Белорусский государственный университет

УТВЕРЖДАЮ
Проректор по учебной работе и образовательным инновациям
_____ О. Г. Прохоренко
«06» апреля 2022 г.

Регистрационный № УД – 10578/уч.

Алгоритмы и структуры данных

Учебная программа учреждения высшего образования по учебной дисциплине для специальности:

1-31 03 04 Информатика

1-97 01 02 Прикладная криптография

2022 г.

Учебная программа составлена на основе типовых учебных планов №G 31-1-029/пр-тип от 30.06.2021, №P 97-1-001/пр-тип от 23.07.2021, учебных планов № G 31-1-031/уч. от 30.06.2021, № G 31-1-021/уч.ин. от 23.07.2021, № P 97-1-002/уч. от 26.07.2021.

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;
А.А. Буславский – старший преподаватель кафедры дискретной математики и алгоритмики факультета прикладной математики и информатики Белорусского государственного университета.

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 № 13 от 17.03.2022 г.);

Научно-методическим Советом БГУ

(протокол № 4 от 18.03.2022 г.)

Заведующий кафедрой _____ В.М. Котов

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Цели и задачи учебной дисциплины

Учебная дисциплина «Алгоритмы и структуры данных» знакомит студентов с фундаментальными понятиями, используемыми при разработке алгоритмов и оценке их качества. В учебную программу учебной дисциплины включены разделы, позволяющие строить эффективные алгоритмы для разнообразных задач дискретной и комбинаторной оптимизации с использованием различных структур данных.

Цель преподавания учебной дисциплины состоит в формировании 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.

Учебная дисциплина относится **к модулю** «Дискретные структуры и алгоритмы» государственного компонента.

Программа составлена с учётом межпредметных **связей** с учебными дисциплинами. Для специальности 1-31 03 04 «Информатика» основой для изучения учебной дисциплины являются дисциплины «Дискретная математика и математическая логика» и «Основы теоретической информатики» модуля «Дискретные структуры и алгоритмы», «Основы и методологии программирования» модуля «Программирование». Знания, полученные в учебной дисциплине, используются при изучении дисциплины государственного компонента «Модели и алгоритмы задач дискретной оптимизации» модуля «Дискретные структуры и алгоритмы», дисциплины вузовского компонента «Исследование операций» модуля «Математические методы принятия решений».

Для специальности 1-97 01 02 «Прикладная криптография» основой для изучения учебной дисциплины являются дисциплины «Теория вероятностей

и математическая статистика» модуля «Основы теории вероятностей», «Основы и методология программирования» и «Язык программирования C++» модуля «Программирование». Знания, полученные в учебной дисциплине, используются при изучении дисциплины государственного компонента «Теория графов» модуля «Дискретные структуры и алгоритмы».

Требования к компетенциям

Освоение учебной дисциплины «Алгоритмы и структуры данных» должно обеспечить формирование следующей **базовой профессиональной компетенции**:

для специальности 1-31 03 04 «Информатика»:

БПК-6. Выполнять построение математических моделей и проводить их анализ в типовых задачах дискретной математики, интерпретировать получаемые результаты анализа математических моделей и осуществлять выбор структур данных для разработки эффективных алгоритмов решения прикладных задач.

для специальности 1-97 01 02 «Прикладная криптография»:

БПК-11. Реализовывать современные структуры данных, строить графовые модели и применять алгоритмы на графах для решения прикладных задач, обосновывать корректность алгоритма и оценивать его асимптотическую сложность.

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 трудоёмкость алгоритмов на основе рекуррентных соотношений,
- выбирать подходящие структуры данных при разработке эффективного алгоритма решения задачи,
- реализовывать поисковые деревья,
- строить графовые модели и применять базовые графовые алгоритмы.

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 4-м семестре.

Всего на изучение учебной дисциплины «Алгоритмы и структуры данных» отведено

для специальности 1-31 03 04 «Информатика»:

– для очной формы получения высшего образования – 216 часов, в том числе 102 аудиторных часа, из них: лекции – 52 часа, лабораторные занятия – 42 часа, управляемая самостоятельная работа – 8 часов.

Трудоёмкость учебной дисциплины составляет 6 зачётных единиц.

Форма текущей аттестации – экзамен.

для специальности 1-97 01 02 «Прикладная криптография»:

– для очной формы получения высшего образования – 198 часов, в том числе 102 аудиторных часа, из них: лекции – 52 часа, лабораторные занятия – 42 часа, управляемая самостоятельная работа – 8 часов.

Трудоёмкость учебной дисциплины составляет 6 зачётных единиц.

Форма текущей 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), дерево отрезков (англ. *segment tree*), дерево Фенвика (англ. *binary indexed tree*). Реализация базовых операций и их трудоёмкость.

Статическая версия задачи RMQ. Специальная структура данных «разрежённая таблица» (англ. *sparse table*). Реализация базовых операций и их трудоемкость.

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, красно-чёрные 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.

Тема 3.6. Структуры данных для работы со строками

Специальные структуры данных для работы со строками: z-функция, префикс функция, суффиксный массив, бор. Алгоритмы полиномиального хеширования в задачах на строки.

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. Алгоритм Диница.

Приложения максимального потока: наибольшее число путей между заданными множествами вершин, которые не пересекаются по дугам/вершинам; наибольшее паросочетание в двудольном графе, обладающее заданными свойствами.

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

УЧЕБНО-МЕТОДИЧЕСКАЯ КАРТА УЧЕБНОЙ ДИСЦИПЛИНЫ

Дневная форма получения образования с применением электронных средств обучения (ДО)

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**	**6**			**2**			
1.1	Введение. Основные понятия и определения	2						Собеседование
1.2	Рекуррентные уравнения и основные методы их решения	4			2			Электронный тест: трудоемкость алгоритмов (iRunner)
2	**Разработка эффективных алгоритмов**	**6**			**4**		**2**	
2.1	Метод «разделяй и властвуй»	2						Отчеты по домашним упражнениям с их устной защитой
2.2	Динамическое программирование	4			4		2	Контрольная работа №1.
3	**Структуры данных и абстрактные типы данных**	**22**			**20**		**4**	
3.1	Простейшие структуры данных	2			2			Отчеты по домашним

	и абстрактные типы данных							упражнениям с их устной защитой. Электронный тест: простейшие структуры данных (iRunner)
3.2	Специальная древовидная структура данных *куча*	4			4			Контрольная работа №2.
3.3	Специальная структура данных *система непересекающихся множеств*	2			2			Отчеты по домашним упражнениям с их устной защитой. Электронный тест: специализированные структуры данных (iRunner)
3.4	Структуры данных для решения задач на интервалах	4			4		2	Собеседование.
3.5	Структуры данных для организации поиска элемента	6			4		2	Дискуссия. Контрольная работа №3. Электронный тест: поисковые деревья (iRunner)
3.6	Структуры данных для работы со строками	4			4			Отчеты по домашним упражнениям с их устной защитой

4	**Алгоритмы на графах**	**18**			**16**		**2**	
4.1	Способы обхода вершин графа	6			6			Отчеты по домашним упражнениям с их устной защитой. Электронный тест: графы (iRunner)
4.2	Кратчайший маршрут	4			4			Коллоквиум по разделам 1-3.
4.3	Минимальное остовное дерево	2			2			Электронный итоговый тест: алгоритмы и структуры данных (iRunner)
4.4	Максимальный поток в сети и его приложения	6			4		2	Дискуссия.

ИНФОРМАЦИОННО-МЕТОДИЧЕСКАЯ ЧАСТЬ

Перечень основной литературы

1. 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).

2. Сборник задач по теории алгоритмов: учеб.-метод. пособие / В. М. Котов [и др.] – Минск : БГУ, 2017. – 183 с.

3. 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

4. Сборник задач по теории алгоритмов. Структуры данных: учеб.-метод. пособие / С. А. Соболь [и др.] – Минск : БГУ, 2020. – 159 с.

5. Теория алгоритмов: учеб. пособие / П. А. Иржавский [и др.] – Минск: БГУ, 2013. – 159 с.

Перечень дополнительной литературы

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

7. 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. – С. 263 – 267.

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

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

Электронные ресурсы

10. Образовательный портал БГУ [Электронный ресурс]. – Режим доступа: https://edufpmi.bsu.by/course/view.php?id=96. – Дата доступа: 02.09.2021.

11. Образовательная платформа Insight Runner [Электронный ресурс]. – Режим доступа: https://acm.bsu.by. – Дата доступа: 02.09.2021.

12. 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). [Электронный ресурс]. – Режим доступа: https://elib.bsu.by/handle/123456789/8522. – Дата доступа: 02.12.2021.

13. Теория алгоритмов: учеб. пособие / П. А. Иржавский [и др.] – Минск: БГУ, 2013. – 159 с. [Электронный ресурс]. – Режим доступа: https://elib.bsu.by/handle/123456789/91612. – Дата доступа: 02.12.2021.
14. Сборник задач по теории алгоритмов: учеб.-метод. пособие / В. М. Котов [и др.] – Минск : БГУ, 2017. – 183 с. [Электронный ресурс]. – Режим доступа: https://elib.bsu.by/handle/123456789/181529. – Дата доступа: 02.12.2021.
15. 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. – С. 263 – 267. Электронный ресурс]. – Режим доступа: https://elib.bsu.by/handle/123456789/231914. – Дата доступа: 02.12.2021.
16. Сборник задач по теории алгоритмов. Структуры данных: учеб.-метод. пособие / С. А. Соболь [и др.] – Минск : БГУ, 2020. – 159 с. Электронный ресурс]. – Режим доступа: https://elib.bsu.by/handle/123456789/255033. – Дата доступа: 02.12.2021

Перечень рекомендуемых средств диагностики и методика формирования итоговой оценки

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 текущей аттестации по дисциплине «Алгоритмы и структуры данных» для специальности 1-31 03 04 «Информатика» учебным планом предусмотрен экзамен, для специальности 1-97 01 02 «Прикладная криптография» - зачет.

При формировании итоговой оценки используется рейтинговая оценка знаний студента, дающая возможность проследить и оценить динамику процесса достижения целей обучения. Рейтинговая оценка предусматривает использование весовых коэффициентов для текущего контроля знаний студентов по дисциплине.

Примерные весовые коэффициенты, определяющие вклад текущего контроля знаний в рейтинговую оценку:

- отчет по домашним упражнениям с их устной защитой – 70 %;
- выполнение теста – 10%;
- контрольные работы – 10 %;
- коллоквиум – 10 %.

Рейтинговая оценка по дисциплине для специальности 1-31 03 04 «Информатика» рассчитывается на основе оценки текущей успеваемости и экзаменационной оценки с учетом их весовых коэффициентов. Вес оценки по текущей успеваемости составляет 40 %, экзаменационной оценки – 60 %.

Точки контроля по текущей успеваемости формируются из расчета общего количества часов (зачетных единиц), выделенных на изучение дисциплины.

Примерный перечень заданий для управляемой самостоятельной работы студентов

В качестве заданий для управляемой самостоятельной работы могут быть выданы задания для самостоятельного решения задач по следующим темам:

1. **Тема 2.2.** «Динамическое программирование»: задача оптимального перемножения группы матриц, задача расстановки k единиц в строке длины n, наибольший палиндром, наибольшая общая подстрока **(2 ч.).**

   Форма контроля – контрольная работа: программная реализация алгоритмов решения задач на любом высокоуровневом языке программирования и проверка их работоспособности на Образовательной платформе Insight Runner.
2. **Тема 3.4.** «Структуры данных для решения задач на интервалах»: префиксные суммы, корневая декомпозиция, дерево отрезков, дерево Фенвика, разрежённая таблица **(2 ч.).**

   Форма контроля – собеседование.
3. **Тема 3.5.** «Структуры данных для организации поиска элемента»: красно-черные деревья, splay-деревья **(2 ч.).**

   Форма контроля – дискуссия.
4. **Тема 4.4.** «Максимальный поток в сети и его приложения»: наибольшее паросочетание в двудольном графе минимального/максимального веса, максимальный поток минимальной стоимости **(2 ч.).**

   Форма контроля – дискуссия.

Примерная тематика лабораторных занятий

Занятие 1. Рекуррентные уравнения и основные методы их решения.

Занятия 2 – 3. Динамическое программирование.

Занятия 4. Простейшие абстрактные типы данных и структуры данных.

Занятие 5 – 6. Специальная структура данных «куча».

Занятие 7. Специальная структура данных система непересекающихся множеств.

Занятие 8 – 9. Структуры данных для решения задач на интервалах.

Занятие 10 – 11. Структуры данных для организации поиска элемента (сбалансированные поисковые деревья, хеш-таблицы).

Занятие 12 – 13. Структуры данных для работы со строками.

Занятие 14 –16. Способы обхода вершин графа.

Занятие 17 –18. Алгоритмы построения кратчайших маршрутов.

Занятие 19. Алгоритмы построения минимального остовного дерева.

Занятия 20 – 21. Максимальный поток в сети и его приложения.

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 в алгоритмах решения прикладных задач на графах».

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 образовательных стандартов высшего образования и учебно-программной документации, в т.ч. вопросы для подготовки к экзамену, задания, тесты, вопросы для самоконтроля, тематика рефератов и др., список рекомендуемой литературы, информационных ресурсов и др.).

При составлении заданий УСР 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 УСР 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. Задана библиотека алгоритмов на графах: алгоритм поиска в ширину, алгоритм поиска в глубину, алгоритм Дейкстры, алгоритм построения Эйлерова цикла, алгоритм Флойда-Варшалла, алгоритм Форда-Беллмана, алгоритм Диница, алгоритм топологической сортимровки, алгоритм Тарьяна, алгоритм Кана. Какие из указанных алгоритмов можно применить, если в графе нужно найти любой маршрут между заданной парой вершин? Какие из указанных алгоритмов можно применить, если в графе

нужно найти наименьший по количеству ребер маршрут между заданной парой вершин?

Задание 2. Приведите пример структур данных, которые используются для выполнения словарных операций. Какие из этих структур данных реализованы в высокоуровневых языках программирования?

Задание 3. Перечислите известные вам алгоритмы внутренней сортировки. Какие из этих алгоритмов реализованы в стандартных функциях сортировки высокоуровневых языков программирования (C++, Java, Python) и каковы особенности их реализации?

Задания на уровне воспроизведения

Задание 1. *Построить дерево*
Имя входного файла: *input.txt*
Имя выходного файла: *output.txt*
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*
Имя входного файла: *input.txt*
Имя выходного файла: *output.txt*
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*
Имя входного файла: *input.txt*
Имя выходного файла: *output.txt*
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 $u_i \ne v_i$, $1 \le u_i, v_i \le n$, $1 \le w_i \le 1000000$ 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. *Порядок перемножения матриц*.
Имя входного файла: *input.txt*
Имя выходного файла: *output.txt*
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 задано число s матриц ($2 \le s \le 100$).

В последующих s строках заданы размеры матриц: строка $i + 1$ 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 i от 1 до $s - 1$.

Формат выходных данных

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

input.txt	*output.txt*
3 2 3 3 5 5 10	130

Задание 2. *Единицы (часть 1).*
Имя входного файла: *стандартный ввод*
Имя выходного файла: *стандартный вывод*
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

Первая строка входного файла содержит 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 $10^9 + 7$.

стандартный ввод	*стандартный вывод*
3 2	3
4 0	1
5 4	5
6 4	15

Задание 3. *Единицы (часть 2).*
Имя входного файла: *стандартный ввод*
Имя выходного файла: *стандартный вывод*
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

Первая строка входного файла содержит два целых неотрицательных числа N и K, ($0 \le K \le N \le 1\ 000\ 000$).

Формат выходных данных

Выведите одно число – ответ на задачу.

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

стандартный ввод	*стандартный вывод*
3 2	3
4 0	1
5 4	5
6 4	15

Задание 4. *Палиндром*
Имя входного файла: *input.txt*
Имя выходного файла: *output.txt*
Ограничение по времени: 1 с
Ограничение по памяти: нет

Вводится непустая строка S, которая имеет длину не более 7000 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 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).

input.txt	*output.txt*
asddfsa	6 asddsa

Задание 5. *Преобразование строк*.
Имя входного файла: *int.txt*
Имя выходного файла: *out.txt*
Ограничение по времени: 1 с
Ограничение по памяти: нет

На вход подаются две символьные последовательности *A* и *B*, каждая последовательность 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 равна *z* баллов.

Формат входных данных

В первой строке находится число *x*.

Во второй строке — число *y*. В третьей строке — число *z*.

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

Формат выходных данных

Выведите минимальный суммарный штраф.

in.txt	*out.txt*
2 3 1 abcd bce	3

Задание 6. *Задача о сумме*.
Имя входного файла: *стандартный ввод*
Имя выходного файла: *стандартный вывод*
Ограничение по времени: 2 с
Ограничение по памяти: 256 МБ

Рассмотрим следующую модельную задачу. Изначально дана последовательность чисел A длины n (индексация с нуля):

$$a_0, a_1, a_2, \dots, a_{n-1}.$$

Поступают запросы двух типов:

Запрос модификации. Задан индекс i и число x. Нужно прибавить к i-ому элементу число x.

Запрос суммы. Задана пара индексов l и r. Нужно вычислить сумму элементов на полуинтервале $[l, r)$, т. е. $a_l + a_{l+1} + \cdots + a_{r-1}$, и вернуть результат.

Формат входных данных

В первой строке находится число n ($1 \le n \le 300\,000$).

Во второй строке записаны n чисел a_i. В третьей строке записано целое число q – количество запросов ($1 \le q \le 300\,000$).

Каждая из следующих q строк задает запрос.

Если это запрос модификации. То в строке записывается Add, затем индекс i ($0 \le i < n$) и число x ($10^{-9} \le x \le 10^9$).

Если это запрос суммы, то он задается словом FindSum и двумя индексами l и r ($0 \le l < r \le n$) – границами полуинтервала.

Формат выходных данных

Для каждого запроса второго типа выведите в отдельной строке сумму элементов на соответствующем полуинтервале.

in.txt	*out.txt*
5 10 30 40 -10 20 7 FindSum 1 2 FindSum 1 3 FindSum1 4 FindSum 3 5 Add 4 -10 FindSum 3 5 FindSum 0 5	30 70 60 10 0 80

Примерный перечень вопросов к зачету/экзамену

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, дерево отрезков, дерево Фенвика). 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”. 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. Красно-черное дерево. Поддержка инварианта сбалансированности. Базовые операции и их трудоёмкость (доказательство).

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

23. Структуры данных для работы со строками. Базовые операции и их трудоёмкость.

24. 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.

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

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

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

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

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

30. 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.

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

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

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

34. Приложения максимального потока: наибольшее паросочетание в двудольном графе; наибольшее паросочетание минимального веса в двудольном графе; наибольшее число попарно-непересекающихся путей между заданной парой вершин; максимальный поток минимальной стоимости: метод устранения отрицательных циклов; метод наименьших путей.

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)
Теория графов	Биомедицинской информатики	нет	Изменений не требуется (протокол № 13 от 17.03.2022 г.)
Исследование операций	Информационных систем управления	нет	Изменений не требуется (протокол № 13 от 17.03.2022 г.)
Модели и алгоритмы задач дискретной оптимизации	Дискретной математики и алгоритмии	нет	Изменений не требуется (протокол № 13 от 17.03.2022 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 _____ 201_ г.)

Заведующий кафедрой
_____ _____ _____

УТВЕРЖДАЮ
Декан факультета
_____ _____ _____