

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

**УТВЕРЖДЕНО**  
**Проректор по учебной работе**

**А.А. Воронов**

	<b>Рабочая программа дисциплины (модуля)</b>
<b>по дисциплине:</b>	Алгоритмы и структуры данных на Python
<b>по направлению:</b>	Прикладные математика и физика
<b>профиль подготовки:</b>	Управление инновациями в бизнесе
	Физтех-школа бизнеса высоких технологий
	кафедра информатики и вычислительной математики
<b>курс:</b>	1
<b>квалификация:</b>	бакалавр

Семестр, формы промежуточной аттестации: 2 (весенний) - Дифференцированный зачет

Аудиторных часов: 80 всего, в том числе:

лекции: 20 час.

семинары: 0 час.

лабораторные занятия: 60 час.

Самостоятельная работа: 100 час.

Всего часов: 180, всего зач. ед.: 4

Количество контрольных работ, заданий: 2

Программу составил: Т.Ф. Хирьянов, старший преподаватель

Программа обсуждена на заседании кафедры информатики и вычислительной математики 27.04.2022

## Аннотация

Курс является фундаментальной базой для дальнейшего изучения программирования. Студенты ознакомятся с классическими алгоритмами и структурами данных, приобретут практику реализации сложных и эффективных алгоритмов.

### 1. Цели и задачи

#### Цель дисциплины

Изучить некоторые классические алгоритмы и структуры данных в реализации на языке Python 3.

#### Задачи дисциплины

1. изложить основы теории сложности алгоритмов;
2. научить студентов обращаться с классическими структурами данных: очередями, стеками, хеш-таблицами;
3. научить студентов использовать теорию графов и алгоритмы обхода графов для решения задач;
4. научить студентов целесообразно применять различные алгоритмы поиска в тексте и обработки текстовой информации;
5. развить у обучающихся навык использования языка программирования Python 3 для решения конкретных прикладных задач.

### 2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-4 Способен осуществлять сбор и обработку научно-технической и (или) технологической информации для решения фундаментальных и прикладных задач	ОПК-4.1 Владеет методами научного поиска и интеллектуального анализа информации при решении задач профессиональной деятельности
ПК-1 Способен планировать и проводить научные эксперименты (в избранной предметной области) и (или) теоретические (аналитические и имитационные) исследования	ПК-1.8 Владеет навыками работы с современными языками программирования и программными пакетами для научных расчетов
ПК-3 Способен выбирать и применять подходящее оборудование, инструменты и методы исследований для решения задач в избранной предметной области	ПК-3.2 Знает области и критерии применимости используемых теоретических подходов и умение оценивать точность приближенных аналитических методов вычислений

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

В результате освоения дисциплины обучающиеся должны

знать:

- общие понятия о структурах данных: стеки, очереди, списки, хеш-таблицы;
- способы хранения графов и деревьев в памяти ЭВМ и алгоритмы их обработки;
- основные алгоритмы эффективного поиска в тексте и его обработки;
- границы применимости изученных алгоритмов и их свойства;
- основы теории сложности алгоритмов, проблемы алгоритмической сложности.

уметь:

- выбирать оптимальные алгоритмы для масштабируемых программ;
- реализовывать известные алгоритмы на языке программирования Python;
- находить и устранять ошибки в алгоритмах на Python с использованием современных средств написания и отладки программ.

владеть:

- навыками программирования для решения исследовательских задач;
- языком программирования Python в объеме, необходимом для реализации изучаемых алгоритмов;
- средствами отладки программ на Python;
- навыками применения коллекций стандартной библиотеки Python, реализующих необходимые структуры данных;
- основами работы со стандартными и дополнительными прикладными пакетами Python.

#### 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	Эффективные сортировки массива	2		6	10
2	Стек, дек и очередь	2		6	10
3	Хеш-таблицы	2		6	10
4	Введение в теорию графов	2		6	10
5	Обход графа в глубину	2		6	10
6	Обход графа в ширину	2		6	10
7	Динамическое программирование на графах	2		6	10
8	NP-алгоритмы на графах	2		6	10
9	Поиск подстроки в строке	2		6	10
10	Эффективная обработка строк	2		6	10
Итого часов		20		60	100
Подготовка к экзамену		0 час.			
Общая трудоёмкость		180 час., 4 зач.ед.			

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

###### Семестр: 2 (Весенний)

###### 1. Эффективные сортировки массива

Рекурсивные сортировки. Быстрая сортировка. Сортировка слиянием.

Модуль heapq

Пирамида (куча). Пирамидальная сортировка.

Устойчивость сортировок.

###### 2. Стек, дек и очередь

Стек. Дек.

Очередь.

Очередь с приоритетами. Пирамида (куча).

Очередь событий графического приложения.

###### 3. Хеш-таблицы

Хеш-функция. Хеширование.

Открытая хеш-таблица.

Закрытая хеш-таблица.

Проблема удаления из закрытой хеш-таблицы. Перехеширование.

#### 4. Введение в теорию графов

Введение в теорию графов.

Взвешенный граф.

Пути и циклы в графах.

Эйлеров цикл. Эйлеров путь.

Расстояние между двумя вершинами.

Графы и способы их представления: список рёбер, матрица смежности, списки смежности

#### 5. Обход графа в глубину

Определение дерева.

Остовное дерево графа.

Минимальное остовное дерево. Алгоритм Прима.

Поиск в глубину.

Связность неориентированных графов: выделение компонент связности.

#### 6. Обход графа в ширину

Обход графа в ширину.

Прикладные применения обхода в ширину.

Алгоритм Дейкстры.

Восстановление кратчайшего пути.

#### 7. Динамическое программирование на графах

Простые случаи ДП на графах.

Алгоритм Флойда-Уоршелла

Алгоритм Беллмана-Форда

#### 8. NP-алгоритмы на графах

Проверка изоморфизма графов.

Задача о коммивояжере.

Гамильтонов цикл.

NP-полные задачи: решение среди экспоненциального множества кандидатов.

Сложные и простые задачи: сравнение нескольких пар задач, которые формулируются похоже, но имеют разную сложность.

Приближенные алгоритмы для NP-полных задач.

#### 9. Поиск подстроки в строке

Алгоритм Кнута-Морриса-Пратта

Z-алгоритм

Алгоритм Рабина-Карпа

#### 10. Эффективная обработка строк

Конечные автоматы для поиска подстрок и регулярных выражений

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

Большая лекционная аудитория, подходящая для учебного потока (факультет, оснащённая мультимедиа проектором и экраном для чтения лекций.

Учебные аудитории — сетевые компьютерные классы с установленным необходимым программным обеспечением.

## **6. Перечень рекомендуемой литературы**

### Основная литература

1. Python 3. Самое необходимое / Н. А. Прохоренко, В. А. Дронов, Санкт-Петербург, БХВ, 2021
2. Алгоритмы. Руководство по разработке [Текст], [учеб. пособие для вузов] /С. Скиена ; [пер. с англ. С. Таранушенко], The Algorithm, esign Manual. -СПб., БХВ-Петербург, 2018

### Дополнительная литература

1. Программирование на Python 3, подробное руководство/М. Саммерфилд,-СПб, Символ-Плюс, 2020
2. Дискретная математика для программистов, учебное пособие / Р. Хаггарты . — Москва, Техносфера, 2012.— URL: <https://ibooks.ru/bookshelf/337430/reading> (дата обращения: 26.11.2020). - Полный текст (Режим доступа : из сети МФТИ / Удаленный доступ)
3. Алгоритмы и программы на языках С и PYTHON. Сортировка. Поиск. Строки, Электронная версия печатной публикации / В. В. Прут. — Москва, МФТИ, 2020

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

Не используются

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

На ПК в компьютерных классах должно быть установлено следующее ПО:

1. Операционная система GNU/Linux;
2. Интерпретатор Python версии не ниже 3.9;
3. Среда разработки IDLE;
4. Среды JupyterLab, Jupyter Notebook, Ipython;
5. Библиотеки Numpy, Pandas, xlrd, NetworkX, Matplotlib, Seaborn и PyGame для Python 3;
6. Среда разработки JetBrains Python Charm community edition;

На лекциях используются мультимедийные технологии, включая демонстрацию презентаций.

Для контроля и коррекции знаний обучающихся используются автоматизированное компьютерное тестирование на основе Ejudge или CMS Moodle.

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

Изложение материала происходит преимущественно на лекциях, сопровождается мультимедиа-презентацией с примерами кода и блок-схемами алгоритмов. На лабораторных занятиях также происходит изложение нового материала: в начале каждой лабораторной работы и далее по мере необходимости. На контрольных работах изложение нового материала исключено, преподаватель оказывает только консультации по условиям задач.

Учёт, контроль и оценка знаний студентов

В течение лабораторной работы успеваемость отслеживается по результатам контестов, а также по своевременности сдачи лабораторных работ. Таким образом достигается раннее выявление отстающих студентов с передачей докладных в деканат.

Посещаемость лекций не отмечается, но каждый контест завязан на материал прошедшей лекции, что делает посещение лекций насущной необходимостью в течение семестра.

Дифференцированный зачёт принимается в устной форме, при этом учитываются оценки по контрольным и оценки по практическим лабораторным работам. Устный ответ практически исключает списывание, показывает владение базовой терминологией предмета, умение говорить на языке информатики, а также позволяет проверить знание сложных алгоритмов, которые долго программируются, но могут быть относительно легко устно объяснены.

Самостоятельная домашняя работа предполагается после каждой лабораторной работы.

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)**

**по направлению:** Прикладные математика и физика  
**профиль подготовки:** Управление инновациями в бизнесе  
Физтех-школа бизнеса высоких технологий  
кафедра информатики и вычислительной математики  
**курс:** 1  
**квалификация:** бакалавр

Семестр, формы промежуточной аттестации: 2 (весенний) - Дифференцированный зачет

**Разработчик:** Т.Ф. Хирьянов, старший преподаватель

## 1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-4 Способен осуществлять сбор и обработку научно-технической и (или) технологической информации для решения фундаментальных и прикладных задач	ОПК-4.1 Владеет методами научного поиска и интеллектуального анализа информации при решении задач профессиональной деятельности
ПК-1 Способен планировать и проводить научные эксперименты (в избранной предметной области) и (или) теоретические (аналитические и имитационные) исследования	ПК-1.8 Владеет навыками работы с современными языками программирования и программными пакетами для научных расчетов
ПК-3 Способен выбирать и применять подходящее оборудование, инструменты и методы исследований для решения задач в избранной предметной области	ПК-3.2 Знает области и критерии применимости используемых теоретических подходов и умение оценивать точность приближенных аналитических методов вычислений

## 2. Показатели оценивания компетенций

В результате изучения дисциплины «Алгоритмы и структуры данных на Python» обучающийся должен:

### знать:

- общие понятия о структурах данных: стеки, очереди, списки, хеш-таблицы;
- способы хранения графов и деревьев в памяти ЭВМ и алгоритмы их обработки;
- основные алгоритмы эффективного поиска в тексте и его обработки;
- границы применимости изученных алгоритмов и их свойства;
- основы теории сложности алгоритмов, проблемы алгоритмической сложности.

### уметь:

- выбирать оптимальные алгоритмы для масштабируемых программ;
- реализовывать известные алгоритмы на языке программирования Python;
- находить и устранять ошибки в алгоритмах на Python с использованием современных средств написания и отладки программ.

### владеть:

- навыками программирования для решения исследовательских задач;
- языком программирования Python в объеме, необходимом для реализации изучаемых алгоритмов;
- средствами отладки программ на Python;
- навыками применения коллекций стандартной библиотеки Python, реализующих необходимые структуры данных;
- основами работы со стандартными и дополнительными прикладными пакетами Python.

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

С целью контроля освоения обучающимися учебного материала проводится устный опрос в начале занятия по теме прошлого занятия.

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

1. Рекурсивные сортировки. Быстрая сортировка. Сортировка слиянием.
2. Пирамида (куча). Пирамидальная сортировка.
3. Устойчивость сортировок.
4. Списки: односвязный, двусвязный, кольцо.
5. Стек. Дек.
6. Очередь.



7. Очередь с приоритетами. Пирамида (куча).
8. Хеш-функция. Хеширование. Открытая и закрытая хеш-таблица.
9. Графы и способы их представления: список рёбер, матрица смежности, списки смежности
10. Определение дерева. Поиск в глубину.
11. Связность неориентированных графов: выделение компонент связности.
12. Поиск в ширину. Алгоритм Дейкстры.
13. Эйлеров цикл. Эйлеров путь.
14. Взвешенный граф. Кратчайшее расстояние между двумя вершинами.
15. Алгоритм Флойда-Уоршелла.
16. Минимальное остовное дерево. Алгоритм Прима.
17. Проверка изоморфизма графов.
18. Постоение гамильтонова цикла.
19. Задача о коммивояжере
20. Орграфы. Топологическая сортировка.
21. Проверка равенства строк. Простой и вероятностный алгоритмы.
22. Поиск подстроки в строке. Алгоритм Рабина-Карпа.
23. Алгоритм Кнута-Морриса-Пратта.
24. Z-алгоритм.
25. Конечный автомат для поиска подстрок и регулярных выражений.

#### Критерии оценивания

Оценка по десятибалльной шкале за работу на лабораторном практикуме выставляется преподавателем практикума исходя из количества и качества выполненных практических работ за семестр.

Оценка за выполнение контестов выставляется автоматически исходя из суммарного рейтинга обучающегося в системе Ejudge и также нормируется к десятибалльной шкале.

Итоговая оценка за зачёт не должна отличаться от среднего арифметического оценок по контестам и по практическим лабораторным работам более чем на три балла.

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

Дифференцированный зачёт принимается в устной форме с учётом оценки по контестам и оценки по лабораторному практикуму. Устный ответ практически исключает списывание, показывает владение базовой терминологией предмета, умение говорить на языке информатики, а также позволяет проверить знание сложных алгоритмов, которые долго программируются, но могут быть относительно легко устно объяснены.

На дифференцированном зачёте предлагается ответить на два-три вопроса по теории и решить одну короткую алгоритмическую задачу на бумаге без использования компьютера.

Пример задания на устном зачёте:

1. Очередь.
2. Эйлеров цикл. Эйлеров путь.
3. Задача: реализовать алгоритм Рабина-Карпа с полиномиальной хеш-функцией.