

УТВЕРЖДАЮ  
 Декан факультета

\_\_\_\_\_  
 (подпись) Суслин А. В.  
 ФИО  
 «\_\_\_» \_\_\_\_\_ 20\_\_

## РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ

Направление/специальность подготовки	15.04.03 Прикладная механика
Специализация/профиль/программа подготовки	Механика процессов обработки давлением
Уровень высшего образования	Магистратура
Форма обучения	Очная
Факультет	Е Оружие и системы вооружения
Выпускающая кафедра	Е4 ВЫСОКОЭНЕРГЕТИЧЕСКИЕ УСТРОЙСТВА АВТОМАТИЧЕСКИХ СИСТЕМ
Кафедра-разработчик рабочей программы	Е4 ВЫСОКОЭНЕРГЕТИЧЕСКИЕ УСТРОЙСТВА АВТОМАТИЧЕСКИХ СИСТЕМ

КУРС	СЕМЕСТР	ОБЩАЯ ТРУДОЁМКОСТЬ (ЗАЧЕТНЫХ ЕДИНИЦ)	ЧАСЫ (по наличию видов занятий)									ВИД ПРОМЕЖУТОЧНОГО КОНТРОЛЯ
			ОБЩАЯ ТРУДОЁМКОСТЬ	АУДИТОРНЫЕ ЗАНЯТИЯ				САМОСТОЯТЕЛЬНАЯ РАБОТА				
				ВСЕГО	ЛЕКЦИИ	ЛАБОРАТОРНЫЙ ПРАКТИКУМ	ПРАКТИЧЕСКИЕ ЗАНЯТИЯ	ВСЕГО	КУРСОВОЙ ПРОЕКТ	КУРСОВАЯ РАБОТА	ДРУГИЕ ВИДЫ САМОСТ. РАБОТЫ	
5	10	3	108	34	0	0	34	74	0	0	74	зач.

*ЛИСТ СОГЛАСОВАНИЯ*

**РАБОЧАЯ ПРОГРАММА СОСТАВЛЕНА В СООТВЕТСТВИИ С ТРЕБОВАНИЯМИ ФЕДЕРАЛЬНОГО  
ГОСУДАРСТВЕННОГО ОБРАЗОВАТЕЛЬНОГО СТАНДАРТА ВЫСШЕГО ОБРАЗОВАНИЯ (ФГОС ВО)**

**15.04.03 Прикладная механика**

год набора группы: 2024

Программу составил:

Кафедра Е4 **ВЫСОКОЭНЕРГЕТИЧЕСКИЕ УСТРОЙСТВА  
АВТОМАТИЧЕСКИХ СИСТЕМ**

Сидоренко Тимофей Владимирович, старший преподаватель

Программа рассмотрена

на заседании кафедры-разработчика

рабочей программы **Е4 ВЫСОКОЭНЕРГЕТИЧЕСКИЕ УСТРОЙСТВА АВТОМАТИЧЕСКИХ  
СИСТЕМ**

Заведующий кафедрой Нестеров Н.И., к.т.н., доц.

Программа рассмотрена

на заседании выпускающей кафедры

**Е4 ВЫСОКОЭНЕРГЕТИЧЕСКИЕ УСТРОЙСТВА АВТОМАТИЧЕСКИХ СИСТЕМ**

Заведующий кафедрой Нестеров Н.И., к.т.н., доц.

# **РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ**

## **Разделы рабочей программы**

1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ
2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП ВО
3. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ
4. ФОРМЫ КОНТРОЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ
5. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ
6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

## **Приложения к рабочей программе дисциплины**

- Приложение 1. Аннотация рабочей программы
- Приложение 2. Технологии и формы обучения
- Приложение 3. Фонды оценочных средств

## 1. ЦЕЛИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Целью освоения дисциплины является формирование следующих компетенций:

ОПК-12 — способность создавать алгоритмы цифровой обработки баз данных результатов испытаний и эксплуатации сложных деталей и узлов в машиностроении, разрабатывать современные цифровые программы расчетов и проектирования деталей, узлов, конструкций, машин и материалов с учетом требований надежности, долговечности и безопасности их эксплуатации

Формированию компетенций служит достижение следующих результатов образования:

### **ОПК-12**

*знания:*

основные алгоритмические конструкции;

базовые алгоритмы обработки данных;

базовый синтаксис и основные структуры данных языка программирования Python;

представление о технологии программирования, основных понятиях и подходах при разработке программного обеспечения;

основные понятия объектно-ориентированного программирования;

*умения:*

оценка вычислительной сложности разрабатываемых алгоритмов;

разработка моделей информационных систем на основе парадигмы объектно-ориентированного программирования;

применение типовых алгоритмов и структур данных для решения прикладных задач;

разработка прикладных программ для инженерных и научных вычислений (автоматизация обработки наборов данных, визуализация данных, применение численных методов) на языке программирования Python с использованием стандартных и специализированных библиотек;

*навыки:*

чтение и понимание исходных кодов программ на языке Python;

разработка программ на языке программирования Python по заданным алгоритмическим схемам;

разработка алгоритмов для решения простых прикладных задач.

## 2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ООП ВО

Дисциплина **АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ** является дисциплиной **обязательной части блока 1** программы подготовки по направлению *15.04.03 Прикладная механика*.

Содержание дисциплины является логическим продолжением дисциплин: **ВЫСШАЯ МАТЕМАТИКА В НАУЧНЫХ ИССЛЕДОВАНИЯХ**.

Содержание дисциплины является основой для освоения дисциплин: **МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПРОЦЕССОВ ОБРАБОТКИ МЕТАЛЛОВ ДАВЛЕНИЕМ, НАУЧНО-ИССЛЕДОВАТЕЛЬСКАЯ РАБОТА**.

Предварительные компетенции, сформированные у обучающегося до начала изучения дисциплины:

- ОПК-5 — Способен разрабатывать аналитические и численные методы при создании математических моделей машин, приводов, оборудования, систем, технологических процессов
- УК-1 — Способен осуществлять критический анализ проблемных ситуаций на основе системного подхода, вырабатывать стратегию действий

### 3. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

Общая трудоемкость дисциплины составляет 3 з.е., 108 ч.

#### 3.1. Содержание (дидактика) дисциплины

КУРС	СЕМЕСТР	Наименование разделов и дидактических единиц	ВСЕГО	Аудиторные занятия в контактной форме		Самостоятельная работа студентов	Формируемая компетенция, %
				ВСЕГО	Практические занятия		ОПК-12
5	10	<b>Раздел 1. Технология программирования. Основы алгоритмизации.</b> Технология программирования, основные понятия и подходы. Понятие алгоритма, его свойства, формы представления. Язык программирования Python.	6	2	2	4	10
5	10	<b>Раздел 2. Базовый синтаксис Python.</b> Введение в процедурное программирование. Типы данных. Типы коллекций. Управляющие структуры и функции. Модули. Работа с файлами.	22	8	8	14	25
5	10	<b>Раздел 3. Основные алгоритмические конструкции. Вычислительная сложность.</b> Основные алгоритмические конструкции. Алгоритмы обработки массивов, сортировки и поиска. Вычислительная сложность.	18	8	8	10	15
5	10	<b>Раздел 4. Основы объектно-ориентированного программирования.</b> Основные понятия объектно-ориентированного программирования. Объекты, классы, методы, атрибуты. Инкапсуляция, наследование и полиморфизм.	12	4	4	8	15
5	10	<b>Раздел 5. Стандартные и специализированные библиотеки Python.</b> Стандартные библиотеки Python (math, time, sys, os). Использование интерактивного блокнота Jupyter. Применение библиотеки NumPy для работы с массивами данных. Применение библиотеки Pandas для обработки и анализа данных. Применение библиотеки Matplotlib для визуализации данных. Применение библиотеки SciPy для специализированных инженерных и научных расчётов.	30	10	10	20	20
5	10	<b>Раздел 6. Создание проекта.</b> Создание проекта. Модульность. Системы контроля версий. GIT. Создание графического интерфейса.	20	2	2	18	15
<b>Всего за 10 семестр</b>			108	34	34	74	100
<b>Всего по дисциплине</b>			108	34	34	74	100

#### 3.2. Аудиторный практикум

№ п/п	Номер и наименование раздела дисциплины	Тема практического занятия	Объем, ауд. часов
1	Раздел 1. Технология программирования. Основы алгоритмизации.	Основные концепции и понятия технологии программирования. Интегрированные среды разработки (IDE). Установка и настройка окружения.	2
2	Раздел 2. Базовый синтаксис Python.	Введение в процедурное программирование. Использование интерактивного блокнота Jupyter.	2
3		Управляющие структуры и функции. Модули.	2
4		Работа с файлами.	2
5		Типы данных. Типы коллекций. Модули.	2
6	Раздел 3. Основные алгоритмические конструкции. Вычислительная сложность.	Разбор основных алгоритмических конструкций.	2
7		Алгоритмы обработки массивов, сортировки и поиска.	2
8		Оценка вычислительной сложности.	2
9		Обработка исключений.	2
10	Раздел 4. Основы объектно-ориентированного программирования.	Основные понятия и принципы объектно-ориентированного программирования.	2
11		Разбор примеров применения объектно-ориентированного программирования.	2
12	Раздел 5. Стандартные и специализированные библиотеки Python.	Стандартные библиотеки Python (math, time, sys, os). Использование интерактивного блокнота Jupyter.	2
13		Применение библиотеки Matplotlib для визуализации данных.	2
14		Применение библиотеки NumPy для работы с массивами данных.	2
15		Применение библиотеки Pandas для обработки и анализа данных.	2
16		Применение библиотеки SciPy для специализированных инженерных и научных расчётов.	2
17	Раздел 6. Создание проекта.	Создание простого проекта.	2

Всего за 10 семестр														
														34

### 3.3. Самостоятельная работа студента (СРС)

№ п/п	Номер и наименование раздела дисциплины	Содержание учебного задания	Объем, часов
1	Раздел 1. Технология программирования. Основы алгоритмизации.	Ознакомление с особенностями наиболее популярных языков программирования. Самостоятельная установка IDE и настройка окружения.	4
2	Раздел 2. Базовый синтаксис Python.	Написание программ использующих базовые структуры данных языка программирования Python с применением простых управляющих структур.	14
3	Раздел 3. Основные алгоритмические конструкции. Вычислительная сложность.	Выполнение домашнего задания на разработку алгоритмов.	2
4		Выполнение задания на обработку массива данных.	3
5		Выполнение задания на сортировку и поиск.	3
6		Выполнение задания на оценку вычислительной сложности.	2
7	Раздел 4. Основы объектно-ориентированного программирования.	Выполнения домашних заданий по реализации объектно-ориентированных моделей в языке программирования Python.	8
8	Раздел 5. Стандартные и специализированные библиотеки Python.	Выполнение домашнего задания с применением стандартных библиотек Python.	4
9		Выполнение домашнего задания с применением библиотеки NumPy.	4
10		Выполнение домашнего задания с применением библиотеки Pandas.	4
11		Выполнение домашнего задания с применением библиотеки Matplotlib.	4
12		Выполнение домашнего задания с применением библиотеки SciPy.	4
13	Раздел 6. Создание проекта.	Выполнение индивидуального практического задания.	18
Всего за 10 семестр			74

## 4. ФОРМЫ КОНТРОЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

СЕМЕСТР	НЕДЕЛИ СЕМЕСТРА																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
10						ДР				ДР				ДЗ, ИПЗ		ДР	зач.

Условные обозначения:

- ДР – диагностическая работа;
- ДЗ – домашнее задание;
- ИПЗ – индивидуальное практическое задание;
- зач. – зачет.

**Текущий контроль успеваемости** студентов проводится в дискретные временные интервалы в следующих формах:

- диагностическая работа;
- домашнее задание;
- индивидуальное практическое задание.

**Промежуточная аттестация** проводится в формах:

- зачет.

## 5. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

### 5.1. Основная литература по дисциплине:

1. А. А. Андрианова, Л. Н. Исмагилов, Т. М. Мухтарова. . Алгоритмизация и программирование. Практикум. Санкт-Петербург: Лань, 2022, эл. рес.
2. А. Н. Гуцин, Т. И. Лазарева, И. В. Мартынова. . Типовые алгоритмы и их программирование. СПб.БГТУ "ВОЕНМЕХ" им. Д. Ф. Устинова, 2016, 450 экз.
3. А. Н. Гуцин, Т. И. Лазарева, И. В. Мартынова. . Типовые алгоритмы и их программирование. СПб.БГТУ "ВОЕНМЕХ" им. Д. Ф. Устинова, 2016, эл. рес.
4. Г. С. Иванова. . Технология программирования. М.: Изд-во МГТУ им. Н. Э. Баумана, 2006, эл. рес.
5. Д. Ю. Фёдоров. . Программирование на языке высокого уровня Python. Москва: Юрайт, 2023, эл. рес.
6. Дж. Кью, М. Джеанини. Объектно-ориентированное программирование. М.: Питер, 2005, 30 экз.
7. П. Дж. Дейтел, Х. М. Дейтел. . Python: Искусственный интеллект, большие данные и облачные вычисления. Санкт-Петербург: Питер, 2021, эл. рес.
8. С. З. Свердлов. . Языки программирования и методы трансляции. СПб.: Лань, 2019, 25 экз.
9. Х. М. Дейтел, П. Дж. Дейтел. . Как программировать на C++. БГТУ "ВОЕНМЕХ" им. Д. Ф. Устинова, 2006, эл. рес.

### 5.2. Дополнительная литература по дисциплине:

1. Т. Кормен, Ч. Лейзерсон, Р. Ривест. . Алгоритмы: построение и анализ. М.: МЦНМО, 2000, 0 экз.

### 5.3. Периодические издания:

не требуются.

### 5.4. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины, электронные библиотечные системы:

1. <https://www.altlinux.org/Books:PythonSchool> — Books:PythonSchool — ALT Linux Wiki;
2. <https://www.yuripetrov.ru/edu/python/index.html> — Программирование на языке высокого уровня (Python) &mdash; Kypc Python (2022);
3. <https://matplotlib.org> — Matplotlib &#8212; Visualization with Python;
4. <https://docs.python.org> — 3.10.6 Documentation;
5. <https://docs.scipy.org> — Numpy and Scipy Documentation &mdash; Numpy and Scipy documentation;
6. <https://numpy.org> — NumPy;
7. <https://pandas.pydata.org> — pandas - Python Data Analysis Library;
8. <https://docs.python.org/3/library/tkinter.html> — tkinter — Python interface to Tcl/Tk &#8212; Python 3.10.6 documentation;
9. [https://ru.wikibooks.org/wiki/GUI\\_Help/Tkinter\\_book](https://ru.wikibooks.org/wiki/GUI_Help/Tkinter_book) — GUI Help/Tkinter book — Викиучебник;
10. <https://git-scm.com/book/ru/v2> — Git - Book.

### Современные профессиональные базы данных:

1. <https://rusneb.ru> – Национальная электронная библиотека (НЭБ);
2. <https://cyberleninka.ru/> - Научная электронная библиотека «Киберленинка»;  
<http://www.rfbr.ru/rffi/ru/library> - Полнотекстовая электронная библиотека Российского фонда фундаментальных исследований.

### Информационные справочные системы:

1. Техэксперт – Информационный портал технического регулирования: Нормы, правила, стандарты РФ;
2. [http://library.voenmeh.ru/jirbis2/index.php?option=com\\_irbis&view=irbis&Itemid=457](http://library.voenmeh.ru/jirbis2/index.php?option=com_irbis&view=irbis&Itemid=457) - БД ГОСТов собственной генерации БГТУ "ВОЕНМЕХ" им. Д. Ф. Устинова;
3. <http://www.consultant.ru/>- КонсультантПлюс- информационный портал правовой информации.

### 5.5. Программное обеспечение:

1. Python 3.4;



2. Spyder;
3. Набор средств трансляции, компоновки, отладки и выполнения Python 3.x с интегрированной средой разработки IDLE.

#### 5.6. Информационные технологии:

взаимодействие с обучающимися посредством ЭИОС Moodle БГТУ «ВОЕНМЕХ» им. Д.Ф. Устинова.

## **6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ**

### **6.1. Практические занятия:**

1. Python 3.4;
2. Spyder;
3. Набор средств трансляции, компоновки, отладки и выполнения Python 3.x с интегрированной средой разработки IDLE.

### **6.2. Прочее:**

1. рабочее место преподавателя, оснащенное компьютером с доступом в Интернет;
2. рабочие места студентов, оснащенные компьютерами с доступом в Интернет, предназначенные для работы в электронной образовательной среде.

### Аннотация рабочей программы

Дисциплина **АЛГОРИТМИЗАЦИЯ И ПРОГРАММИРОВАНИЕ** является дисциплиной **обязательной части блока 1** программы подготовки по направлению *15.04.03 Прикладная механика*. Дисциплина реализуется на факультете *Е Оружие и системы вооружения* БГТУ "ВОЕНМЕХ" им. Д.Ф. Устинова кафедрой *Е4 ВЫСОКОЭНЕРГЕТИЧЕСКИЕ УСТРОЙСТВА АВТОМАТИЧЕСКИХ СИСТЕМ*.

Дисциплина нацелена на формирование *компетенций*:

ОПК-12 способность создавать алгоритмы цифровой обработки баз данных результатов испытаний и эксплуатации сложных деталей и узлов в машиностроении, разрабатывать современные цифровые программы расчетов и проектирования деталей, узлов, конструкций, машин и материалов с учетом требований надежности, долговечности и безопасности их эксплуатации.

Содержание дисциплины охватывает круг вопросов, связанных с формированием основных представлений об алгоритмизации и программировании, а именно: разработке алгоритмов и оценки их вычислительной сложности, основных парадигм и концепций разработки программного обеспечения, основами объектно-ориентированном программировании. Формирует базовые навыки написания прикладных программ на языке программирования Python, даёт представление об основных возможностях стандартных и специализированных библиотек (NumPy, Pandas, Matplotlib, SciPy), используемых для инженерных и научных вычислений.

Программой дисциплины предусмотрены следующие **виды контроля**:

**Текущий контроль успеваемости** студентов проводится в дискретные временные интервалы в следующих формах:

- диагностическая работа;
- домашнее задание;
- индивидуальное практическое задание.

**Промежуточная аттестация** проводится в формах:

- зачет.

Общая трудоемкость освоения дисциплины составляет **3 з.е., 108 ч**. Программой дисциплины предусмотрены практические занятия (**34 ч.**), самостоятельная работа студента (**74 ч.**).

## ТЕХНОЛОГИИ И ФОРМЫ ОБУЧЕНИЯ

### Рекомендации по освоению дисциплины для студента

Трудоемкость освоения дисциплины составляет 108 ч., из них 34 ч. аудиторных занятий, и 74 ч., отведенных на самостоятельную работу студента.

Рекомендации по распределению учебного времени по видам самостоятельной работы и разделам дисциплины приведены в таблице.

Контроль освоения дисциплины производится в соответствии с Положением о текущем, рубежном контроле успеваемости и промежуточной аттестации обучающихся.

Формы контроля и критерии оценивания приведены в приложении 3 к Рабочей программе.

Наименование работы	Рекомендуемая литература	Трудоемкость, час.
<b>Раздел 1. Технология программирования. Основы алгоритмизации.</b>		
Ознакомление с особенностями наиболее популярных языков программирования. Самостоятельная установка IDE и настройка окружения.	А. А. Андрианова, Л. Н. Исмагилов, Т. М. Мухтарова. . Алгоритмизация и программирование. Практикум: Санкт-Петербург: Лань, 2022 (1) А. Н. Гуцин, Т. И. Лазарева, И. В. Мартынова. . Типовые алгоритмы и их программирование: СПб.БГТУ "ВОЕНМЕХ" им. Д. Ф. Устинова, 2016 (1) Г. С. Иванова. . Технология программирования: М.: Изд-во МГТУ им. Н. Э. Баумана, 2006 (1, 2) Х. М. Дейтел, П. Дж. Дейтел. . Как программировать на C ++: БГТУ "ВОЕНМЕХ" им. Д. Ф. Устинова, 2006 (1) С. З. Свердлов. . Языки программирования и методы трансляции: СПб.: Лань, 2019 (1)	4
Итого по разделу 1		4
<b>Раздел 2. Базовый синтаксис Python.</b>		
Написание программ использующих базовые структуры данных языка программирования Python с применением простых управляющих структур.	Д. Ю. Фёдоров. . Программирование на языке высокого уровня Python: Москва: Юрайт, 2023 (1-11) П. Дж. Дейтел, Х. М. Дейтел. . Python: Искусственный интеллект, большие данные и облачные вычисления: Санкт-Петербург: Питер, 2021 (1-6, 9)	14
Итого по разделу 2		14
<b>Раздел 3. Основные алгоритмические конструкции. Вычислительная сложность.</b>		
Выполнение домашнего задания на разработку алгоритмов.	А. Н. Гуцин, Т. И. Лазарева, И. В. Мартынова. . Типовые алгоритмы и их программирование: СПб.БГТУ "ВОЕНМЕХ" им. Д. Ф. Устинова, 2016 (1) А. А. Андрианова, Л. Н. Исмагилов, Т. М. Мухтарова. . Алгоритмизация и программирование. Практикум: Санкт-Петербург: Лань, 2022 (1-5) П. Дж. Дейтел, Х. М. Дейтел. . Python: Искусственный интеллект, большие данные и облачные вычисления: Санкт-Петербург: Питер, 2021 (3, 4, 7) Т. Кормен, Ч. Лейзерсон, Р. Ривест. .	2
Выполнение задания на обработку массива данных.		3
Выполнение задания на сортировку и поиск.		3
Выполнение задания на оценку вычислительной сложности.		2

	Алгоритмы: построение и анализ: М.: МЦНМО, 2000 (1, 2)	
Итого по разделу 3		10
Раздел 4. Основы объектно-ориентированного программирования.		
Выполнения домашних заданий по реализации объектно-ориентированных моделей в языке программирования Python.	Дж. Кью, М. Джеанини. Объектно-ориентированное программирование: М.: Питер, 2005 (1,2) Д. Ю. Фёдоров. . Программирование на языке высокого уровня Python: Москва: Юрайт, 2023 (12) П. Дж. Дейтел, Х. М. Дейтел. . Python: Искусственный интеллект, большие данные и облачные вычисления: Санкт-Петербург: Питер, 2021 (10)	8
Итого по разделу 4		8
Раздел 5. Стандартные и специализированные библиотеки Python.		
Выполнение домашнего задания с применением стандартных библиотек Python.	П. Дж. Дейтел, Х. М. Дейтел. . Python: Искусственный интеллект, большие данные и облачные вычисления: Санкт-Петербург: Питер, 2021 (7)	4
Выполнение домашнего задания с применением библиотеки NumPy.		4
Выполнение домашнего задания с применением библиотеки Pandas.		4
Выполнение домашнего задания с применением библиотеки Matplotlib.		4
Выполнение домашнего задания с применением библиотеки SciPy.		4
Итого по разделу 5		20
Раздел 6. Создание проекта.		
Выполнение индивидуального практического задания.	Д. Ю. Фёдоров. . Программирование на языке высокого уровня Python: Москва: Юрайт, 2023 (14) П. Дж. Дейтел, Х. М. Дейтел. . Python: Искусственный интеллект, большие данные и облачные вычисления: Санкт-Петербург: Питер, 2021 (1 - 10)	18
Итого по разделу 6		18

## ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

Фонд оценочных средств, позволяющие оценить результаты обучения по данной дисциплине, включают в себя:

- диагностическая работа
- индивидуальное практическое задание;
- домашнее задание;
- зачет.

### Критерии оценивания

#### Диагностическая работа

Диагностическая работа проводится в форме теста в ЭИОС Moodle:

- при правильном ответе менее чем на 60% вопросов - не аттестация;
- при правильном ответе на 60% вопросов и более - аттестация.

#### Индивидуальное практическое задание

Индивидуальное практическое задание связано с разработкой программы на языке программирования Python на свободную тему. Индивидуальное задание считается выполненным, если программа выполняет свои функции в соответствии с заданием и студент способен объяснить алгоритм её работы.

#### Домашнее задание

Домашние задания связаны с разработкой программ на языке программирования Python. Тематика программ соответствует темам раздела. Домашнее задание считается выполненным, если программа выполняет свои функции в соответствии с заданием и студент способен объяснить алгоритм её работы.

#### Зачет

Зачёт ставится на основании выполнения домашних заданий и/или индивидуального практического задания в соответствии с технологической картой на дисциплину.

Паспорт фонда оценочных средств

КУРС	СЕМЕСТР	Наименование разделов и дидактических единиц	ВСЕГО	Аудиторные занятия в контактной форме		Самостоятельная работа студентов	Формируемая компетенция, %		НАИМЕНОВАНИЕ ОЦЕНОЧНОГО СРЕДСТВА
				ВСЕГО	Практические занятия		ОПК-12		
5	10	Раздел 1. Технология программирования. Основы алгоритмизации.	6	2	2	4	10		Индивидуальное практическое задание
5	10	Раздел 2. Базовый синтаксис Python.	22	8	8	14	25		Домашнее задание
5	10	Раздел 3. Основные алгоритмические конструкции. Вычислительная сложность.	18	8	8	10	15		Индивидуальное практическое задание
5	10	Раздел 4. Основы объектно-ориентированного программирования.	12	4	4	8	15		Домашнее задание
5	10	Раздел 5. Стандартные и специализированные библиотеки Python.	30	10	10	20	20		Индивидуальное практическое задание
5	10	Раздел 6. Создание проекта.	20	2	2	18	15		Индивидуальное практическое задание
Всего за 10 семестр			108	34	34	74	100		
Всего по дисциплине			108	34	34	74	100		

## Критерии оценивания

### ОПК-12

№ 1 *Вопросы открытого типа:*  
Что выведет следующий код:

```
data = [1, 2, 5, 4, 1]
```

```
total = 0
```

```
for el in data:
```

```
    if el % 2 == 0:
```

```
        total += el
```

```
print(total)
```

№ 2 Что выведет следующий код:

```
string = 'В теории, теория и практика неразделимы. На практике это не так.'
```

```
chosen_word = string.split()[4]
```

```
length = len(chosen_word)
```

```
print(length)
```

№ 3 Что выведет следующий код:

```
counter = 1
```

```
value = 1
```

```
while counter < 5:
```

```
    if value % 2 == 0:
```

```
        value += 2
```

```
    elif value % 2 == 1:
```

```
        value += 1
```

```
        break
```

```
    counter += 1
```

```
print(value)
```

№ 4 Что выведет следующий код:

```
counter = 1
```

```
value = 1
```

```
while counter < 5:
```



- ```
if value % 2 == 0:
    value += 2
elif value % 2 == 1:
    value += 1
    pass
counter += 1
print(value)
```
- № 5 Что выведет следующий код:
- ```
var = 1/3
print(f'{var:.3f}')
```
- № 6 Что выведет следующий код:
- ```
total = 0
for i in range(2,9,3):
    total += i
print(total)
```
- № 7 Что выведет следующий код:
- ```
data = {'a': [3, 2], 'b': [4, 3], 'c': [1, 3, 2]}
ans = 0
for key in data.keys():
    sub_var = 1
    for element in data[key]:
        sub_var *= element
    ans += sub_var
print(ans)
```
- № 8 Выберите неизменяемые типы данных
- Кортежи (tuple)
  - Списки (list)
  - Строки (str)
  - Словари (dict)
- № 9 Что выведет следующий код:
- ```
a = 1
b = a
```

- № 10      `b = 2`  
`print(a)`  
Выберите, что нужно подставить вместо `_`, чтобы программа вывела ответ:  
`5`
- Программа
- ```
sum = 0
for i in range(_):
    sum += 1
print(i)
```
- Вопросы закрытого типа:
- № 1      Python является ...
- a. интерпретируемым языком.
  - b. компилируемым языком.
  - c. не интерпретируемым языком.
  - d. языком с интерпретацией компилирующего типа.
- № 2
- С точки зрения скорости выполнения программного кода, компилируемые языки программирования чаще всего ...
- a. медленнее интерпретируемых при использовании виртуальной машины.
  - b. быстрее интерпретируемых.
  - c. медленнее интерпретируемых.
  - d. аналогичны интерпретируемым.
- № 3      Python является ...
- a. квазистатическим типизированным языком.
  - b. кинематически типизированным языком.
  - c. динамически типизированным языком.
  - d. статически типизированным языком.
- № 4      При статической типизации ...
- a. переменная заранее не связывается с конкретным типом хранимых в ней данных, при этом в ней можно хранить сразу несколько разных типов.
  - b. переменная связывается с типом хранимых в ней данных в момент её объявления в программе, при этом её тип не может быть изменён позже.
  - c. переменная связывается с типом хранимых в ней данных в момент её в программе, при этом её тип может быть изменён позже.
  - d. переменная связывается с типом хранимых в ней данных в момент присвоения ей значения, при этом её тип может быть изменён позже при

- присвоении нового значения.
- № 5
- При динамической типизации ...
- a. переменная связывается с типом хранимых в ней данных в момент присвоения ей значения, при этом её тип может быть изменён при присвоении значения другого типа.
  - b. переменная связывается с типом хранимых в ней данных в момент её объявления в программе, при этом её тип не может быть изменён позже.
  - c. переменная связывается с типом хранимых в ней данных в момент её объявления в программе, при этом при необходимости изменения её типа придётся динамически создать новую переменную с требуемым типом.
  - d. переменная связывается с типом хранимых в ней данных в момент присвоения ей значения, при этом её тип не может быть изменён позже.
- № 6
- Императивное программирование отличается от декларативного тем, что ...
- a. при императивном стиле программирования реализуется объектно-ориентированная модель программирования, а при декларативном используется структурный стиль программирования.
  - b. при императивном стиле используется структурная парадигма программирования, а при декларативном реализуется объектно-ориентированная модель программирования.
  - c. при императивном стиле программирования описывается алгоритм получения желаемого результата, а при декларативном описывается какой именно результат должен быть получен в результате работы программы.
  - d. при императивном стиле программирования описывается какой именно результат должен быть получен в результате работы программы, а при декларативном алгоритм получения желаемого результата.
- № 7
- При структурном программировании ...
- a. программа представляется в виде иерархической структуры блоков.
  - b. программа создаётся путём манипуляции с графическими объектами.
  - c. программа представляется в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.
  - d. процесс вычисления трактуется как вычисление значений функций в математическом понимании последних.
- № 8
- При объектно-ориентированном программировании ...
- a. процесс вычисления трактуется как вычисление значений функций в математическом понимании последних.
  - b. программа представляется в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.
  - c. программа создаётся путём манипуляции с графическими объектами.
  - d. программа представляется в виде иерархической структуры блоков.
- № 9
- При функциональном программировании ...

- a. процесс вычисления трактуется как вычисление значений функций в математическом понимании последних.
- b. программа представляется в виде иерархической структуры блоков
- c. программа создаётся путём манипуляции с графическими объектами.
- d. программа представляется в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.
- № 10 При визуальном программировании ...
- a. программа представляется в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.
- b. процесс вычисления трактуется как вычисление значений функций в математическом понимании последних.
- c. программа создаётся путём манипуляции с графическими объектами.
- d. программа представляется в виде иерархической структуры блоков.
- № 11 Рекурсивная функция ...
- a. это функция обладающая асимптотической сложностью  $O(1)$ .
- b. это любая функция заданная в программном коде.
- c. это функция, заданная самоподобным образом.
- d. это функция задающая однозначные соответствия между переменными величинами на всём области её определения.
- № 12 Вычислительная сложность ...
- a. это временные затраты на разработку алгоритма решения поставленной программисту задачи.
- b. это когда процесс численных вычислений имеет стохастический характер.
- c. это понятие, обозначающее функцию зависимости объема работы алгоритма от размера обрабатываемых данных.
- d. это частный случай работы программы, когда алгоритм задан самоподобным образом.
- № 13 Выберите наихудшую асимптотическую сложность работы алгоритма.
- a.  $O(\log N)$
- b.  $O(N)$
- c.  $O(N^2)$
- d.  $O(N \cdot \log N)$
- № 14 Выберите наихудшую асимптотическую сложность работы алгоритма.
- a.  $O(N \cdot \log N)$
- b.  $O(1)$
- c.  $O(\log N)$
- d.  $O(N)$
- № 15 Выберите наилучшую асимптотическую сложность работы алгоритма.

- a.  $O(N \cdot \log N)$   
 b.  $O(\log N)$   
 c.  $O(N)$   
 d.  $O(1)$
- № 16 Выберите наилучшую асимптотическую сложность работы алгоритма.
- a.  $O(N \cdot \log N)$   
 b.  $O(N^2)$   
 c.  $O(\log N)$   
 d.  $O(N)$
- № 17 Выберите верное утверждение.
- a. Python является сугубо функциональным языком программирования.  
 b. Python является мультипарадигменным языком программирование.  
 c. Python является сугубо объектно-ориентированным языком программирования.  
 d. Python является сугубо структурным языком программирования.
- № 18 Какое значение примет переменная:
- `var = '4' + '5'`
- a. 45  
 b. '45'  
 c. 9  
 d. '9'
- № 19 е. Программа выдаст ошибку  
 Что выведет следующий код:
- `string = 'штурм'`  
`string[2] = 'о'`  
`print(string)`
- a. 'шуурм'  
 b. 'шторм'  
 c. 'штурм'  
 d. 'о'
- № 20 е. Исключение `TypeError`  
 Какая ошибка допущена в функции вычисления n-члена ряда Фибоначчи:

```
def fib_calc(n):
    return fib_calc(n-1)
```

- a. Отсутствует обработка базового случая рекурсии
- b. Нельзя вызывать в функции саму себя
- c. Не задано значение аргумента по-умолчанию
- d. Отсутствует объявление типа переменной
- e. Ошибка в синтаксисе задания функции

№ 21 Как вывести справку по объекту из строк документации?

- a. info(x)
- b. dir(x)
- c. help(x)
- d. doc(x)

№ 22 Пусть задан кортеж:

```
t = (1, 2, 3)
```

Как сделать чтобы программа возвращала кортеж типа:

```
(1, 2, 4)
```

- a. t[:-1] + (4,)
- b. t[3] = 4
- c. t[:-1] + (4)
- d. t[2] = 4

№ 23 Как подключить дополнительную библиотеку?

- a. connect library
- b. plug library
- c. add library
- d. #include library
- e. import library

№ 24 Как вычислить возведение 3 в степень 5?

- a. 3\*\*5
- b.

```
def func(num, p):
```

```
ans = 0

for i in range(num):

    ans *= i

return ans
```

```
func(3, 5)
```

```
c. mod(5, 3)
```

```
d. int(3)^5
```

```
e. 3^5
```

№ 25 Как добавить новую запись в словарь?

```
a. dict(key) = 'value'
```

```
b. dict[key] = 'value'
```

```
c. dict.append(value)
```

```
d. dict(key, 'value')
```

№ 26 Выберите, что нужно подставить вместо \_, чтобы программа верно определила площадь окружности

```
import math
```

```
r = 3
```

```
area = _
```

```
print(area)
```

```
a. math.pi * r**2
```

```
b. pi * r ^ r
```

```
c. pi * r**2
```

```
d. math.pi * r^2
```

```
e. math['pi'] * r**2
```

№ 27 Как получить матрицу вида:

```
a.
```

```
import numpy as np
```

```
arr = np.array((3, 3))
```

```
arr = np.numerate(1, 9, 1)
```

```
print(arr)
```

b.

```
import numpy as np
```

```
arr = np.linspace(1, 10)
```

```
arr = np.reshape(arr, (3, 3))
```

```
print(arr)
```

c.

```
import numpy as np
```

```
arr = np.arange(1, 10)
```

```
print(arr)
```

d.

```
import numpy as np
```

```
arr = np.arange(1, 10)
```

```
arr = np.reshape(arr, (3, 3))
```

```
print(arr)
```

e.

```
import numpy as np
```

```
arr = np.zeros((3, 3))
```

```
arr = np.numerate(1, 9, 1)
```

```
print(arr)
```

№ 28

Выберите правильный синтаксис для создания массива из списка

a. `numpy.linspace(1, 2, 3)`

b. `numpy.massive(1, 2, 3)`

c. `numpy.array([1, 2, 3])`

d. `numpy.massive([1, 2, 3])`

e. `numpy.linspace([1, 2, 3])`

f. `numpy.array(1, 2, 3)`

№ 29

Какой функцией можно создать массив 3x3, состоящий только из нулей?

a. `numpy.null([3, 3])`

b. `numpy.zeros(3, 3)`

c. `numpy.zeros([3, 3])`



- d. `numpy.empty(3, 3)`  
e. `numpy.null(3, 3)`  
f. `numpy.empty([3, 3])`
- № 30 Как определить количество элементов в каждом измерении массива `arr`?
- a. `arr.size`  
b. `arr.shape()`  
c. `len(arr)`  
d. `length(arr)`  
e. `arr.shape`  
f. `arr.size()`  
g. `size`
- № 31 Какая из библиотек используется непосредственно для работы с матрицами?
- a. `scipy`  
b. `sklearn`  
c. `numpy`  
d. `pandas`  
e. `matplotlib`
- № 32 Что нужно подставить вместо \_\_\_\_\_, чтобы построить 2D график из линий?
- ```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5]
y = [1, 4, 9, 16, 25]
plt.____(x, y)
```
- a. `plot`  
b. `bar`  
c. `scatter`  
d. `hist`  
e. `graph`
- № 33 Какая из библиотек используется для построения графиков?
- a. `pandas`  
b. `matplotlib`  
c. `scipy`  
d. `sklearn`



- c. matplotlib
- d. scipy
- e. numpy
- № 38 Какая из библиотек лучше всего подходит обработки и анализа табличных данных?
- a. scipy
- b. pandas
- c. sklearn
- d. numpy
- e. matplotlib
- № 39 Какая функция из scipy подойдёт для одномерной интерполяции зависимости  $y$  от  $x$ ?
- spl = interpolate.\_\_\_\_(x, y)
- a. int
- b. interpolation1d
- c. interp1d
- d. interpolation
- e. interp
- № 40 Как скалярно умножить матрицы arr1 и arr2 с помощью numpy?
- a. arr1 \* arr2
- b. numpy.scalar(arr1, arr2)
- c. numpy.vector(arr1, arr2)
- d. numpy.dot(arr1, arr2)
- e. numpy.cross(arr1, arr2)
- № 41 Как векторно умножить матрицы arr1 и arr2 с помощью numpy?
- a. numpy.vector(arr1, arr2)
- b. numpy.scalar(arr1, arr2)
- c. arr1 \* arr2
- d. numpy.dot(arr1, arr2)
- e. numpy.cross(arr1, arr2)
- № 42 Как поэлементно умножить элементы матриц arr1 и arr2 с помощью numpy?
- a. numpy.cross(arr1, arr2)
- b. numpy.scalar(arr1, arr2)
- c. numpy.vector(arr1, arr2)
- d. arr1 \* arr2
- e. numpy.dot(arr1, arr2)