

**Федеральное государственное автономное образовательное  
учреждение высшего образования  
«Московский физико-технический институт  
(национальный исследовательский университет)»**

**УТВЕРЖДЕНО**

**Проректор по учебной работе и  
довузовской подготовке**

**А.А. Воронов**

	<b>Рабочая программа дисциплины (модуля)</b>
<b>по дисциплине:</b>	Алгоритмы на дискретных структурах данных
<b>по направлению:</b>	Информатика и вычислительная техника
<b>профиль подготовки:</b>	Прикладная математика и информатика Физтех-школа Прикладной Математики и Информатики кафедра алгоритмов и технологий программирования
<b>курс:</b>	1
<b>квалификация:</b>	магистр

Семестры, формы промежуточной аттестации:

1 (осенний) - Экзамен

2 (весенний) - Дифференцированный зачет

Аудиторных часов: 90 всего, в том числе:

лекции: 0 час.

семинары: 90 час.

лабораторные занятия: 0 час.

Самостоятельная работа: 105 час.

Подготовка к экзамену: 30 час.

Всего часов: 225, всего зач. ед.: 5

Количество контрольных работ, заданий: 2

Программу составили:

В.В. Яковлев, канд. физ.-мат. наук, заведующий кафедрой

О.Н. Ивченко, старший преподаватель

Программа обсуждена на заседании кафедры алгоритмов и технологий программирования 04.06.2020

## Аннотация

Курс дает базовые знания в области алгоритмов и структур данных. В явном виде они, может быть, не пригодятся, но очень важны для понимания работы библиотек, алгоритмов и языков программирования.

Курс состоит из трех частей. При просмотре видеозаписей лекций студенты получают необходимую теоретическую базу. На семинарских занятиях будут разбираться задачи, которые показывают применения и скрытые возможности пройденных структур данных.

Домашние задания по курсу закрепляют полученные знания и воспитывают хороший стиль написания кода, который позволяет избежать стандартных, но от этого ничуть не менее распространенных даже у опытных разработчиков, ошибок.

## 1. Цели и задачи

### Цель дисциплины

Дать студентам базовые знания в области алгоритмов и структур данных, важные для понимания работы библиотек, алгоритмов и языков программирования.

### Задачи дисциплины

1. Познакомиться с основными алгоритмами и структурами данных поиска.
2. Получить представление о проблемах, возникающих при применении известных алгоритмов анализа данных для решения практических задач поиска. Научиться преодолевать эти сложности имеющимися в распоряжении средствами.
3. Научиться оценивать учетную стоимость операций и алгоритмическую сложность кода.
4. Изучить задачи сортировки, модели вычислений, структуры данных с хранением истории, деревья поиска, задачи о динамическом поиске, алгоритмы обхода графов, поиска кратчайших путей, задачи подстроки в строке.
5. Получить практический опыт программирования, выработать хороший стиль написания кода, который позволяет избежать стандартных, но от этого ничуть не менее распространенных даже у опытных разработчиков, ошибок.

## 2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-2 Имеет представление об актуальных проблемах науки и техники в области информатики и вычислительной техники, способен на научном языке формулировать профессиональные задачи	ОПК-2.1 Имеет представление о современном состоянии исследований в рамках тематической области своей профессиональной деятельности
ОПК-4 Способен успешно реализовывать решение поставленной задачи, провести анализ результата и представить выводы, применяя знания и навыки в области математики, естественных наук и информационно-коммуникационных технологий	ОПК-4.2 Способен применять знание информационно-коммуникационных технологий для решения поставленной задачи, формулирования выводов и оценки полученных результатов

## 3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

Основные алгоритмы анализа данных, их преимущества и недостатки, а также структуры данных поиска.

уметь:

Использовать средства языка программирования C++ для разработки надежных и быстро работающих программных систем. Создавать качественный код для реализации алгоритмов анализа данных.

владеть:

Средствами разработки и тестирования программного кода на языке C++.

#### 4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

##### 4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Сложность и модели вычислений. Анализ учетных стоимостей.		6		9
2	Алгоритмы Merge-Sort и Quick-Sort.		6		8
3	Порядковые статистики. Кучи.		8		9
4	Хеширование.		5		8
5	Деревья поиска. Система непересекающихся множеств.		6		9
6	Задачи RMQ и LCA.		8		8
7	Структуры данных для геометрического поиска.		6		9
8	Динамическая связность в ненаправленном графе.		5		7
9	Обход в ширину. Обход в глубину. 2-разрезы.		8		6
10	Поиск кратчайших путей.		6		7
11	Минимальные остовные деревья. Минимальные разрезы.		8		6
12	Поиск подстрок.		6		6
13	Суффиксные деревья. Суффиксные массивы.		6		6
14	Длинейные общие подстроки. Приближенный поиск подстрок.		6		7
Итого часов			90		105
Подготовка к экзамену		30 час.			
Общая трудоёмкость		225 час., 5 зач.ед.			

##### 4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

###### Семестр: 1 (Осенний)

###### 1. Сложность и модели вычислений. Анализ учетных стоимостей.

Основные ресурсы: память и время. O-символика. Примеры моделей вычисления: машина Тьюринга, RAM-машина. Сложность в среднем и худшем случаях. Пример: задача сортировки. Сортировка выбором. Теоретико-информационная нижняя оценка сложности. Разрешающие деревья. Нижняя оценка сложности в модели разрешающих деревьев. Массивы переменного размера: аддитивная и мультипликативная схемы реаллокации. Анализ мультипликативной схемы для массива переменного размера с помощью банковского метода. Анализ учетных стоимостей операций: функция потенциала, истинные и учетные стоимости. Стеки и очереди.

###### 2. Алгоритмы Merge-Sort и Quick-Sort.

Понятие о методе «разделяй и властвуй». Алгоритм Merge-Sort. Слияние двух упорядоченных списков. Оценка сложности. K-way Merge-Sort для работы во внешней памяти. Сортировка слиянием без использования дополнительной памяти. Общая схема алгоритма Quick-Sort. Два варианта реализации Partition. Примеры неудачного выбора опорных элементов.

### 3. Порядковые статистики. Кучи.

Нахождение порядковых статистик с помощью рандомизированной модификации алгоритма Quick-Sort. Линейность матожидания времени работы. Приближенные медианы. Выбор k-й порядковой статистики за линейное в худшем случае. Деревья со свойствами кучи. Почти полные бинарные деревья: нумерация вершин, навигация. Двоичная куча. Операция просеивания вниз и вверх.

### 4. Хеширование.

Хеш-функции. Коллизии. Разрешение коллизий методом цепочек, методом последовательных проб и методом двойного хеширования. Гипотеза простого равномерного хеширования, оценка средней длины цепочки. Универсальные семейства хеш-функций, оценка средней длины цепочки.

### 5. Деревья поиска. Система непересекающихся множеств.

Определение дерева поиска. Вставка и удаление элементов. Inorder-обход дерева. Красно-черные деревья: определение и основные свойства. Реализация операций вставки для красно-черного дерева. Splay-деревья. Операция splay: zig, zig-zig и zig-zag шаги. Реализация операций вставки, удаления, слияния и разделения для splay-деревьев. Декартовы деревья (дучи). Единственность декартова дерева для заданного набора различных ключей и приоритетов. Логарифмическая оценка матожидания высоты дучи. Операции слияния и разделения для дуч. Операции вставки и удаления элементов для дуч.

### 6. Задачи RMQ и LCA.

Задачи RMQ (range minimum query) и LCA (least common ancestor). Сведение от задачи RMQ к задаче LCA, декартово дерево. Алгоритм Таржана для offline-версии задачи LCA. Простейшие алгоритмы для online-версии задачи LCA: полная и разреженная таблицы ответов. Алгоритм Фарах-Колтона-Бендера для задачи  $\pm 1$ -RMQ. Сведение задачи LCA к задаче  $\pm 1$ -RMQ: эйлеров обход дерева.

### 7. Структуры данных для геометрического поиска.

Location problem, stabbing problem. Деревья интервалов. Сведение системы интервалов к двумерной задаче. Задача поиска точек в коридоре. Priority search tree. Задача поиска точек в прямоугольнике. Дерево отрезков по координате X, упорядоченные по Y списки точек в каждой вершине. Сложность  $O(n \log n)$  для построения и  $O(\log^2 n)$  для запроса. Уменьшение времени поиска до  $O(\log n)$ . Задача одновременного поиска в наборе упорядоченных списков. Fractional cascading.

## Семестр: 2 (Весенний)

### 8. Динамическая связность в ненаправленном графе.

Location problem, stabbing problem. Деревья интервалов. Сведение системы интервалов к двумерной задаче. Задача поиска точек в коридоре. Priority search tree. Задача поиска точек в прямоугольнике. Дерево отрезков по координате X, упорядоченные по Y списки точек в каждой вершине. Сложность  $O(n \log n)$  для построения и  $O(\log^2 n)$  для запроса. Уменьшение времени поиска до  $O(\log n)$ . Задача одновременного поиска в наборе упорядоченных списков. Fractional cascading.

### 9. Обход в ширину. Обход в глубину. 2-разрезы.

Графы: основные определения, обозначения и способы хранения. Обход в ширину и его использование для нахождения кратчайших путей. Обход в глубину и его основные свойства. Эйлеровы обходы графов. Построение эйлерового обхода с помощью варианта обхода в глубину. Дерево обхода в глубину. Классификация дуг графа относительно дерева обхода в глубину (дуги дерева, обратные, прямые, перекрестные). Отсутствие перекрестных дуг при поиске в ненаправленном графе. Точки сочленения и их нахождение за линейное время с помощью обхода в глубину. Отношение взаимной достижимости вершин. Компоненты сильной связности, конденсация. Ацикличность конденсации.

#### 10. Поиск кратчайших путей.

Кратчайшие пути в графе, примеры функции длин. Оценки расстояний и их релаксация. Алгоритмы Форда-Беллмана и Флойда. Алгоритм Дейкстры. Критерий консервативности функции длин дуг в терминах наличия допустимого набора потенциалов. Алгоритм Джонсона для задачи APSP при произвольных длинах дуг. Использование маяков (landmarks) для быстрого поиска кратчайших путей. Алгоритм ALT.

#### 11. Минимальные остовные деревья. Минимальные разрезы.

Задача об оптимальном остовном дереве. Хорошие множества, лемма о минимальном ребре в разрезе. Алгоритмы Краскала, Прима и Борушки. Оценки сложности. Задачи о минимальном глобальном разрезе и о минимальном s-t разрезе, их связь. Стягивания графа. Алгоритм Штёра-Вагнера.

#### 12. Поиск подстрок.

Z-функция: определение и использование в задаче поиска подстроки. Построение Z-функции за линейное время. Оптимизация поиска подстрок с помощью Z-функции по памяти. Использование Z-функции для задачи приближенного поиска подстрок с одной ошибкой за линейное время. Задача множественного поиска подстрок, ожидаемая асимптотика времени работы. Бор для набора слов: определение и способы представления. Префикс-функция на боре. Алгоритм Ахо--Корасик для множественного поиска подстрок.

#### 13. Суффиксные деревья. Суффиксные массивы.

Общая схема алгоритма Укконена для построения сжатого суффиксного дерева за время, линейное по длине строки. Итерации и шаги алгоритма. Классификация шагов. Лемма о возможных переходах между шагами различных типов. Элиминация шагов типа 1: неявные пометки листовых дуг. Элиминация шагов типа 3: досрочное окончание итерации. Оценка количества шагов типа 2. Поиск положений для шагов типа 2: суффиксные ссылки. Прием «скачок по счетчику» для быстрого вычисления суффиксных ссылок. Лемма об изменении вершинной глубины при переходе по суффиксной ссылке.

#### 14. Длиннейшие общие подстроки. Приближенный поиск подстрок.

Задача приближенного поиска подстрок в тексте. Формулировка в терминах расстояний по графу динамического программирования. Алгоритм Ландау-Вишкина: множества достижимых вершин и их границы. База и шаг алгоритма, использование LCP.

### 5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Учебная аудитория, оснащенная компьютером и мультимедийным оборудованием (проектор, звуковая система).

### 6. Перечень рекомендуемой литературы

#### Основная литература

1. Алгоритмы [Текст] : [учеб. пособие для вузов] / С. Дасгупта, Х. Пападимитриу, У. Вазирани ; пер. с англ. А. А. Куликова ; под ред. А. Шеня .— М. : МЦНМО, 2014 .— 320 с.
2. Дискретный анализ. Формальные системы и алгоритмы [Текст], учеб. пособие для академического бакалавриата /Ю. И. Журавлев, Ю. А. Флёров, М. Н. Вялый. М., Юрайт, 2019

#### Дополнительная литература

1. Алгоритмы и структуры данных на языке С [Текст] / В. В. Прут - М.МФТИ,2016

### **7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)**

1. [http://en.wikipedia.org/wiki/Introduction\\_to\\_Algorithms](http://en.wikipedia.org/wiki/Introduction_to_Algorithms)
2. [http://en.wikipedia.org/wiki/Design\\_Patterns](http://en.wikipedia.org/wiki/Design_Patterns)
3. [http://en.wikipedia.org/wiki/Code\\_Complete](http://en.wikipedia.org/wiki/Code_Complete)
4. <http://misko.hevery.com/2008/11/11/clean-code-talks-dependency-injection/>
5. <http://misko.hevery.com/attachments/Guide-Writing%20Testable%20Code.pdf>
6. [http://en.wikipedia.org/wiki/The\\_Pragmatic\\_Programmer](http://en.wikipedia.org/wiki/The_Pragmatic_Programmer)
7. <http://lib.ru/CPPHB/cpptut.txt> учебник Бьёрн Страуструп, "Язык C++"

#### Список дополнительной литературы:

2. Scott Meyers, Effective C++: 55 Specific Ways to Improve Your Programs and Designs, 3rd Edition
3. Martin Fowler, Refactoring: Improving the Design of Existing Code
4. Scott Meyers, More Effective C++: 35 New Ways to Improve Your Programs and Designs
5. Scott Meyers, Effective STL: 50 Specific Ways to Improve Your Use of the Standard Template Library

### **8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)**

На занятиях используются мультимедийные технологии, включая демонстрацию презентаций.

Для контроля и коррекции знаний обучающиеся могут использовать компьютерное тестирование, в том числе на портале [www.i-exam.ru](http://www.i-exam.ru).

В процессе самостоятельной работы обучающихся возможно использование таких программных средств, как Mathcad, Scilab и др.

### **9. Методические указания для обучающихся по освоению дисциплины (модуля)**

Решение задач по заданию преподавателя – решаются задачи, выданные преподавателем по итогам лекционных занятий, используются электронный конспект лекций и учебники, рекомендуемые данной программой.

Самостоятельная работа студента:

- 4 домашних задания в течение семестра; прием работы происходит удаленно в виде автоматизированных тестов и ручного ревью кода.

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)**

<b>по направлению:</b>	Информатика и вычислительная техника
<b>профиль подготовки:</b>	Прикладная математика и информатика Физтех-школа Прикладной Математики и Информатики кафедра алгоритмов и технологий программирования
<b>курс:</b>	<u>1</u>
<b>квалификация:</b>	магистр

Семестры, формы промежуточной аттестации:

1 (осенний) - Экзамен

2 (весенний) - Дифференцированный зачет

**Разработчики:**

В.В. Яковлев, канд. физ.-мат. наук, заведующий кафедрой

О.Н. Ивченко, старший преподаватель

## 1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-2 Имеет представление об актуальных проблемах науки и техники в области информатики и вычислительной техники, способен на научном языке формулировать профессиональные задачи	ОПК-2.1 Имеет представление о современном состоянии исследований в рамках тематической области своей профессиональной деятельности
ОПК-4 Способен успешно реализовывать решение поставленной задачи, провести анализ результата и представить выводы, применяя знания и навыки в области математики, естественных наук и информационно-коммуникационных технологий	ОПК-4.2 Способен применять знание информационно-коммуникационных технологий для решения поставленной задачи, формулирования выводов и оценки полученных результатов

## 2. Показатели оценивания компетенций

В результате изучения дисциплины «Алгоритмы на дискретных структурах данных» обучающийся должен:

### знать:

Основные алгоритмы анализа данных, их преимущества и недостатки, а также структуры данных поиска.

### уметь:

Использовать средства языка программирования C++ для разработки надежных и быстро работающих программных систем. Создавать качественный код для реализации алгоритмов анализа данных.

### владеть:

Средствами разработки и тестирования программного кода на языке C++.

## 3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

Пример задачи для Code review

Вы хотите набрать футбольную команду. У каждого игрока своя эффективность, она описывается одним целым числом. Чем больше число, тем больше эффективность футболиста. Обязательным условием для любой команды является сплоченность. Если один из игроков играет сильно лучше всех остальных, его будут недолюбливать, и команда распадется. Поэтому эффективность любого игрока команды не должна превышать сумму эффективностей любых двух других игроков. Ваша задача — набрать команду, которая будет удовлетворять условию сплоченности, и при этом иметь наибольшую суммарную эффективность.

## 4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

Базовые задания:

1. Реализация на основе массива переменного размера и на основе связанного списка. Моделирование очереди с помощью двух стеков. Задача о поддержании динамического максимума в стеке и очереди.
2. Изменяемые (mutable) и неизменяемые (immutable) структуры данных. Структуры данных с хранением истории (persistent). Immutable-стек и immutable-очередь. Проблема множественного будущего при анализе учетных стоимостей в persistent-структурах.
3. Рандомизированный выбор опорного элемента. Сложность Quick-Sort в худшем и среднем случаях. Глубина рекурсии в худшем и среднем случаях. Элиминация хвостовой рекурсии. Задача об оптимальном дереве слияний.



4. Коды Хаффмана. Слияние двух упорядоченных последовательностей различной длины. Теоретико-информационная нижняя оценка. Бинарный поиск "от края" (galloping).
5. Реализация операций вставки, удаления и поиска минимума. Преобразование произвольного массива ключей в кучу (операция Make-Heap), линейность времени работы. Алгоритм сортировки Heap-Sort. k-ичные кучи, зависимость сложности операций от выбора k. Биномиальные (binomial), левачьи (leftlist) и косые (skew) кучи.
6. Построение универсального семейства для целочисленных ключей. Совершенные хеш-функции. Построение совершенной хеш-функции с помощью универсального семейства. Интерфейс множества с ошибками. Фильтр Блюма (Bloomfilter). Оценка вероятности ложноположительного срабатывания. Интерфейс словаря с ошибками. Модификация фильтра Блюма (bloomierfilter).
7. Построение декартового дерева за линейное время при условии предварительной сортировки ключей. B+ деревья: определения и основные свойства. Операции поиска, вставки и удаления для B+ деревьев. Системы непересекающихся множеств.
8. Реализация с использованием леса. Ранги вершин, эвристика ранга. Логарифмическая оценка ранга через количество элементов. Рандомизированная ранговая эвристика. Эвристика сжатия путей. Оценка учетной стоимости операций (без доказательства).
9. Поиск сильно связанных компонент с помощью обхода в глубину. Топологическая сортировка конденсации. 1- и 2-разрезы. Линейное пространство на ребрах графа, его размерность. Подпространство циклов графа, его размерность и базис. Подпространство разрезов графа.
10. Разложение пространства в прямую ортогональную сумму подпространств циклов и разрезов. Генерация случайного равновероятного элемента в пространстве циклов. Рандомизированный поиск 2-разрезов на основе fingerprints.cascading.
11. Анализ сложности алгоритма Дейкстры. Использование бинарных и k-ичных куч. Двухнаправленный алгоритм Дейкстры. Системы потенциалов в задаче о кратчайших путях.
12. Поиск подстрок: джокеры типа "?" и "\*". Использование алгоритма Ахо-Корасик. Оценка сложности. Редукция по символу алфавита. Поиск подходящих позиций с использованием сверток. Связь сверток с умножением полиномов. Пара оптимизаций: бинаризация алфавита и сокращение длины текста с помощью двуслойного покрытия.
13. Доказательство линейности времени работы алгоритма Укконена. Суффиксные массивы. Поиск подстрок с помощью суффиксных массивов и бинарного поиска. Построение суффиксного массива за время  $O(n \log n)$  методом маркировки (алгоритм Карпа-Миллера-Розенберга).
14. Задача о длиннейшей общей подстроке. Решение за линейное время с помощью суффиксного дерева и суффиксного массива.

#### Билет 1

Задача о длиннейшей общей подстроке. Решение за линейное время с помощью суффиксного дерева и суффиксного массива.

Построение декартового дерева за линейное время при условии предварительной сортировки ключей. B+ деревья: определения и основные свойства. Операции поиска, вставки и удаления для B+ деревьев. Системы непересекающихся множеств.

#### Билет 2

Реализация на основе массива переменного размера и на основе связанного списка. Моделирование очереди с помощью двух стеков. Задача о поддержании динамического максимума в стеке и очереди.

Рандомизированный выбор опорного элемента. Сложность Quick-Sort в худшем и среднем случаях. Глубина рекурсии в худшем и среднем случаях. Элиминация хвостовой рекурсии. Задача об оптимальном дереве слияний.

#### Критерии оценивания

отлично (10) - выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений.

отлично (9) - выставляется студенту, показавшему свободное оперирование знаниями учебной программы дисциплины, выполнение заданий творческого характера.

отлично (8) - выставляется студенту, показавшему владение программным учебным материалом с наличием несущественных ошибок в действиях, самостоятельно исправляемых учащимся.

хорошо (7) - выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускается в ответе или в решении задач некоторые неточности.

хорошо (6) - выставляется студенту если он осознает воспроизведение программного учебного материала, в том числе и различной степени сложности, с несущественными ошибками, затруднения в применении отдельных навыков.

хорошо (5) - выставляется студенту если теоретическое содержание освоено не полностью, некоторые практические навыки сформированы недостаточно, в некоторых случаях были допущены ошибки.

удовлетворительно (4) - выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и может применять полученные знания по образцу в стандартной ситуации.

удовлетворительно (3) - выставляется студенту в случае большого количества недочетов и неправильных ответов, а также пассивной работе в ходе занятий, многие учебные задания не выполнены.

неудовлетворительно (2) - выставляется студенту, который не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных понятий дисциплины и не умеет использовать полученные знания при решении типовых практических задач.

неудовлетворительно (1) - выставляется студенту, который не освоил теоретическое и практическое содержание курса, все выполненные учебные задания содержат грубые ошибки.

## **5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности**

Во время проведения дифференцированного зачета/экзамена по дисциплине обучающиеся могут пользоваться программой дисциплины. Дифференцированный зачет и экзамен проходят путем специального опроса, проводимого в устной форме.