

**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>	Программирование на языке Python
<b>по направлению:</b>	Прикладные математика и физика
<b>профиль подготовки:</b>	Системная и синтетическая биология Физтех-школа Биологической и Медицинской Физики кафедра информатики и вычислительной математики
<b>курс:</b>	3
<b>квалификация:</b>	бакалавр

Семестр, формы промежуточной аттестации: 6 (весенний) - Дифференцированный зачет

Аудиторных часов: 60 всего, в том числе:

лекции: 0 час.

семинары: 0 час.

лабораторные занятия: 60 час.

Самостоятельная работа: 30 час.

Всего часов: 90, всего зач. ед.: 2

Количество контрольных работ, заданий: 4

Программу составили:

Т.Ф. Хирьянов, старший преподаватель

М.Н. Герцев, канд. физ.-мат. наук

Программа обсуждена на заседании кафедры информатики и вычислительной математики 06.02.2020

## Аннотация

Курс направлен на получение практических навыков совместной разработки профессиональных программных продуктов средствами языка Python 3. Студенты знакомятся с основами объектно-ориентированного программирования и конструкциями языка Python: декоратор, генератор, классы. Студенты получают навыки в создании многопоточных и сетевых программ, познакомятся с обширным набором модулей Python, научатся кооперативно создавать сложные проекты, используя для совместной работы git репозиторий.

### 1. Цели и задачи

#### Цель дисциплины

- освоение студентами знаний в области разработки современных приложений. Осмысленное применение полученных знаний при изучении других дисциплин.

#### Задачи дисциплины

- формирование понимания процессов, происходящих в вычислительной системе при запуске и работе программ и программных систем, принципов корректной передачи информации между ними и их взаимной синхронизации;
- обучение студентов методам создания корректно работающих и взаимодействующих программ с помощью системных вызовов операционных систем;
- формирование способности эффективно использовать современные вычислительные системы при изучении других дисциплин и при выполнении исследований студентами в рамках выпускных работ на степень бакалавра.

### 2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
УК-3 Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде	УК-3.2 Взаимодействует с другими членами команды для достижения поставленной задачи
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

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

В результате освоения дисциплины обучающиеся должны знать:

- историю эволюции вычислительных систем, основные функции, выполняемые современными операционными системами, принципы их внутреннего построения;
- концепцию процессов в операционных системах;
- основные алгоритмы планирования процессов;
- логические основы взаимодействия процессов;
- концепцию нитей исполнения и их отличие от обычных процессов;
- программные алгоритмы организации взаимодействия процессов и предъявляемые к ним требования;
- основные механизмы синхронизации в операционных системах;
- организацию управления оперативной памятью использующиеся при этом алгоритмы;
- основные принципы управления файловыми системами;
- организацию управления устройствами ввода-вывода на уровне как технического, так и программного обеспечения, основные функции подсистемы ввода-вывода;
- принципы сетевого взаимодействия вычислительных систем и построения работы сетевых частей операционных систем;
- основные проблемы безопасности операционных систем и подходы к их решению.
- идеологию объектно-ориентированного подхода;
- принципы программирования структур данных для современных программ;
- типовые решения, применяемые для создания программ.

уметь:

- пользоваться командами командного интерпретатора операционной системы Linux;
- порождать новые процессы, запускать новые программы и правильно завершать их функционирование;
- порождать новые нити исполнения и правильно завершать их функционирование;
- организовывать взаимодействие процессов через потоковые средства связи, разделяемую память и очереди сообщений;
- использовать семафоры и сигналы для синхронизации работы процессов и нитей исполнения;
- использовать системные вызовы для работы с файловой системой;
- разрабатывать программы для сетевого взаимодействия.
- применять объектно-ориентированный подход для написания программ;
- создавать безопасные программы;
- использовать современные средства для написания и отладки программ.

владеть:

- навыками использования команд командного интерпретатора в операционной системе Linux;
- навыками написания и отладки программ, порождающих несколько процессов или нитей исполнения;
- навыками написания и отладки программ, использующих системные вызовы для взаимодействия локальных процессов;
- навыками написания и отладки программ, использующих системные вызовы для работы с файловыми системами и устройствами ввода-вывода;
- навыками написания и отладки сетевых приложений.
- объектно-ориентированным языком программирования (Python 3);
- средствами использования стандартных библиотек.

#### 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	Повторение материала			4	1
2	Работа в командной строке Linux			4	2
3	Работа с файлами			4	1
4	Аргументы командной строки			4	2
5	Декораторы			4	2
6	Объектно-ориентированное программирование			12	8
7	Многопоточность в Python			4	2
8	Генераторы			4	2
9	Итераторы и сопроцессы			4	2
10	Асинхронное программирование			4	2
11	Регулярные выражения			4	2
12	Библиотека pickle			4	2
13	Git			4	2
Итого часов				60	30
Подготовка к экзамену		0 час.			
Общая трудоёмкость		90 час., 2 зач.ед.			

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

### Семестр: 6 (Весенний)

#### 1. Повторение материала

Цели и задачи курса. Повторение синтаксиса языка Python: цикл for; цикл while; ветвление с использованием if ... elif ... else; операторы break и continue; работа со списками (list), словарями (dict), строками, кортежами; асимптотика операций; функции; параметры по умолчанию; функции с неизвестным количеством параметров; работа со строками. Обработка исключений.

#### 2. Работа в командной строке Linux

Короткий обзор команд и программ консоли Linux: ls, cd, mkdir, cat, head, tail, vim, tmux, jobs, fg, bg, ps, kill,

#### 3. Работа с файлами

Операции чтения из файла, запись в файл, проверка существования файла и пр. средствами Python. Форматы csv, json.

#### 4. Аргументы командной строки

Библиотеки sys и argparse

#### 5. Декораторы

Синтаксис написания декораторов. Их применение.

#### 6. Объектно-ориентированное программирование

Парадигмы ООП: инкапсуляция, наследование, полиморфизм. SOLID принципы. Классы. Статические и классовые методы. Магические методы класса. Абстрактные классы. Декораторы @staticmethod, @abstractmethod, @property. Декомпозиция программы на классы.

#### 7. Многопоточность в Python

Библиотеки thread, multiprocessing. Особенности использования нитей, GIL. Обмен информацией между потоками, Pipe, Value, Queue.

#### 8. Генераторы

Понятие генератора. Оператор yield. Функция next. Примеры генераторов, написание генераторов range, zip, map, enumerate/

#### 9. Итераторы и сопроцессы

Магические методы класса \_\_iter\_\_, \_\_next\_\_. Примеры использования. Сопроцессы. Метод seed().

#### 10. Асинхронное программирование

Понятия асинхронности параллелизма и конкурентности. Библиотека asyncio

#### 11. Регулярные выражения

Библиотека re. Практическое использование методов re.match, re.search, re.findall, re.split, re.sub, re.compile.

## 12. Библиотека pickle

Использование библиотеки pickle для сериализации и десериализации данных.

## 13. Git

Системы контроля версий. Использование команд git init; git config; git add; git commit; git push; git pull; git branch; git checkout. Использование github.com задачи, цели, нити.

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

Учебная аудитория, оснащенная мультимедиа проектором и экраном для чтения лекций.  
Учебный сетевой компьютерный класс с установленной операционной системой Linux, Python 3.9 с библиотеками: numpy, matplotlib, sklearn, flask, pandas, socket, sqlite3, pyqt5.

## 6. Перечень рекомендуемой литературы

### Основная литература

1. Python на практике [Текст], создание качественных программ с использованием параллелизма, библиотек и паттернов/М. Саммерфилд, -М, ДМК Пресс, 2014

### Дополнительная литература

1. Python и машинное обучение [Текст], крайне необходимое издание по новейшей предсказательной аналитике для более глубокого понимания методологии машинного обучения/С. Рашка, -М., ДМК Пресс, 2017
2. СУБД : Язык SQL в примерах и задачах [Текст] : учеб. пособие для вузов / И. Ф. Астахова [и др.] .— М. : Физматлит, 2009 .— 168 с.

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

1. <http://www.intuit.ru>,
2. <http://cs.mipt.ru>

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

На лабораторных занятиях используются мультимедийные технологии, включая демонстрацию презентаций. На компьютерах в компьютерных классах должна быть установлены операционная система Linux, python3.6 с библиотеками socket, pandas, numpy, matplotlib, sklearn sqlite3.

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

Студент, изучающий курс, должен, с одной стороны, овладеть общим понятийным аппаратом, а с другой стороны, должен научиться применять теоретические знания на практике.

В результате изучения дисциплины студент должен знать основные определения, понятия, алгоритмы, уметь писать многопроцессные и многопоточные приложения в среде операционной системы Linux, корректно организовывать взаимодействие процессов и нитей, как локальных, так и удаленных, работать с файлами и устройствами ввода-вывода.

Успешное освоение курса требует напряжённой работы студента. В программе курса приведено минимально необходимое время для работы студента над темой. Самостоятельная работа включает в себя:

- чтение и конспектирование рекомендованной литературы;
- проработку учебного материала, доказательство отдельных утверждений, свойств;
- решение задач, предлагаемых студентам на лекциях и лабораторных работах;
- подготовку к лабораторным работам;
- решение заданий.

Руководство и контроль за самостоятельной работой студента осуществляется в форме индивидуальных консультаций.

Показателем владения материалом служит умение решать теоретические и практические задачи. Для формирования умения применять теоретические знания на практике студенту необходимо решать как можно больше практических задач. При решении задач каждое действие необходимо аргументировать, ссылаясь на известные теоретические сведения. Программы должны легко читаться и иметь подробные комментарии.

Важно добиться понимания изучаемого материала, а не механического его запоминания. При затруднении изучения отдельных тем, вопросов, следует обращаться за консультациями к лектору или преподавателю, проводящему лабораторные работы.

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)**

**по направлению:** Прикладные математика и физика  
**профиль подготовки:** Системная и синтетическая биология  
Физтех-школа Биологической и Медицинской Физики  
кафедра информатики и вычислительной математики  
**курс:** 3  
**квалификация:** бакалавр

Семестр, формы промежуточной аттестации: 6 (весенний) - Дифференцированный зачет

**Разработчики:**

Т.Ф. Хирьянов, старший преподаватель  
М.Н. Герцев, канд. физ.-мат. наук

## 1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
УК-3 Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде	УК-3.2 Взаимодействует с другими членами команды для достижения поставленной задачи
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

## 2. Показатели оценивания компетенций

В результате изучения дисциплины «Программирование на языке Python» обучающийся должен:

### знать:

- историю эволюции вычислительных систем, основные функции, выполняемые современными операционными системами, принципы их внутреннего построения;
- концепцию процессов в операционных системах;
- основные алгоритмы планирования процессов;
- логические основы взаимодействия процессов;
- концепцию нитей исполнения и их отличие от обычных процессов;
- программные алгоритмы организации взаимодействия процессов и предъявляемые к ним требования;
- основные механизмы синхронизации в операционных системах;
- организацию управления оперативной памятью использующиеся при этом алгоритмы;
- основные принципы управления файловыми системами;
- организацию управления устройствами ввода-вывода на уровне как технического, так и программного обеспечения, основные функции подсистемы ввода-вывода;
- принципы сетевого взаимодействия вычислительных систем и построения работы сетевых частей операционных систем;
- основные проблемы безопасности операционных систем и подходы к их решению.
- идеологию объектно-ориентированного подхода;
- принципы программирования структур данных для современных программ;
- типовые решения, применяемые для создания программ.

### уметь:

- пользоваться командами командного интерпретатора операционной системы Linux;
- порождать новые процессы, запускать новые программы и правильно завершать их функционирование;
- порождать новые нити исполнения и правильно завершать их функционирование;
- организовывать взаимодействие процессов через потоковые средства связи, разделяемую память и очереди сообщений;
- использовать семафоры и сигналы для синхронизации работы процессов и нитей исполнения;
- использовать системные вызовы для работы с файловой системой;
- разрабатывать программы для сетевого взаимодействия.
- применять объектно-ориентированный подход для написания программ;
- создавать безопасные программы;
- использовать современные средства для написания и отладки программ.

### владеть:

- навыками использования команд командного интерпретатора в операционной системе Linux;
- навыками написания и отладки программ, порождающих несколько процессов или нитей исполнения;
- навыками написания и отладки программ, использующих системные вызовы для взаимодействия локальных процессов;
- навыками написания и отладки программ, использующих системные вызовы для работы с файловыми системами и устройствами ввода-вывода;
- навыками написания и отладки сетевых приложений.
- объектно-ориентированным языком программирования (Python 3);
- средствами использования стандартных библиотек.



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

Примеры типовых задач для текущего контроля включают задачи создания различной сложности классов с использованием магических методов и итераторов:

1. Класс трехмерных векторами.
2. Класс квадратных матриц.
3. Класс обработки текстовой информации с итератором по словам по частоте встречаемости в тексте.
4. Класс для осуществления классификатора набора числовых данных.
5. Очередь без использования list.
6. Стек без использования list.
7. Именованный массив, без использования dict.
8. Создать иеррархию классов, включающую параллелограмм, квадрат и ромб.
9. Класс двоичных деревьев поиска.
10. Класс heap.

От всех студентов требуется умение создавать стандартные генераторы map, range, zip, enumerate и декораторы.

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

1. Ссылочная модель данных, параметры функция по умолчанию, функции с неизвестным числом параметров.
2. Парадигмы ООП, SOLID принципы.
3. Создание класса в Python, инстанцирование, декоратор @staticmethod.
4. Магические методы классов.
5. Абстрактный класс, библиотека abc.
6. Генераторы, оператор yield.
7. Итераторы в Python.
8. Сопроцессы.
9. Асинхронное программирование, asyncio.
10. Регулярные выражения, библиотека re.
11. Библиотека threading, GIL.
12. Библиотека subprocess.
13. Обмен данными между процессами, pipe.
14. Работа с фалами Python.
15. Декораторы в Python.
16. Сериализация и десериализация с использованием pickle

#### Критерии оценивания

Промежуточная аттестация по дисциплине «Объектно-ориентированное программирование» осуществляется в форме дифференцированного зачета.

Дифференцированный зачёт принимается в устной форме с учётом 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.

Для положительной оценки (больше 3 баллов) необходимо реализовать рабочую программу на языке Python с использованием классов, генераторов и декораторов. Точная оценка ставится по результатам устного опроса по созданной программе.

- 10 - Обучающийся ответил на все вопросы, но не с первой попытки.
- 9 - Обучающийся допустил не более одной ошибки или воспользовался помощью преподавателя.
- 8 - Обучающийся, работая самостоятельно, допустил не более двух численных ошибок в лабораторной работе.
- 7 - Обучающийся если он ответил на подавляющее большинство вопросов в лабораторной работе, может быть с помощью преподавателя или товарищей.
- 6 - Обучающийся ответил на подавляющее большинство вопросов в лабораторной работе, может быть с помощью преподавателя или товарищей.
- 5 - Обучающийся ответил на основные вопросы в лабораторной работе, может быть с помощью преподавателя или товарищей.
- 4 - Обучающийся ответил на основные вопросы в лабораторной работе
- 3 - Обучающийся ответил на некоторые вопросы в лабораторной работе.
- 2 - Обучающийся не справился с работой.
- 1 - Обучающийся демонстрирует полное отсутствие знаний по предмету или пытался выдать чужую работу за свою.

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

Время проведения дифференцированного зачёта 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.).