Белорусский государственный университет

УТВЕРЖДАЮ
Проректор по учебной работе
и образовательным инновациям
_____О.Г. Прохоренко
«20» октября_____ 2023 г.
Регистрационный № УД – 463/б.

Алгоритмы и структуры данных

Учебная программа учреждения высшего образования по учебной дисциплине для специальности:

6-05-0533-09 Прикладная математика
6-05-0533-11 Прикладная информатика

2023 г.

Учебная программа составлена на основе ОСВО 6-05-0533-09-2023 специальности 6-05-0533-09 «Прикладная математика», примерного учебного плана, регистрационный № 6-05-05-020/пр. от 20.12.2022, учебных планов БГУ: № 6-5.3-57/01 от 15.05 2023 г., № 6-5.3-57/02 от 15.05 2023 г., № 6-5.3-57/03 от 15.05 2023 г., № 6-5.3-57/04 от 15.05 2023 г.;

на основе ОСВО 6-05-0533-11-2023 специальности 6-05-0533-11 «Прикладная информатика», примерного учебного плана, регистрационный № 6-05-05-029/пр. от 30.01.2023, учебных планов БГУ: №6-5.3-59/03 от 15.05. 2023 г., №6-5.3-59/04 от 15.05. 2023 г., №6-5.3-59/05 от 15.05. 2023 г., №6-5.3-59/01ин. от 31.05. 2023 г., №6-5.3-59/02ин. от 31.05. 2023 г., №6-5.3-59/03ин. от 31.05. 2023 г.

СОСТАВИТЕЛИ:

Е.П. Соболевская – доцент кафедры дискретной математики и алгоритмики факультета прикладной математики и информатики Белорусского государственного университета, кандидат физико-математических наук, доцент;

В.М. Котов – заведующий кафедрой дискретной математики и алгоритмики факультета прикладной математики и информатики Белорусского государственного университета, доктор физико-математических наук, профессор;

РЕЦЕНЗЕНТЫ:

П.В. Гляков – профессор кафедры информационных технологий в культуре Белорусского государственного университета культуры и искусства, кандидат физико-математических наук, профессор;

Н.В. Лапицкая – заведующая кафедрой программного обеспечения информационных систем Учреждения образования «Белорусский государственный университет информатики и радиоэлектроники», кандидат технических наук, доцент.

РЕКОМЕНДОВАНА К УТВЕРЖДЕНИЮ:

Кафедрой дискретной математики и алгоритмики
(протокол № 3 от 19.10.2023);

Научно-методическим советом БГУ

(протокол № 2 от 19.10.2023)

Заведующий кафедрой _____ В.М. Котов

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Цели и задачи учебной дисциплины

Цель учебной дисциплины «Алгоритмы и структуры данных» – формирование навыков для построения и анализа методов и алгоритмов при решении модельных задач дискретной оптимизации и их применение на практике. При изложении материала учебной дисциплины целесообразно выделить этап построения математической модели, существенно влияющей на ее адекватность реальной проблеме, а также показать возможность использования аппарата теории алгоритмов для анализа и обоснования выбора наиболее эффективных методов и алгоритмов для решения прикладных задач.

Задачи учебной дисциплины

1. Сформировать такие фундаментальные понятий, как размерность задачи и трудоёмкость алгоритма.
2. Изучить подходы для определения трудоёмкости алгоритма посредством составления и решения рекуррентных уравнений.
3. Изучить современные структуры данных, уметь обосновывать выбор соответствующей структуры в зависимости от набора базовых операций, используемых в алгоритме.
4. Уметь строить графовые модели и применять алгоритмы на графах для решения прикладных задач

Место учебной дисциплины

В системе подготовки специалиста с высшим образованием для специальности 6-05-0533-09 «Прикладная математика» учебная дисциплина относится **к модулю** «Дискретная математика и алгоритмика» компонента учреждения образования.

В системе подготовки специалиста с высшим образованием для специальности 6-05-0533-11 «Прикладная информатика» учебная дисциплина относится **к модулю** «Дискретная математика и алгоритмы» государственного компонента.

Учебная программа составлена с учетом межпредметных **связей** и программ по дисциплинам.

Для специальности 6-05-0533-09 «Прикладная математика» основой для изучения учебной дисциплины является дисциплина государственного компонента «Основы и методологии программирования» модуля «Программирование», дисциплина компонента учреждения образования «Дискретная математика и математическая логика» модуля «Дискретная математика и алгоритмика». Знания, полученные в учебной дисциплине, используются при изучении дисциплины компонента учреждения образования «Исследование операций» модуля «Математические методы принятия решений», дисциплины по выбору компонента учреждения образования «Анализ и обработка больших данных».

Для специальности 6-05-0533-11 «Прикладная информатика» основой для изучения учебной дисциплины являются дисциплины государственного компонента «Дискретная математика и математическая логика», «Теория графов» модуля «Дискретная математика и алгоритмы», дисциплина «Основы и методологии программирования» модуля «Программирование». Знания, полученные в учебной дисциплине, используются при изучении дисциплины компонента учреждения образования «Исследование операций» модуля «Интеллектуальные системы», дисциплин по выбору компонента учреждения образования «Методы и алгоритмы обработки данных», «Анализ и обработка больших данных».

Требования к компетенциям

Освоение учебной дисциплины «Алгоритмы и структуры данных» должно обеспечить формирование следующей ***специализированной*** компетенции *для специальности 6-05-0533-09 «Прикладная математика»*:

СК-2. Реализовывать современные структуры данных, строить графовые модели и применять алгоритмы на графах для решения прикладных задач, обосновывать корректность алгоритма и оценивать его асимптотическую сложность.

Освоение учебной дисциплины «Алгоритмы и структуры данных» должно обеспечить формирование следующей ***базовой профессиональной*** компетенции *для специальности 6-05-0533-11 «Прикладная информатика»*:

БПК-3. Характеризовать предмет и объекты дискретной математики и математической логики, использовать основные приемы разработки эффективных алгоритмов и знания об основных структурах данных при решении прикладных задач.

В результате освоения учебной дисциплины студент должен:

знать:

- понятие размерности задачи и трудоемкости алгоритма,
- основные способы решения рекуррентных уравнений,
- основные подходы при разработке эффективных алгоритмов,
- способы организации структур данных и технологию их использования,
- виды поисковых деревьев,
- базовые алгоритмы на графах.

уметь:

- сводить решение исходной задачи к решению подзадач и определять трудоемкость алгоритмов на основе рекуррентных соотношений,
- выбирать подходящие структуры данных при разработке эффективного алгоритма решения задачи,
- реализовывать поисковые деревья,
- строить графовые модели и применять базовые базовые алгоритмы на графах.

владеть:

– основными подходами для разработки эффективных алгоритмов: метод «разделяй и властвуй» и динамическое программирование;
– навыками реализации и использования современных структур данных.

Структура учебной дисциплины

Дисциплина изучается в 3-м семестре. Всего на изучение учебной дисциплины «Алгоритмы и структуры данных» отведено:

для специальности 6-05-0533-09 «Прикладная математика»:

– в очной форме получения высшего образования: 100 часов, в том числе 68 аудиторных часов, из них: лекции – 34 часа, лабораторные занятия – 34 часа часов.

Трудоемкость учебной дисциплины составляет 3 зачетные единицы.

Форма промежуточной аттестации – зачет.

для специальности 6-05-0533-11 «Прикладная информатика»:

– в очной форме получения высшего образования: 108 часов, в том числе 68 аудиторных часов, из них: лекции – 34 часа, лабораторные занятия – 34 часа.

Трудоемкость учебной дисциплины составляет 3 зачетные единицы.

Форма промежуточной аттестации – зачет.

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

Раздел 1. Анализ алгоритмов

Тема 1.1 Введение. Основные понятия и определения

Предмет теории алгоритмов. Историческое развитие теории алгоритмов и ее место среди других математических наук и в естествознании. Формальное описание задачи. Размерность задачи. Асимптотики O, Ω, Θ. Трудоёмкость алгоритма. Полиномиальные и экспоненциальные алгоритмы. Примеры алгоритмов решения задач и оценка их трудоёмкости.

Тема 1.2. Рекуррентные уравнения и основные методы их решения

Понятие рекуррентного уравнения. Полное рекуррентное уравнение. Основные методы решения рекуррентных уравнений.

Оценка трудоемкости базовых алгоритмов внутренней сортировки и поиска, используя рекуррентные уравнения. Особенности реализации алгоритмов внутренней сортировки в высокоуровневых языках программирования.

Раздел 2. Разработка эффективных алгоритмов

Тема 2.1. Метод «разделяй и властвуй»

Основные подходы к разработке эффективных алгоритмов: метод «разделяй и властвуй». Основные этапы метода. Примеры решения задач.

Тема 2.2. Динамическое программирование

Основные подходы к разработке эффективных алгоритмов: динамическое программирование. Основные этапы метода. Подходы к реализации динамического программирования. Демонстрация метода на примерах решения прикладных задач.

Раздел 3. Структуры данных и абстрактные типы данных

Тема 3.1. Простейшие структуры данных и абстрактные типы данных

Понятие абстрактного типа данных и структуры данных и их концептуальное отличие. Способы организации простейших структур данных: массив (англ. *array*), динамический массив (англ. *dynamic array*), связный список (англ. *linked list*). Способы организации динамического массива на базе статического, реаллокации (англ. *reallocation*). Реализация базовых операций и их трудоемкость. Сравнение связных списков и динамических массивов. Применение на практике.

Особенности реализации интерфейса простейших абстрактных типов данных: список (англ. *array*), стек (англ. *stack*), очередь (англ. *queue*), двухсторонняя очередь (англ. *deque*), множество (англ. *set*), ассоциативный массив (англ. *associative array*) в высокоуровневых языках программирования.

Тема 3.2. Специальная древовидная структура данных *куча*

Абстрактный тип данных приоритетная очередь (англ. *priority queue*). Подходы к реализации интерфейса приоритетной очереди. Реализация интерфейса приоритетной очереди в высокоуровневых языках программирования.

Специализированная древовидная структура данных *куча* (англ. *heap*). Способы реализации структуры данных *куча* с помощью корневых деревьев: бинарная куча, биномиальная куча, куча Фибоначчи. Представление структуры данных *куча* в памяти, реализация базового и расширенного набора операций и их трудоемкость. Амортизированная (усреднённая) оценка трудоёмкости операции. Использование структуры *куча* для разработки эффективных алгоритмов решения прикладных задач на примере реализации метода сжатия информации Хаффмана и алгоритма пирамидальной сортировки элементов массива.

Тема 3.3. Специальная структура данных *система непересекающихся множеств*

Система непересекающихся множеств – СНМ (англ. *disjoint set union*). Способы реализации интерфейса СНМ: на массиве, на связном списке с указателем на представителя, с помощью корневых деревьев с эвристиками объединения по размеру и сжатия пути. Реализация базовых операций и их трудоемкость. Использование структуры данных СНМ для разработки эффективных алгоритмов решения прикладных задач.

Тема 3.4. Структуры данных для решения задач на интервалах

Динамическая и статическая версия задач. Онлайн (англ. *online*) и офлайн (англ. *offline*) версия. Постановка задачи о сумме на интервале – RSQ (англ. *range sum query*) и задачи поиска минимального значения на интервале – RMQ (англ. *range minimum query*).

Специальные структуры данных для решения задач на интервалах: подсчёт префиксных сумм, корневая эвристика (*sqrt*-декомпозиция). Реализация базовых операций и их трудоёмкость.

Использование структур данных для решения задач на интервалах для разработки эффективных алгоритмов решения прикладных задач.

Тема 3.5. Структуры данных для организации поиска элемента

Методы хранения деревьев в памяти компьютера Бинарные поисковые деревья. Реализация базовых операций и их трудоемкость.

Сбалансированные поисковые деревья: АВЛ-деревья, 2–3-деревья. Поддержка инвариантов сбалансированности. Реализация базовых операций и их трудоёмкость.

Прямая адресация. Хеш-таблицы и хеш-функции. Коллизии. Методы разрешения коллизий: метод цепочек, открытая адресация. Универсальное семейство хеш-функций. Совершенное хеширование. Хеш-таблицы на практике.

Раздел 4. Алгоритмы на графах

Тема 4.1. Способы обхода вершин графа

Графовые модели. Методы хранения графов в памяти компьютера.

Алгоритмы поиска в ширину (BFS) и в глубину (DFS) и их трудоёмкость. Алгоритмы определения связности и двудольности графа, выделения сильно связных компонент ориентированного графа их трудоёмкость.

Алгоритмы топологической сортировки вершин ориентированного графа и их трудоёмкость.

Алгоритмы построения эйлерова цикла графа и их трудоёмкость.

Тема 4.2. Кратчайший маршрут

Постановка задачи построения кратчайшего маршрута и подходы к её решению. Произвольные веса: алгоритмы Форда-Беллмана, Флойда и их трудоёмкость. Поиск контура отрицательного веса. Неотрицательные веса: алгоритм Дейкстры, подходы к программной реализации алгоритма и его трудоёмкость. Сведение к неотрицательным весам – метод потенциалов (преобразование Джонсона). Алгоритмы построения кратчайших маршрутов для специальных классов графов.

Тема 4.3. Минимальное остовное дерево

Минимальное остовное дерево графа – MST (англ. *minimum spanning tree*). Алгоритмы Прима и Крускала построения MST и их трудоёмкость.

Тема 4.4. Максимальный поток в сети и его приложения

Постановка классической задачи нахождения максимального потока в двухполюсной сети с ограничениями. Максимальный поток и минимальный разрез сети. Метод Форда-Фалкерсона. Алгоритм Эдмондса–Карпа.

Алгоритмы построения максимального потока минимальной стоимости и их трудоемкость: метод устранения отрицательных циклов, метод минимальных путей.

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА УЧЕБНОЙ ДИСЦИПЛИНЫ

Очная форма получения высшего образования с применением дистанционных образовательных технологий (ДОТ)

Номер раздела, темы	Название раздела, темы	Количество аудиторных часов					Количество часов УСР	Форма контроля знаний
		Лекции	Практические занятия	Семинарские занятия	Лабораторные занятия	Иное		
1	2	3	4	5	6	7	8	9
1	**Анализ алгоритмов**	**4**			**4**			
1.1	Введение. Основные понятия и определения	2			2			Собеседование. Электронный тест: асимптотики (iRunner)
1.2	Рекуррентные уравнения и основные методы их решения	2			2			Электронный тест: решение рекуррентных уравнений (iRunner)
2	**Разработка эффективных алгоритмов**	**6**			**6**			
2.1	Метод «разделяй и властвуй»	2			2			Отчеты по домашним упражнениям с их устной защитой
2.2	Динамическое программирование	4			4			Контрольная работа №1.
3	**Структуры данных и абстрактные типы данных**	**12**			**12**			

3.1	Простейшие структуры данных и абстрактные типы данных	2			2			Отчеты по домашним упражнениям с их устной защитой. Электронный тест: массив, связный список (iRunner)
3.2	Специальная древовидная структура данных *куча*	2			2			Контрольная работа №2. Электронный тест: кучи (iRunner)
3.3	Специальная структура данных *система непересекающихся множеств*	2			2			Отчеты по домашним упражнениям с их устной защитой. Электронный тест: DSU (iRunner)
3.4	Структуры данных для решения задач на интервалах	2			2			Собеседование.
3.5	Структуры данных для организации поиска элемента	4			4			Дискуссия. Контрольная работа №3. Электронный тест: поисковые деревья (iRunner)
4	**Алгоритмы на графах**	**12**			**12**			
4.1	Способы обхода вершин графа	4			4			Отчеты по домашним упражнениям с их устной защитой. Электронный тест: графы (iRunner)
4.2	Кратчайший маршрут	2			2			Коллоквиум по разделам 1–3.
4.3	Минимальное остовное дерево	2			2			Отчеты по домашним упражнениям с их устной

								защитой.
4.4	Максимальный поток в сети и его приложения	4			4			Электронный итоговый тест: по разделам 1–4 (iRunner). Дискуссия.

ИНФОРМАЦИОННО-МЕТОДИЧЕСКАЯ ЧАСТЬ

Перечень основной литературы

1. Котов, В. М. Сборник задач по теории алгоритмов. Организация перебора и приближенные алгоритмы [Электронный ресурс] : электронный учебно-методический комплекс для специальности: 1-31 03 04 «Информатика» / В. М. Котов, Е. П. Соболевская, Г. П. Волчкова ; БГУ, Фак. прикладной математики и информатики, Каф. дискретной математики и алгоритмики. – Минск : БГУ, 2021. – [Электронный ресурс]. – Режим доступа: – URL: https://elib.bsu.by/handle/123456789/272717.

2. Котов, В. М. Теория алгоритмов. Организация перебора и приближенные алгоритмы : учеб.-метод. пособие / В. М. Котов, Е. П. Соболевская, Г. П. Волчкова. – Минск: БГУ, 2022. – 151 с.

3. Сборник задач по теории алгоритмов : учеб.-метод. пособие / В. М. Котов [и др.] – Минск : БГУ, 2017. – 183с. – URL: https://elib.bsu.by/handle/123456789/181529.

4. Сборник задач по теории алгоритмов. Структуры данных: учеб.-метод. пособие / С. А. Соболь [и др.] – Минск : БГУ, 2020. – 159 с. – URL: https://elib.bsu.by/handle/123456789/255033.

5. Тюкачев, Н. А. C#. Алгоритмы и структуры данных / Н. А. Тюкачев, В. Г. Хлебостроев. – 6-е изд., стер. – Санкт-Петербург : Лань, 2023. – 232 с. – Текст : электронный // Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/346067.

Перечень дополнительной литературы

6. Алгоритмы: построение и анализ / Т. Кормен [и др.] – М.: Вильямс, 2005. – 1296 с.

7. Котов, В. М. Алгоритмы и структуры данных: учеб. пособие / В. М. Котов, Е. П. Соболевская, А. А. Толстиков. – Минск: БГУ, 2011. – 267 с. – (Классическое университетское издание). – URL: https://elib.bsu.by/handle/123456789/8522.

8. Опыт использования образовательной платформы Insight Runner на факультете прикладной математики и информатики Белорусского государственного университета. Роль университетского образования и науки в современном обществе: материалы междунар. науч. конф., Минск, 26–27 февр. 2019 г. / редкол.: А. Д. Король (пред.) [и др.]. – Минск: БГУ, 2019. – – URL: С. 263 – 267. https://elib.bsu.by/handle/123456789/231914.

9. Пападимитриу, Х. Комбинаторная оптимизация. Алгоритмы и сложность / Х. Пападимитриу, К. Стайглиц – М.: Мир, 1985. – 512 с.

10. Рейнтгольд, Э. Комбинаторные алгоритмы. Теория и практика / Э. Рейнтгольд., Ю. Нивергельт, Н. Део – М.: Мир, 1980. – 466 с.

11. Соболь, С. А. Методика преподавания дисциплин по теории алгоритмов с использованием образовательной платформы iRUNNER / С. А. Соболь, В. М. Котов, Е. П. Соболевская // Электронный науч.-методич. журнал «Педагогика информатики». – 2020 – № 2. http://pcs.bsu.by/2020_2/6ru.pdf

12. Теория алгоритмов: учеб. пособие / П. А. Иржавский [и др.] – Минск: БГУ, 2013. – 159 с. – URL: https://elib.bsu.by/handle/123456789/91612.

Электронные ресурсы

13. Образовательный портал БГУ [Электронный ресурс]. – Режим доступа: https://edufpmi.bsu.by/course/view.php?id=118. – Дата доступа: 02.09.2023.

14. Образовательная платформа Insight Runner [Электронный ресурс]. – Режим доступа: https://acm.bsu.by. – Дата доступа: 02.09.2023.

Перечень рекомендуемых средств диагностики и методика формирования итоговой отметки

Объектом диагностики компетенций студентов являются знания, умения, полученные ими в результате изучения учебной дисциплины. Выявление учебных достижений студентов осуществляется с помощью мероприятий текущего контроля и промежуточной аттестации.

Для диагностики компетенций в рамках учебной дисциплины рекомендуется использовать следующие формы:

1. Устная форма: собеседование, дискуссия, коллоквиум.
2. Письменная форма: контрольные работы.
3. Устно-письменная форма: отчеты по домашним упражнениям с их устной защитой.
4. Техническая форма: электронные тесты.

В качестве рекомендуемых технических средств диагностики используется Образовательная платформа Insight Runner (www.acm.bsu.by), а также обучение, организованное на платформе Moodle (https://edufpmi.bsu.by).

Формой промежуточной аттестации по дисциплине «Алгоритмы и структуры учебным планом предусмотрен **зачет.**

При формировании итоговой отметки используется рейтинговая система оценки знаний студента, дающая возможность проследить и оценить динамику процесса достижения целей обучения.

Рейтинговая система предусматривает использование весовых коэффициентов в ходе проведения контрольных мероприятий текущей аттестации.

Примерные весовые коэффициенты, определяющие вклад текущей аттестации в отметку при прохождении промежуточной аттестации:

Формирование отметки за текущую аттестацию:

- отчет по домашним упражнениям с их устной защитой – 60 %;
- выполнение теста – 20%;
- контрольные работы – 10 %;
- коллоквиум – 10 %.

Итоговая отметка по дисциплине рассчитывается на основе отметки текущей аттестации (рейтинговой системы оценки знаний) – 60% и отметки на зачете – 40%.

Примерная тематика лабораторных занятий

Занятие 1. Анализ алгоритмов. Размерность задачи, асимптотики, полиномиальные и экспоненциальные алгоритмы (выполнение теста в системе iRunner).

Занятие 2. Рекуррентные уравнения и основные методы их решения (выполнение теста в системе iRunner).

Занятие 3. Метод разделяй и властвуй (поиск максимального и минимального элементов массива, алгоритмы внутренней сортировки сравнениями: сортировка слиянием, сортировка Ч. Хоара), принцип балансировки. Особенности реализации алгоритмов поиска и внутренней сортировки в высокоуровневых языках программирования.

Занятия 4 – 5. Динамическое программирование ($n!$, числа Фибоначчи, Ханойские башни, игра Баше; решение общих и индивидуальных задач в системе iRunner).

Занятия 6. Простейшие абстрактные типы данных и структуры данных (массив, связный список, способы реализации динамического массива на базе статического, оценки трудоемкости, выполнение теста в системе iRunner).

Занятие 7. Специальная структура данных «куча» (специальный алгоритм построения бинарной кучи за линейное время, использование бинарной кучи при решении задач, решение общих задач, выполнение теста в системе iRunner).

Занятие 8. Специальная структура данных система непересекающихся множеств (использование DSU при решении задач, решение общих задач и в выполнение теста в системе iRunner).

Занятие 9. Специальная структура данных система непересекающихся множеств (способы реализации DSU в памяти компьютера, особенности реализации на семействе корневых деревьев с эвристиками объединения по

размеру и сжатия пути, использование DSU при решении задач, решение общих задач и в выполнение теста в системе iRunner).

Занятие 10 – 11. Структуры данных для организации поиска элемента (сбалансированные поисковые деревья, хеш-таблицы, решение общих задач по бинарным поисковым деревьям, задача проверки бинарного дерева на поисковость, выполнение теста в системе iRunner).

Занятие 12 –13. Способы обхода вершин графа (решение общих задач и в выполнение теста в системе iRunner).

Занятие 14. Алгоритмы построения кратчайших маршрутов (решение общих задач в системе iRunner).

Занятие 15. Алгоритмы построения минимального остовного дерева (решение индивидуальных задач в системе iRunner).

Занятия 16 – 17. Максимальный поток в сети и его приложения (решение общих задач в системе iRunner).

Рекомендуемая тематика контрольных работ и коллоквиума:

1) Контрольная работа № 1 «Динамическое программирование».
2) Контрольная работа № 2 «Специальная структура данных *куча*».
3) Контрольная работа № 3. «Структуры данных для организации поиска элемента».
4) Коллоквиум «Использование современных структур данных при разработке эффективных алгоритмов решения задач».

Описание инновационных подходов и методов к преподаванию учебной дисциплины

При организации образовательного процесса используются следующие методы:

- ***метод учебной дискуссии,*** который предполагает участие студентов в целенаправленном обмене мнениями, идеями для предъявления и/или согласования существующих позиций по определенной проблеме. Использование метода обеспечивает появление нового уровня понимания изучаемой темы, применение знаний (теорий, концепций) при решении проблем, определение способов их решения.

- ***метод группового обучения,*** который представляет собой форму организации учебно-познавательной деятельности обучающихся, предполагающую функционирование разных типов малых групп, работающих как над общими, так и специфическими учебными заданиями.

В качестве технических средств для организации работы в рамках учебной дисциплины рекомендуется использовать Образовательную платформу Insight Runner (www.acm.bsu.by), Образовательный портал БГУ

(https://edufpmi.bsu.by) – инструмент с эффективной функциональностью контроля, тренинга и самостоятельной работы.

Для контроля самостоятельности выполнения работ рекомендуется использовать автоматические системы определения несанкционированных материалов, функционирующие в Insight Runner.

Методические рекомендации по организации самостоятельной работы обучающихся

Для организации самостоятельной работы студентов по учебной дисциплине следует использовать современные информационные ресурсы: разместить на образовательном портале комплекс учебных и учебно-методических материалов (учебно-программные материалы, учебное издание для теоретического изучения дисциплины, методические указания к лабораторным занятиям, материалы текущего контроля и текущей аттестации, позволяющие определить соответствие учебной деятельности обучающихся требованиям образовательного стандарта высшего образования и учебно-программной документации, в т.ч. вопросы для подготовки к зачету, задания, тесты, вопросы для самоконтроля, тематика рефератов и др., список рекомендуемой литературы, информационных ресурсов и др.).

При составлении общих и индивидуальных заданий по учебной дисциплине необходимо предусмотреть возрастание их сложности: от заданий, формирующих достаточные знания по изученному учебному материалу на уровне узнавания, к заданиям, формирующим компетенции на уровне воспроизведения, и далее к заданиям, формирующим компетенции на уровне применения полученных знаний.

Таким образом, задания по учебной дисциплине рекомендуется делить на три модуля:

- задания, формирующие достаточные знания по изученному учебному материалу на уровне узнавания;
- задания, формирующие компетенции на уровне воспроизведения;
- задания, формирующие компетенции на уровне применения полученных знаний.

Примерный перечень заданий

Задания на уровне узнавания

Задание 1. Перечислите известные вам алгоритмы внутренней сортировки сравнениями. Какие из этих алгоритмов реализованы в стандартных библиотеках высокоуровневых языков программирования (C++, Java, Python) и каковы особенности их реализации?

Задание 2. Приведена некоторая древовидная структура, в вершинах которой стоят некоторые целые числа. Может ли данная древовидная структура быть заданием: бинарного дерева, бинарного поискового дерева,

АВЛ-дерева, сбалансированного дерева, идеально-сбалансированного дерева, бинарной кучи, биномиальной кучи?

Задание 3. Задана библиотека базовых алгоритмов на графах. Какие из алгоритмов можно применить, если в графе нужно найти любой маршрут между заданной парой вершин? Какие из алгоритмов можно применить, если в графе нужно найти наименьший по количеству ребер маршрут между заданной парой вершин?

Задания на уровне воспроизведения

Задание 1. *Построить дерево*
Ограничение по времени: 1 с
Ограничение по памяти: 256 МБ

По набору ключей постройте бинарное поисковое дерево и выполните его прямой левый обход.

Формат входных данных

Входной файл содержит последовательность чисел — ключи вершин в порядке добавления в дерево. Ключи задаются в формате по одному в строке.

Формат выходных данных

Выходной файл должен содержать последовательность ключей вершин, полученную прямым левым обходом дерева.

input.txt	*output.txt*
5 2 4 1 8 7	5 2 1 4 8 7

Задание 2. *Удалить из дерева*
Ограничение по времени: 1 с
Ограничение по памяти: 256 МБ

По набору ключей постройте бинарное поисковое дерево. Удалите из него ключ (правым удалением), если он есть в дереве. Выполните прямой левый обход полученного дерева.

Формат входных данных

В первой строке записано целое число — ключ, который нужно удалить из дерева. Вторая строка пустая. Последующие строки содержат последовательность чисел – ключи вершин в порядке добавления в дерево. Ключи задаются в формате по одному в строке. Дерево содержит хотя бы две вершины. Напомним, что в поисковом дереве все ключи по определению уникальны, поэтому при попытке добавить в дерево ключ, который там уже есть, он игнорируется.

Формат выходных данных

Выведите последовательность ключей вершин, полученную прямым левым обходом дерева.

input.txt	*output.txt*
2 4 2 1 3 5	4 3 1 5

Задание 3. *Максимальный поток*

Ограничение по времени: 1 с

Ограничение по памяти: 256 МБ

Задан ориентированный граф G, содержащий n вершин и m дуг, в котором каждой дуге (u, v) приписана пропускная способность $c(u, v)$. Требуется найти в этой сети поток максимальной величины из источника (вершины 1) в сток (вершину n).

Формат входных данных

Первая строка содержит два числа n и m – число вершин и дуг в графе соответственно. Гарантируется, что $1 \le n, m \le 200$.

Каждая из следующих m строк содержит три числа u_i, v_i, w_i – две вершины, которые соединены дугой и пропускная способность этой дуги.

Гарантируется, что $u_i \ne v_i$, $1 \le u_i, v_i \le n$, $1 \le w_i \le 1\,000\,000$ для каждой дуги.

Формат выходных данных

Выведите одно число — максимально возможную величину потока в заданном графе.

input.txt	*output.txt*
4 6 1 2 1 1 3 1 1 4 1 2 3 1 2 4 1 3 4 1	3

Задания на уровне применения полученных знаний

Задание 1. *Путь лягушки*.

Ограничение по времени: 1 с

Ограничение по памяти: 256 МБ

В одном очень длинном и узком пруду по кувшинкам прыгает лягушка. Кувшинки в пруду расположены в один ряд. Лягушка начинает прыгать с первой кувшинки ряда и хочет закончить на последней. Но в силу вредности характера лягушка согласна прыгать только вперед через одну или через две

кувшинки. Например, с кувшинки номер 1 она может прыгнуть лишь на кувшинки номер 3 и номер 4.

На некоторых кувшинках сидят комарики. А именно, на *i*-й кувшинке сидят a_i комаров.

Когда лягушка приземляется на кувшинку, она съедает всех комариков, сидящих на ней. хочет спланировать свой маршрут так, чтобы съесть как можно больше комаров.

Помогите ей: скажите, какие кувшинки она должна посетить на своем пути.

Формат входных данных

Первая строка входного файла содержит число *n* – число кувшинок в пруду $(1 \le n \le 100\,000)$. Вторая строка содержит *n* чисел, которые разделены пробелами: *i*-ое число сообщает, сколько комаров сидит на *i*-ой кувшинке $(1 \le i \le n)$. Все числа целые, неотрицательные и не превосходят $1\,000$.

Формат выходных данных

В первой строке выведите одно число – максимальное число комаров, которые может съесть лягушка.

Во второй строке выведите последовательность чисел – номера тех кувшинок, на которых должна побывать лягушка, в возрастающем порядке. Если решений несколько, выведите любое.

Если лягушка не может добраться до последней кувшинки, то выведите одно число –1.

входной файл	*выходной файл*
6 1 100 3 4 1000 0	5 1 4 6
2 8 9	-1

Задание 2. *Единицы (часть 1)*.

Ограничение по времени: 1 с

Ограничение по памяти: 256 МБ

Дано число *N*. Необходимо определить, сколько есть бинарных строк длины *N*, в которых ровно *K* единиц.

Формат входных данных

В строке входного файла записаны два целых неотрицательных числа *N* и *K* $(0 \le K \le N \le 1000)$.

Формат выходных данных

Выведите одно число – ответ на задачу.

Так как ответ может быть очень большим, необходимо его вывести по модулю $10^9 + 7$.

входной файл	*выходной файл*

3 2	3
4 0	1
5 4	5
6 4	15
7 2	21
8 0	1

Задание 3. *Единицы (часть 2)*.
Ограничение по времени: 1 с
Ограничение по памяти: 256 МБ

Дано число N.

Необходимо определить, сколько есть бинарных строк длины N, в которых ровно K единиц.

Формат входных данных

В строке входного файла записаны два целых неотрицательных числа N и K $\left(0 \le K \le N \le 10^6\right)$.

Формат выходных данных

Выведите одно число – ответ на задачу.

Так как ответ может быть очень большим, необходимо его вывести по модулю $10^9 + 7$.

входной файл	*выходной файл*
3 2	3
4 0	1
5 4	5
6 4	15
7 2	21
8 0	1

Задание 4. *Порядок перемножения матриц*.
Ограничение по времени: 1 с
Ограничение по памяти: 256 МБ

Дана последовательность из s матриц $A_1, A_2, \ldots, A_s$.

Требуется определить, в каком порядке их следует перемножать, чтобы число атомарных операций умножения было минимальным.

Матрицы предполагаются совместимыми по отношению к матричному умножению (т. е. число столбцов матрицы A_{i-1} совпадает с числом строк матрицы A_i).

Будем считать, что произведение матриц – операция, которая принимает на вход две матрицы размера $k \times m$ и $m \times n$ и возвращает матрицу размера $k \times n$, затратив на это $k \cdot m \cdot n$ атомарных операций умножения.

Базовый тип позволяет хранить любой элемент итоговой и любой возможной промежуточной матрицы, поэтому умножение двух элементов требует одной атомарной операции.

Так как перемножение матриц ассоциативно, итоговая матрица не зависит от порядка выполнения операций умножения.

Другими словами, нет разницы, в каком порядке расставляются скобки между множителями, результат будет один и тот же.

Формат входных данных

В первой строке входного задано число матриц s $(2 \le s \le 100)$.

В последующих s строках заданы размеры матриц: $i+1$ содержит через пробел число n_i строк и число m_i столбцов матрицы A_i $(1 \le n_i, m_i \le 100)$. Гарантируется, что m_i совпадает с n_{i+1} для всех индексов от 1 до $s-1$.

Формат выходных данных

Выведите минимальное число атомарных операций умножения, необходимое для перемножения s матриц.

входной файл	*выходной файл*
3 2 3 3 5 5 10	130
4 20 5 5 35 35 4 4 25	3100

Замечание.

В первом примере можно умножать двумя способами:

$(A_1 \cdot (A_2 \cdot A_3))$: требуется $3 \times 5 \times 10 + 2 \times 3 \times 10 = 150 + 60 = 210$ операций;

$((A_1 \cdot A_2) \cdot A_3)$: требуется $2 \times 3 \times 5 + 2 \times 5 \times 10 = 30 + 100 = 130$ операций.

Второй способ эффективнее.

Задание 5. *Наибольшая общая подпоследовательность*.

Ограничение по времени: 2 с

Ограничение по памяти: 64 МБ

Даны две последовательности *A* и *B*, каждая имеет длину *n*.

Найти наибольшее k, для которого существуют две последовательности индексов $0 \le i_1 < i_2 < \ldots < i_k < n$ и $0 \le j_1 < j_2 < \ldots < j_k < n$ такие, что $A_{i_1} = B_{j_1}, A_{i_2} = B_{j_2}, \ldots, A_{i_k} = B_{j_k}$.

Также нужно найти и сами последовательности индексов.

Формат входных данных

В первой строке записано число n $(1 \le n \le 1000)$, длина последовательностей *A* и *B*.

Во второй строке содержится n целых чисел a_i $(1 \le i \le 1000)$ – элементы последовательности *A*.

В третьей строке содержится n целых чисел b_j $(1 \le j \le 1000)$ – элементы последовательности *B*.

Формат выходных данных

В первой строке выведите число k. Во второй строке выведите индексы $i_1, i_2, \ldots, i_k$. В третьей строке выведите индексы $j_1, j_2, \ldots, j_k$. Если подходящий последовательностей индексов несколько, выведите любые из них.

входной файл	*выходной файл*
5 1 2 3 4 5 1 3 2 4 4	3 0 1 3 0 2 4

Задание 6. *Палиндром*.

Ограничение по времени: 1 с

Ограничение по памяти: нет

Вводится непустая строка *S*, которая имеет длину не более 7 000 символов и состоит только из строчных латинских букв.

Необходимо удалить из строки минимальное число символов так, чтобы получился палиндром (строка символов, которая читается слева направо и справа налево одинаково).

Формат входных данных

В первой строке входного файла записана исходная строка *S*.

Формат выходных данных

Выведите в первой строке длину получившегося палиндрома, а во второй строке сам палиндром (если палиндромов несколько, то выведите только один из них).

входной файл	*выходной файл*
asddfsa	6 asddsa

Задание 7. *Строго возрастающая без разрывов последовательность.*

Ограничение по времени: от 1 с до 5 с
Ограничение по памяти: нет

Необходимо из заданной числовой последовательности A, состоящей из n элементов, вычеркнуть минимальное число элементов так, чтобы оставшиеся элементы образовали строго возрастающую подпоследовательность элементов.

Построенный алгоритм должен иметь трудоемкость $O(n \cdot \log n)$.

Формат входных данных

Первая строка входного файла содержит число n $(1 \le n \le 700\,000)$.

Следующая строка содержит n элементов последовательности A, которые разделены пробелами (элементы последовательности – целые числа, не превосходящие по модулю $1\,000\,000\,000$).

Формат выходных данных

Выведите одно число – длину строго возрастающей подпоследовательности элементов.

входной файл	*выходной файл*
6 1 2 3 4 7 6	5

Задание 8. *Преобразование строк.*
Ограничение по времени: 1 с
Ограничение по памяти: нет

На вход подаются две символьные последовательности *A* и *B*, каждая последовательность непустая, состоит из маленьких латинских букв и имеет длину не более 1000 символов.

Необходимо преобразовать последовательность *A* в последовательность *B* с минимальным суммарным штрафом, который определяется следующим образом:

- удаление символа из строки *A* равно *x* баллов;
- вставка символа в строку *A* равна *y* баллов;
- замена символа в строке *A* на любой другой символ равна *z* баллов.

Формат входных данных

В первых трех строках входного файла находятся числа *x*, *y*, *z* соответственно.

Все числа целые положительные и не превосходят 1 000 000.

В следующих двух строках находятся символьные последовательности *A* и *B* (тип элементов последовательности string).

Формат выходных данных

Выведите минимальный суммарный штраф.

входной файл	*выходной файл*
2 3 1 abcd bce	3

Примерный перечень вопросов к зачету

1. Размерность задачи. Время работы алгоритма. Асимптотики. Трудоемкость алгоритмов. Полиномиальные и экспоненциальные алгоритмы. Примеры.

2. Рекуррентные уравнения для базовых алгоритмов поиска элемента в массиве (последовательный поиск, дихотомия). Оценка времени работы алгоритмов.

3. Простейшие алгоритмы внутренней сортировки: включение, выбор, пузырьковая, шейкерная. Особенности реализации. Использование рекуррентных соотношений для оценки трудоемкости алгоритмов внутренней сортировки.

4. Сортировка Ч.Хоара. Правила выбора сепаратора. Доказательство трудоёмкости алгоритма.

5. Сортировка слиянием. Доказательство трудоёмкости алгоритма. Использование дополнительной памяти при программной реализации алгоритма.

6. Реализация алгоритмов сортировки в высокоуровневых языках программирования.

7. Технологии разработки эффективных алгоритмов: принцип «разделяй и властвуй». Основные этапы метода. Демонстрация техники на конкретных примерах и доказательство времени работы алгоритма.

8. Технологии разработки эффективных алгоритмов: «динамическое программирование». Основные этапы метода. Демонстрация техники на конкретных примерах и доказательство времени работы алгоритма. Виды динамического программирования (ДП): ДП вперед, ДП назад, «ленивое» ДП.

9. Абстрактные типы данных (abstract data type) в высокоуровневых языках программирования. Структуры данных. Концептуальное отличие абстрактных типов данных и структур данных.

10. Структура данных массив (array), динамический массив (dynamic array).

11. Различные подходы к организации динамического массива на базе статического. Реаллокации. Усредненные оценки. Реализация массивов в высокоуровневых языках программирования.

12. Структура данных связный список (linked list). Базовые операции и их трудоёмкость. Реализация связных списков в высокоуровневых языках программирования.

13. Сравнение связных списков и динамических массивов. Применение на практике.

14. Абстрактные типы данных список (list), стек (stack), очередь (queue), двухсторонняя очередь (deque). Базовые операции и их трудоёмкость (доказательство). Реализация абстрактных типов в высокоуровневых языках программирования. Примеры использования структур данных при решении практических задач.

15. Абстрактный тип данных приоритетная очередь (priority queue). Реализация в высокоуровневых языках программирования. Примеры использования.

16. Специальная структура данных бинарная куча. Базовые операции и их трудоёмкость (доказательство). Операция модификации ключа. Примеры использования.

17. Специальная структура данных биномиальная куча. Куча Фибоначчи. Базовые операции и их трудоёмкость (доказательство). Усреднённая оценка трудоемкости.

18. Структуры данных для интервальных запросов (наивный подход - подсчёт префиксных сумм, sqrt-декомпозиция. Базовые операции и их трудоёмкость.

19. Специальная структура данных система непересекающихся множеств (DSU). Способы задания в памяти компьютера. Базовые операции и их трудоемкость (доказательства). Эвристика «объединение по размеру» и «сжатия пути». Примеры использования.

20. Деревья. Корневые деревья. Бинарные поисковые деревья. Базовые операции для бинарных поисковых деревьев и доказательство их трудоёмкости. Примеры использования структуры данных для решения практических задач.

21. Сбалансированные деревья. K-сбалансированное корневое дерево. K- идеально сбалансированное корневое дерево. Сбалансированные поисковые деревья: АВЛ-дерево. Поддержка инварианта сбалансированности. Базовые операции и их трудоёмкость (доказательство).

22. Сбалансированные поисковые деревья: 2–3-деревья. Базовые операции и их трудоёмкость (доказательство).

23. Графы. Классы графов. Методы хранения графов в памяти компьютера. Преимущества и недостатки. Примеры зависимости времени работы базовых алгоритмов на графах от способа представления графа в памяти компьютера.

24. Поиск в ширину в графе и его приложения. Доказательство оценки времени работы алгоритма.

25. Поиск в глубину в графе и его приложения. Доказательство оценки времени работы алгоритма.

26. Топологическая сортировка вершин орграфа. Алгоритмы топологической сортировки и доказательство (алгоритм Кана, алгоритм Тарьяна). Оценки времени работы алгоритмов. Демонстрация работы алгоритма на конкретном примере.

27. Алгоритмы определения двудольности графа. Доказательство оценок времени работы алгоритмов. Демонстрация работы алгоритма на конкретном примере.

28. Алгоритм нахождения сильно связных компонент ориентированного графа (алгоритм Касарайю–Шарира). Оценка времени работы алгоритма. Демонстрация работы алгоритма на конкретном примере.

29. Задача поиска кратчайшего маршрута. Подходы к решению задачи. Кратчайший маршрут в графе с неотрицательными весами ребер. Алгоритм Дейкстры. Использование базовых структур данных при реализации алгоритма Дейкстры и доказательство оценки времени работы алгоритма.

30. Эйлеров цикл в графе. Критерий существования в графе эйлерова цикла. Алгоритм построения эйлерова цикла и доказательство оценки времени работы алгоритма.

31. Задача построения минимального остовного дерева связного графа. Алгоритм Краскала. Доказательство оценки времени работы алгоритма. Демонстрация работы алгоритма на конкретном примере.

32. Максимальный поток в сети. Метод Форда-Фалкерсона. Алгоритм Эдмонса-Карпа.

33. Максимальный поток в сети и его приложения.

ПРОТОКОЛ СОГЛАСОВАНИЯ УЧЕБНОЙ ПРОГРАММЫ УВО

Название учебной дисциплины, с которой требуется согласование	Название кафедры	Предложения об изменениях в содержании учебной программы учреждения высшего образования по учебной дисциплине	Решение, принятое кафедрой, разработавшей учебную программу (с указанием даты и номера протокола)
1. Исследование операций	Информационных систем управления	нет	Изменений не требуется (протокол № 3 от 19.10.2023 г.)
2. Анализ и обработка больших данных	Вычислительной математики	нет	Изменений не требуется (протокол № 3 от 19.10.2023 г.)
3. Методы и алгоритмы обработки данных	Вычислительной математики	нет	Изменений не требуется (протокол № 3 от 19.10.2023 г.)

ДОПОЛНЕНИЯ И ИЗМЕНЕНИЯ К УЧЕБНОЙ ПРОГРАММЕ ПО ИЗУЧАЕМОЙ УЧЕБНОЙ ДИСЦИПЛИНЕ

на _____/_____ учебный год

№ п/п	Дополнения и изменения	Основание

Учебная программа пересмотрена и одобрена на заседании кафедры
_____ (протокол № ____ от _____ 202_ г.)

Заведующий кафедрой
_____ _____ _____

УТВЕРЖДАЮ
Декан факультета
_____ _____ _____