

**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>	Информатика
<b>по направлению:</b>	Системный анализ и управление
<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: 1 (осенний) - Экзамен

Аудиторных часов: 60 всего, в том числе:

лекции: 20 час.

семинары: 0 час.

лабораторные занятия: 40 час.

Самостоятельная работа: 45 час.

Подготовка к экзамену: 30 час.

Всего часов: 135, всего зач. ед.: 3

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 базой для дальнейшего детального изучения средств программирования и алгоритмов. Студенты ознакомятся с классическими алгоритмами обработки данных и структурами данных, приобретут практику реализации базовых алгоритмов.

### 1. Цели и задачи

#### Цель дисциплины

Научить студентов основам информатики и ИКТ, а также программированию на языке Python 3 на уровне, достаточном для прохождения последующих курсов.

#### Задачи дисциплины

1. Обеспечить чёткое понимание студентами основ информатики и ИКТ, включая некоторые области математики (системы счисления, логика, дискретная математика);
2. сформировать у обучающихся представление о архитектуре ЭВМ, операционной системе и прикладных вычислительных процессах;
3. обучить студентов базовым алгоритмам обработки числовой и текстовой информации;
4. сформировать 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-6 Способен применять математические, системно-аналитические, вычислительные методы и программные средства для решения прикладных задач в области создания систем анализа и автоматического управления и их компонентов	ОПК-6.3 Использует программные средства для разработки информационных систем

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

В результате освоения дисциплины обучающиеся должны

знать:

- основы архитектуры электронно-вычислительной машины (ЭВМ), представление информации в ЭВМ;
- общие характеристики интерпретируемых и компилируемых языков программирования;
- основные принципы устройства и работы операционной системы;
- основы алгоритмического языка программирования Python;
- приёмы разработки программ.

уметь:

- использовать знания по информатике для приложений в инновационной, конструкторско-технологической и производственно-технологической сферах деятельности;
- разрабатывать небольшие программы на языке высокого уровня Python;
- использовать специализированные среды программирования для написания и отладки программ;
- выбирать адекватные алгоритмы для написания программ по обработке числовой и текстовой информации.

владеть:

- навыками программирования для решения исследовательских задач на языке программирования Python;
- средствами отладки программ на Python;
- основами работы с 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	Основы архитектуры ПК	2		4	4
2	Переменные в Python	2		4	4
3	Однопроходные алгоритмы	2		4	6
4	Условный оператор и основы логики	2		4	4
5	Строки в Python	2		4	4
6	Списки и алгоритмы на списках	2		4	5
7	Множества и словари в Python	2		4	4
8	Функции в языке Python	2		4	4
9	Бисекция и сортировка списка	2		4	4
10	Рекурсия и динамическое программирование	2		4	6
Итого часов		20		40	45
Подготовка к экзамену		30 час.			
Общая трудоёмкость		135 час., 3 зач.ед.			

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

Семестр: 1 (Осенний)

##### 1. Основы архитектуры ПК

Основы архитектуры компьютера. Принципы фон Неймана.  
Операционная система. Место прикладных программ.  
Разделы жесткого диска. Файловая система.  
Виртуальные машины.  
Компиляция и интерпретация.  
Отличие интерпретируемых и компилируемых языков.  
Свободное программное обеспечение. 4 свободы свободного ПО.  
Свободные лицензии: GPL, MIT, BSD, Apache. Почему GPLv3 лучше

##### 2. Переменные в Python

Преимущества и недостатки языка Python 3  
Дзен Python. Antigravity  
Python2 и Python3  
Ресурсы для обучения Python: stepic.com, checkio.org, pythontutor.com  
Концепция присваивания в Python  
Переменные, значения и их типы. Понятие о динамической типизации.  
Обмен двух переменных значениями.  
Кортежи и их использование.  
Кортежи переменных. Обмен значений.  
Арифметические операции. Возведение в степень, деление нацело.  
«Hello, World!» на Python

##### 3. Однопроходные алгоритмы

Цикл while. Инструкции управления циклом.  
Позиционные системы счисления  
Литералы чисел в Python  
Разложение числа на цифры.  
Однопроходные алгоритмы: подсчёт, сумма, произведение.  
Среднее арифметическое.

#### 4. Условный оператор и основы логики

Оператор if. Каскадная условная конструкция elif.  
Логические операции в Python.  
Основы алгебры логики  
Однопроходные алгоритмы: поиск числа в потоке, максимум.  
Тест простоты числа.  
Разложение числа на множители.

#### 5. Строки в Python

ASCII и Unicode.  
Тип str. Длина строки len(s). Неизменяемость строки.  
Срезы строк.  
Методы строк find, count, replace, startswith, endswith.  
Наивный поиск подстроки в строке.  
Приведение строки к числу с указанием системы счисления.

#### 6. Списки и алгоритмы на списках

Тип list. Изменяемость списка.  
Ссылочная модель данных в Python. Операторы == и is. Копирование объектов.  
Алгоритм обращения массива.  
Алгоритм циклического сдвига в массиве.  
Срезы списков. Присваивание в срез. Методы списка.  
Стандартные функции len, max, min, sum.  
Список строк. Методы split и join для строки.  
Тип tuple как замороженный list.

#### 7. Множества и словари в Python

Тип set. Множества и работа с ними.  
Тип dict. Словарь (ассоциативный массив) и операции с ним.  
Dict comprehensions: генерация множеств и словарей.  
Частотный анализ для строк.  
Генераторы, yield.

#### 8. Функции в языке Python

Подключение модулей инструкцией import  
Модуль math  
Модуль random  
Запись арифметических выражений в выражения на Python.  
Создание функции в Python.  
Полиморфизм в Python. Duck typing.  
Значения параметров по умолчанию.  
Именованные параметры.

## 9. Бисекция и сортировка списка

## 10. Рекурсия и динамическое программирование

Рекурсия. Прямой и обратный ход рекурсии.

Факториал числа.

Вычисление чисел Фибоначчи.

Проблема алгоритмической сложности задачи.

Ханойские башни.

Генерация всех перестановок (рекурсивная)

Максимальная глубина рекурсии в Python

Одномерное динамическое программирование.

Двумерное динамическое программирование

Наибольшая общая подпоследовательность.

Наибольшая возрастающая подпоследовательность

Рекурсия с кешированием на примере факториала.

## 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

Дополнительная литература

1. Программирование на Python 3, подробное руководство/М. Саммерфилд,-СПб, Символ-Плюс, 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 Ejjudge 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 констестов, 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 констест завязан на материал прошедшей лекции, что делает посещение лекций насущной необходимостью в течение семестра.

Экзамен 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.

Дополнительная литература:

1. Програмируем на Python. Майкл Доусон. Издательство: Питер ISBN 978-5-459-00314-7, 978-1435455009; 2012 г.
2. Python. Карманный справочник. Марк Лутц. Издательство: Вильямс ISBN 978-5-8459-1965-6; 2014 г.

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)**

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

Семестр, формы промежуточной аттестации: 1 (осенний) - Экзамен

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

## 1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
ОПК-6 Способен применять математические, системно-аналитические, вычислительные методы и программные средства для решения прикладных задач в области создания систем анализа и автоматического управления и их компонентов	ОПК-6.3 Использует программные средства для разработки информационных систем

## 2. Показатели оценивания компетенций

В результате изучения дисциплины «Информатика» обучающийся должен:

### знать:

- основы архитектуры электронно-вычислительной машины (ЭВМ), представление информации в ЭВМ;
- общие характеристики интерпретируемых и компилируемых языков программирования;
- основные принципы устройства и работы операционной системы;
- основы алгоритмического языка программирования Python;
- приёмы разработки программ.

### уметь:

- использовать знания по информатике для приложений в инновационной, конструкторско-технологической и производственно-технологической сферах деятельности;
- разрабатывать небольшие программы на языке высокого уровня Python;
- использовать специализированные среды программирования для написания и отладки программ;
- выбирать адекватные алгоритмы для написания программ по обработке числовой и текстовой информации.

### владеть:

- навыками программирования для решения исследовательских задач на языке программирования Python;
- средствами отладки программ на Python;
- основами работы с 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. Основы архитектуры компьютера. Принципы фон Неймана.
2. Отличие интерпретируемых и компилируемых языков.
3. Концепция присваивания в Python
4. Обмен двух переменных значениями.
5. Кортежи и их использование.
6. Цикл while. Инструкции управления циклом.
7. Позиционные системы счисления
8. Однопроходные алгоритмы: подсчёт, сумма, произведение.
9. Оператор if. Каскадная условная конструкция elif.
10. Логические операции в Python.
11. Основы алгебры логики
12. Однопроходные алгоритмы: поиск числа в потоке, максимум.



13. Тест простоты числа.
14. Разложение числа на множители.
15. Тип str. Длина строки len(s). Неизменяемость строки.
16. Срезы строк.
17. Методы строк find, count, replace, startswith, endswith.
18. Наивный поиск подстроки в строке.
19. Ссылочная модель данных в Python. Операторы == и is. Копирование объектов.
20. Алгоритм обращения массива.
21. Алгоритм циклического сдвига в массиве.
22. Срезы списков. Присваивание в срез. Методы списка.
23. Список строк. Методы split и join для строки.
24. Цикл for и его особенности в Python.
25. Listcomprehensions: генерация списков.
26. Двумерные массивы (списки списков). Вложенная генерация.
27. Полиморфизм в Python. Ducktyping.
28. Именованные параметры.
29. Поиск корня функции методом бисекции.
30. Поиск значения в упорядоченном массиве методом бисекции.
31. Сортировка обезьяны.
32. Сортировка вставками.
33. Сортировка выбором.
34. Сортировка методом пузырька.
35. Сортировка дурака
36. Сортировка подсчётом.
37. Поразрядная сортировка.
38. Прагматическая сортировка TimSort.
39. Рекурсия. Прямой и обратный ход рекурсии.
40. Проблема алгоритмической сложности задачи.
41. Ханойские башни.
42. Генерация всех перестановок (рекурсивная)
43. Одномерное динамическое программирование.
44. Двумерное динамическое программирование
45. Рекурсия с кешированием на примере факториала.
46. Тип set. Множества и работа с ними.
47. Тип dict. Словарь (ассоциативный массив) и операции с ним.
48. Dict comprehensions: генерация множеств и словарей.
49. Частотный анализ для строк.
50. Генераторы, yield.

Пример билета:

1. Одномерное динамическое программирование.
2. Двумерное динамическое программирование

### Критерии оценивания

На устном экзамене преподаватель оценивает ответ студента в целом и выставляет оценку согласно приведённым ниже критериям и изложенным выше замечаниям касательно письменной части экзамена:

Оценка «отлично (10)» выставляется студенту, показавшему всесторонние систематизированные глубокие знания учебной программы и за её пределами, а также умение уверенно применять их на практике при решении сложных нестандартных задач.

Оценка «отлично (9)» выставляется студенту, показавшему всесторонние систематизированные глубокие знания учебной программы и умение уверенно применять их на практике при решении нестандартных задач.

Оценка «отлично (8)» выставляется студенту, показавшему всесторонние систематизированные глубокие знания учебной программы и умение уверенно применять их на практике при решении нестандартных задач, однако допустившему некоторые неточности при ответе.

Оценка «хорошо (7)» выставляется студенту, если он продемонстрировал твердое знание и уверенное понимание материала учебной программы и умение свободно применять физические законы на практике при решении типовых задач.

Оценка «хорошо (6)» выставляется студенту, если он продемонстрировал твердое знание материала учебной программы и умение применять физические законы на практике при решении типовых задач.

Оценка «хорошо (5)» выставляется студенту, если он продемонстрировал твердое знание и понимание материала учебной программы и умение применять физические законы на практике при решении типовых задач, однако допустил при ответе ряд грубых неточностей.

Оценка «удовлетворительно (4)» выставляется студенту, показавшему фрагментарный характер знаний, допускавшему неточности в формулировке основных законов и базовых понятий, но при этом продемонстрировавшему способность решать простые задачи и владение основными разделами учебной программы, необходимыми для дальнейшего обучения.

Оценка «удовлетворительно (3)» выставляется студенту, показавшему сильно фрагментарный характер знаний, допускавшему грубые ошибки в формулировке основных законов и базовых понятий, но при этом продемонстрировавшему способность решать простые задачи и владение основными разделами учебной программы, необходимыми для дальнейшего обучения.

Оценка «неудовлетворительно (2)» или «неудовлетворительно (1)» выставляется студенту, который не знает значительную часть основного содержания программы, систематически допускает грубые ошибки при формулировании основных физических законов или не способен корректно применять физические законы даже для решения простых задач.

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

Промежуточная аттестация по дисциплине «Информатика» осуществляется в форме экзамена. Экзамен принимается в устной форме с учётом оценки по контрольным 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.

### **3. Перечень типовых контрольных заданий, используемых для оценки знаний, умений, навыков**

Примерный перечень контрольных вопросов на экзамене:

1. Основы архитектуры компьютера. Принципы фон Неймана.
2. Отличие интерпретируемых и компилируемых языков.
3. Концепция присваивания в Python
4. Обмен двух переменных значениями.
5. Кортежи и их использование.
6. Цикл while. Инструкции управления циклом.
7. Позиционные системы счисления
8. Однопроходные алгоритмы: подсчёт, сумма, произведение.
9. Оператор if. Каскадная условная конструкция elif.
10. Логические операции в Python.
11. Основы алгебры логики
12. Однопроходные алгоритмы: поиск числа в потоке, максимум.
13. Тест простоты числа.
14. Разложение числа на множители.
15. Тип str. Длина строки len(s). Неизменяемость строки.
16. Срезы строк.
17. Методы строк find, count, replace, startswith, endswith.
18. Наивный поиск подстроки в строке.
19. Ссылочная модель данных в Python. Операторы == и is. Копирование объектов.
20. Алгоритм обращения массива.
21. Алгоритм циклического сдвига в массиве.
22. Срезы списков. Присваивание в срез. Методы списка.
23. Список строк. Методы split и join для строки.
24. Цикл for и его особенности в Python.
25. Listcomprehensions: генерация списков.
26. Двумерные массивы (списки списков). Вложенная генерация.
27. Полиморфизм в Python. Ducktyping.
28. Именованные параметры.
29. Поиск корня функции методом бисекции.
30. Поиск значения в упорядоченном массиве методом бисекции.
31. Сортировка обезьяны.
32. Сортировка вставками.
33. Сортировка выбором.
34. Сортировка методом пузырька.
35. Сортировка дурака
36. Сортировка подсчётом.
37. Поразрядная сортировка.
38. Прагматическая сортировка TimSort.
39. Рекурсия. Прямой и обратный ход рекурсии.
40. Проблема алгоритмической сложности задачи.
41. Ханойские башни.
42. Генерация всех перестановок (рекурсивная)
43. Одномерное динамическое программирование.
44. Двумерное динамическое программирование
45. Рекурсия с кэшированием на примере факториала.
46. Тип set. Множества и работа с ними.
47. Тип dict. Словарь (ассоциативный массив) и операции с ним.
48. Dict comprehensions: генерация множеств и словарей.
49. Частотный анализ для строк.

## 50. Генераторы, yield.

На экзамене 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. Сортировка обезьяны.
2. Рекурсия. Прямой и обратный ход рекурсии.
3. Тип set. Множества и работа с ними.
4. Задача: реализовать слияние двух отсортированных списков.

## 4. Критерии оценивания

Промежуточная аттестация по дисциплине «Информатика» осуществляется в форме экзамена.

Экзамен принимается в устной форме с учётом оценки по контрольным 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.

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 констестов выставляется автоматически исходя из суммарного рейтинга обучающегося в системе Ejudgeи также нормируется к десятибалльной шкале.

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

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

Время проведения экзамена составляет 30 минут на одного обучающегося.

Во время подготовки к ответу обучающиеся не могут пользоваться литературой, печатными материалами, рукописными записями, а также электронными средствами (сотовыми телефонами, планшетами, умными часами и т.п.).