEvenIndex` – первого четного значения элемента в векторе. Преобразовать последние `firstEvenIndex` элементов вектора путем их умножения на значение первого элемента с нечетной целой частью;
8. вычислить разность между суммой элементов, имеющих четные индексы, и суммой элементов, имеющих нечетные индексы; на полученную разность увеличить первую половину вектора;
9. определить количество смен знака (`numSignChange`) для его элементов; Если `numSignChange > 0` , то все элементы после `a[numSignChange]` заменить значением куба разности элемента с нулевым индексом и минимальным элементами вектора;
10. найти максимум среди элементов первой половины вектора и минимум среди второй половины вектора, после чего найденные элементы переставить;
11. выяснить, какое число в векторе встретится ранее – положительное или отрицательное (нули не рассматривать); если положительное – найти в векторе максимальный элемент и возвести его в куб, если отрицательное – возвести в квадрат минимальный элемент;
12. найти минимум среди элементов первой половины вектора и максимум среди второй половины; вычислить сумму найденных значений и заменить им элемент, стоящий перед найденным минимальным элементом (предполагается, что минимальный элемент никогда не будет иметь нулевой индекс);
13. вычислить наибольшее и наименьшее значения разности между соседними элементами; найденными значениями заменить, соответственно, второй и последний элементы вектора;
14. найти произведение первых двух положительных элементов (предполагается, что они всегда существуют); произведением заменить все элементы с нечетной целой частью;
15. найти наименьший элемент среди тех, которые находятся на четных позициях; на полученное значение уменьшить элементы с четными индексами;
16. найти среднее арифметическое `sr` максимума и минимума вектора; далее возвести в куб все элементы меньшие чем `sr`.

ЛИТЕРАТУРА

1. Кравчук, А.С. Язык С++. Императивное программирование: учебные материалы для студентов специальности: 1-31 03 08 «Математика и информационные технологии (по направлениям)» / А.С. Кравчук, А.И. Кравчук, Е.В. Кремень; БГУ, Механико-математический фак., Каф. веб-технологий и компьютерного моделирования. – Минск: БГУ, 2023. – 389 с. URL: https://elib.bsu.by/handle/123456789/296928
2. Кравчук, А.С. Язык С++. Объектно-ориентированное программирование. Библиотека STL: учебные материалы для студентов специальности: 6-05-0533-07 «Математика и компьютерные науки (по профилизациям)» / А.С. Кравчук, А.И. Кравчук, Е.В. Кремень; БГУ, Механико-математический фак., Каф. веб-технологий и компьютерного моделирования. – Минск : БГУ, 2023. – 291 с. URL: https://elib.bsu.by/handle/123456789/300184
3. Кравчук, А.И. Сборник лабораторных работ и примеров решения задач по алгоритмизации и программированию на языке Си / А.И. Кравчук, А.С. Кравчук – Минск: Технопринт, 2002. – 116 с.
4. Демидович, Е.М. Основы алгоритмизации и программирования. Язык Си / Е.М. Демидович – Санкт-Петербург: BHV, 2008. – 440 с.
5. Монастырный, П.И. Системы нелинейных численных уравнений / П.И. Монастырный, М.В. Игнатенко, Н.П.Феденко и др. – Мн.: БГУ, 2003. – 30 с.
6. Бронштейн, И.Н. Справочник по математике для инженеров и учащихся втузов / И.Н. Бронштейн, К.А. Семендяев – М.: Наука, 1986. – 544 с.
7. Ахмадиев, Ф.Г. Численные методы. Примеры и задачи / Ф.Г. Ахмадиев, Ф.Г. Габбасов, Л.Б. Ермолаева, И.В. Маланичев. – Казань: КГАСУ, 2017. – 107 с.
8. Аленский, Н.А. Практическое руководство по языку С++ / Н.А. Аленский, ГУО «Акад. Последипл. Образования». – Минск: АПО, 2007. – 276 с.