МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Воронежский государственный технический университет»

УТВЕРЖДАЮ

Декан ФЭМИТ

_____ С.А. Баркалов

_____

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Алгоритмы и структуры данных (продвинутый Python)»

Направление подготовки 27.03.03 Системный анализ и управление

Профиль Бизнес-аналитика и системы больших данных

Квалификация выпускника бакалавр

Нормативный период обучения 4 года

Форма обучения очная

Год начала подготовки 2023

Автор программы _____ В.Е. Белоусов

Заведующий кафедрой
Базовая кафедра
кибернетики в системах
организационного
управления _____ В.Е. Белоусов

Руководитель ОПОП _____ О.С. Перевалова

Воронеж 2023

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

1.1. Цели дисциплины является овладение современными понятиями и способами написания программ, необходимыми в профессиональной практической деятельности.

1.2. Задачи освоения дисциплины - развитие навыков программирования на языке Python;

- формирование у обучаемых практических знаний для самостоятельного создания и использования сложных структур данных;
- разработка архитектуры, алгоритмических и программных решений системного и прикладного программного обеспечения;
- развитие и использование математических и информационных инструментальных средств, автоматизированных систем в научной и практической деятельности.

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП

Дисциплина «Алгоритмы и структуры данных (продвинутый Python)» относится к дисциплинам обязательной части блока Б1.

3. ПЕРЕЧЕНЬ ПЛАНИРУЕМЫХ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ

Процесс изучения дисциплины «Алгоритмы и структуры данных (продвинутый Python)» направлен на формирование следующих компетенций:

ОПК-6 - Способен разрабатывать методы моделирования, анализа и технологии синтеза процессов и систем, а также алгоритмы и программы, основанные на этих методах, пригодные для практического применения в области техники и технологии

ОПК-8 - Способен принимать научно обоснованные решения в области системного анализа и автоматического управления на основе знаний профильных разделов математики, физики, информатики, методов системного и функционального анализа, теории управления и теории знаний

ОПК-10 - Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности

Компетенция	Результаты обучения, характеризующие сформированность компетенции
ОПК-6	знать современные информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности
	уметь выбирать современные информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности

	владеть навыками применения современных информационных технологий и программных средств, в том числе отечественного производства, при решении задач профессиональной деятельности
ОПК-8	знать основные требования, синтаксис и принципы разработки программного обеспечения на языке Python
	уметь разрабатывать консольные приложения и приложения с графическим пользовательским интерфейсом на языке Python
	владеть навыками разработки приложений на языке Python
ОПК-10	знать основные языки программирования и работы с базами данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий
	уметь применять языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ.
	владеть навыками программирования, отладки и тестирования прототипов

4. ОБЪЕМ ДИСЦИПЛИНЫ

Общая трудоемкость дисциплины «Алгоритмы и структуры данных (продвинутый Python)» составляет 4 з.е.

Распределение трудоемкости дисциплины по видам занятий

очная форма обучения

Виды учебной работы	Всего часов	Семестры
		3
Аудиторные занятия (всего)	90	90
В том числе:		
Лекции	36	36
Лабораторные работы (ЛР)	54	54
Самостоятельная работа	54	54
Курсовой проект	+	+
Виды промежуточной аттестации - зачет с оценкой	+	+
Общая трудоемкость: академические часы зач.ед.	 144 4	 144 4

5. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

5.1 Содержание разделов дисциплины и распределение трудоемкости по видам занятий

очная форма обучения

№ п/п	Наименование темы	Содержание раздела	Лекц	Лаб. зан.	СРС	Всего, час
1	Введение в Python	История PYTHON. Принципы PYTHON. Синтаксис и первый код. Создание переменных. Операции с переменными. Вывод переменных. Пользовательский ввод. Ошибки и исключения. Описание стандартных типов данных. Проверка типа данных. Явное преобразование типов. Неявное преобразование типов. Арифметические операции. Порядок операций. Продвинутая арифметика. Обработка вещественных чисел в PYTHON. Библиотека MATH. Итоговые задачи.	6	10	8	24
2	Условные операторы функции	Значение true/false. Преобразование к типу bool. Операторы сравнения. Сравнение строк. Логические операции. Ленивые вычисления. Сложные логические операции. Инструкция if. Инструкция if: else:. Инструкция x if y else z. Инструкция if-elif-else. Вложенные инструкции if (if внутри if). Объявление функции. Результат функции. Аргументы функции. Итоговые задачи.	6	10	8	24
3	Работа со строками	Простые приемы работы со строками. Методы строки. UNICODE И BYTES. Форматирование строк. Метод format(). Функции и методы для работы со строками и символами. Регулярные выражения: синтаксис, поиск по шаблону. Итоговые задачи.	6	10	8	24
4	Программирование разветвляющихся и циклических процессов на языке Python.	Операции сравнения. Операторы условного перехода. Операторы цикла. Функции range() и enumerate(). Операторы перехода на следующую итерацию и прерывания цикла. Вложенные циклы. Циклы и ветвления в программах. Перебор элементов коллекций. Использование контекста. Итоговые задачи.	6	8	10	24
5	Массивы	Списки. Создание списка. Операции над списками. Многомерные списки. Перебор элементов списка. Генераторы списков и выражения-генераторы. Функции для работы со списками. Добавление и удаление элементов списка. Поиск элемента в списке и получение сведений о значениях, входящих в список. переворачивание и перемешивание списка. Выбор элемента списка случайным образом. Сортировка списка. Заполнение списка числами. Преобразование списка в строку. Кортежи. Операции над кортежами. Множества. Операции над множествами. Диапазоны. Операции над диапазонами. Словари. Создание словаря. Операции над словарями. Перебор элементов словаря. Методы для работы над словарями. Генераторы словарей. Итоговые задачи.	6	8	10	24
6	Объектно-ориентированное программирование на языке Python	Понятие класса, атрибута и метода. Определение класса и создание экземпляра класса. Конструкторы и деструкторы. Наследование. Множественное наследование. Понятие примесей и их использование. Специальные методы классов. Перегрузка операторов. Статические методы и методы классов. Абстрактные методы. Ограничение	6	8	10	24

		доступа к идентификаторам внутри класса. Свойства классов. Декораторы классов. Итератор класса. Понятие итератора класса и его использование. Контейнеры. Контейнеры-последовательности. Контейнеры словари. Перечисления. Атрибуты и методы перечислений. Итоговые задачи.				
		Итого	**36**	**54**	**54**	**144**

5.2 Перечень лабораторных работ

Лабораторная работа №1: «Линейные программы».

Лабораторная работа №2: «Условия и функции».

Лабораторная работа №3: «Работа с массивами. Часть 1».

Лабораторная работа №4: «Работа с массивами. Часть 2».

Лабораторная работа №5: «Основы объектно-ориентированного программирования на Python».

Лабораторная работа №6: «Продвинутое Основы объектно-ориентированного программирование на Python».

6. ПРИМЕРНАЯ ТЕМАТИКА КУРСОВЫХ ПРОЕКТОВ (РАБОТ) И КОНТРОЛЬНЫХ РАБОТ

В соответствии с учебным планом освоение дисциплины предусматривает выполнение курсового проекта в 4 семестре для очной формы обучения.

Примерная тематика курсового проекта: «Методы продвинутой работы с датасетом Titanic.csv».

Задачи, решаемые при выполнении курсового проекта:

• Анализ датасета Titanic.csv.

• Приведение нечисловых параметров к числовым. Поля Pclass, Age, Sibsp, Parch, Fare являются числовыми и не требуют преобразования. Визуальное изучение данных показало, что поля Age и Fare могут принимать пустые значения. Для пассажиров, значение которых не указано, предполагается использование медианы среди ненулевых значений этого параметра. В поле Sex предполагается значения male заменить на 0, значения female — на 1. Поле Embarked также может принимать пустое значение. Для таких записей поле Embarked будет заполнено значением S. Преобразование к числу значений поля Embarked происходит следующим образом: S заменяется на 0, C — на 1, Q — на 2. Для обучения используются следующие поля: Pclass, Sex, Age, SibSp, Parch, Fare, Embarked.

Примеры индивидуальных заданий

№ варианта	Группа (0-погибшие, 1-выжившие)	Условия
1	0	Женщины от 20 до 30 лет, севших в порту Квинстаун, имеющих братьев
2	0	Мужчины от 20 до 30 лет, севших в порту Шербур, имеющих сестер
3	0	Дети до 7 лет, севших в порту Саутгемптон,

		имеющих отца и мать
4	0	Женщины от 31 до 40 лет, севших в порту Квинстаун, имеющих дочерей
5	0	Мужчины от 31 до 40 лет, севших в порту Шербур, имеющих сыновей
6	1	Дети от 8 до 10 лет, севших в порту Шербур, имеющих отца и мать
7	1	Женщины от 41 до 50 лет, севших в порту Саутгемптон, имеющих сыновей
8	1	Мужчины от 41 до 50 лет, севших в порту Квинстаун, имеющих братьев
9	1	Дети от 11 до 13 лет, севших в порту Саутгемптон, имеющих сестер
10	1	Пассажиры 1 класса, имевшие более одной каюты, имеющие братьев
11	0	Пассажиры 2 класса, ехавшие в одной одной каюте, имеющие сестер
12	0	Пассажиры 3 класса, севшие в Квинстаун, имевшие либо брата либо сестру
13	0	Дети, имеющие 1 брата или сестру, севшие в порту Шербур, из полных семей
14	0	Дети, имеющие 2 братьев и сестер, севшие в порту Саутгемптон, из неполных семей
15	0	Дети, имеющие 1 родителя, севшие в порту Квинстаун, имеющие братьев
16	1	Дети, имеющие двух родителей, пассажиры 2 класса, имеющие братьев или сестер, севшие в порту Саутгемптон
17	1	Дети (мальчики), пассажиры 1 класса, севшие в порту Саутгемптон, Шербур, из полных семей
18	1	Дети (девочки), пассажиры 2 класса, ехавшие в одной одной каюте, имеющие сестер

Курсовой проект включат в себя графическую часть и расчетно-пояснительную записку.

7. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

7.1. Описание показателей и критериев оценивания компетенций на различных этапах их формирования, описание шкал оценивания

7.1.1 Этап текущего контроля

Результаты текущего контроля знаний и межсессионной аттестации оцениваются по следующей системе:

«аттестован»;

«не аттестован».

Компе-тенция	Результаты обучения, характеризующие сформированность компетенции	Критерии оценивания	Аттестован	Не аттестован
ОПК-6	знать современные	Полное или частичное	Выполнение работ в	Невыполнение

	информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности	посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	срок, предусмотренный в рабочих программах	работ в срок, предусмотренный в рабочих программах
	уметь выбирать современные информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности	Полное или частичное посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах
	владеть навыками применения современных информационных технологий и программных средств, в том числе отечественного производства, при решении задач профессиональной деятельности	Полное или частичное посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах
ОПК-8	знать основные требования, синтаксис и принципы разработки программного обеспечения на языке Python	Полное или частичное посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах
	уметь разрабатывать консольные приложения и приложения с графическим пользовательским интерфейсом на языке Python	Полное или частичное посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах
	владеть навыками разработки приложений на языке Python	Полное или частичное посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах
ОПК-10	знать основные языки программирования и работы с базами данных, операционные системы и оболочки,	Полное или частичное посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах

	современные программные среды разработки информационных систем и технологий			
	уметь применять языки программирования и работы с базами данных, современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ.	Полное или частичное посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах
	владеть навыками программирования, отладки и тестирования прототипов	Полное или частичное посещение лекционных, практических и лабораторных занятий. Выполненые СРС, УО, ПО	Выполнение работ в срок, предусмотренный в рабочих программах	Невыполнение работ в срок, предусмотренный в рабочих программах

7.1.2 Этап промежуточного контроля знаний

Результаты промежуточного контроля знаний оцениваются в 3 семестре для очной формы обучения по четырехбалльной системе:

«отлично»;

«хорошо»;

«удовлетворительно»;

«неудовлетворительно».

Компе-тенция	Результаты обучения, характеризующие сформированность компетенции	Критерии оценивания	Отлично	Хорошо	Удовл.	Неудовл.
ОПК-6	знать современные информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности	Тест	Выполнение теста на 90- 100%	Выполнение теста на 80-90%	Выполнение теста на 70-80%	В тесте менее 70% правильных ответов
	уметь выбирать современные информационные технологии и программные средства, в том числе отечественного производства при решении задач	Решение стандартных практических задач	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены

	профессиональной деятельности					
	владеть навыками применения современных информационных технологий и программных средств, в том числе отечественного производства, при решении задач профессиональной деятельности	Решение прикладных задач в конкретной предметной области	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены
ОПК-8	знать основные требования, синтаксис и принципы разработки программного обеспечения на языке Python	Тест	Выполнение теста на 90- 100%	Выполнение теста на 80-90%	Выполнение теста на 70-80%	В тесте менее 70% правильных ответов
	уметь разрабатывать консольные приложения и приложения с графическим пользовательским интерфейсом на языке Python	Решение стандартных практических задач	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены
	владеть навыками разработки приложений на языке Python	Решение прикладных задач в конкретной предметной области	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены
ОПК-10	знать основные языки программирования и работы с базами данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий	Тест	Выполнение теста на 90- 100%	Выполнение теста на 80-90%	Выполнение теста на 70-80%	В тесте менее 70% правильных ответов
	уметь применять языки программирования и работы с базами данных, современные программные среды разработки	Решение стандартных практических задач	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены

	информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, ведения баз данных и информационных хранилищ.					
	владеть навыками программирования, отладки и тестирования прототипов	Решение прикладных задач в конкретной предметной области	Задачи решены в полном объеме и получены верные ответы	Продемонстрирован верный ход решения всех, но не получен верный ответ во всех задачах	Продемонстрирован верный ход решения в большинстве задач	Задачи не решены

7.2 Примерный перечень оценочных средств (типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности)

7.2.1 Примерный перечень заданий для подготовки к тестированию

1. Что получится в результате выполнения следующего кода?

```
a = bool("True")
b = bool()
print(f"{a}-{b}")
```

А. True-True
Б. True-False
В. False-True
Г. False-False

2. Сколько элементов содержит в себе range(999)?
А. 1000
Б. 1
В. 999
Г. 998

3. Какой список сформируется, если выполнить команду list(range(9, 3))?
А. [3, 4, 5, 6, 7, 8]
Б. [3, 4, 5, 6, 7, 8, 9]
В. []
Г. [9, 8, 7, 6, 5, 4]
Д. [9, 8, 7, 6, 5, 4, 3]

4. Сколько раз программа ниже напечатает фразу «hello world»?

```
t = 3
while t <= 8:
print("hello world")
t += 1
```

А. 6
Б. Бесконечность
Г. 3
Д. 2

5. Сколько раз программа ниже напечатает фразу «hello world»?

```
t = 3
while t <= 8:
print("hello world")
t -= 1
```

А. 6
Б. Бесконечность
В. 3
Г. 2

6. Какое значение примет переменная "a" после выполнения программы?

```
a = 3
while a <= 48:
a *= 2
```

А. 48
Б. 24
В. 96
Г. 192

7. Каков результата выполнения кода?

```
number = 1234567890
count = 0
while number > 0:
last_digit = number % 10
    if last_digit < 3:
        count = count + 1
    number = number // 10
print(count)
```

А. 4
Б. 5
В. 3

8. Какой оператор используется для выполнения итераций по элементам списка в Python?

А. for
Б. while
В. do-while
Г. foreach

9. Какой оператор используется для прерывания цикла и перехода к следующей итерации в Python?

А. pass
Б. break
В. continue
Г. return

10. Какая функция используется для создания списка чисел в заданном диапазоне в Python?

А. range()
Б. list()
В. append()
Г. sort()

7.2.2 Примерный перечень заданий для решения стандартных задач

1. Что важно учитывать при разработке алгоритма?

А. время его работы
Б. количество потребляемой памяти
В. количество строк кода
Г. сложность его прочтения

2. Какие группы алгоритмов выделяются в академических курсах?

А. линейные
Б. квадратичные
В. циклические
Г. с ветвлением

3. Что из перечисленного является алгоритмом?

А. правила техники безопасности
Б. инструкция по приготовлению бутерброда
В. список книг в библиотек

4. При оценивании функций какая оценка соответствует символике f = O(g)?

А. оценка снизу
Б. оценка сверху
В. средняя оценка

5. Какая оценка по времени у простого алгоритма сортировки?

А. O(n^2)
Б. O(n)
В. O(n*log(n))

6. Какая оценка по времени у самого быстрого алгоритма сортировки?

А. O(n^2)
Б. O(n)
В. O(nlog(n))

7. Какая оценка по времени у алгоритма бинарного поиска?
А. O(n^2)
Б. O(n)
В. O(log(n))

8. Ниже представлены три алгоритма решения следующей задачи: Для заданного списка выведите последовательно: только положительные элементы, только отрицательные, нулевые.

Алгоритм 1

```
positive = []
   for elem in array:
      if elem > 0:
         positive.append(elem)
negative = []
    for elem in array:
       if elem < 0:
          negative.append(elem)
zero = []
     for elem in array:
        if elem == 0:
           zero.append(elem)
print(*positive)
print(*negative)
print(*zero)
```

Алгоритм 2

```
positive = []
negative = []
zero = []
   for elem in array:
      if elem > 0:
         positive.append(elem)
      elif elem < 0:
         negative.append(elem)
      else:
         zero.append(elem)
print(*positive)
print(*negative)
print(*zero)
```

Алгоритм 3

```
for elem in array:
    if elem > 0:
        print(elem)
for elem in array:
    if elem < 0:
        print(elem)
for elem in array:
    if elem == 0:
        print(elem)
```

Какой из представленных алгоритмов имеет наименьшую оценку по времени?

А. Алгоритм 1

Б. Алгоритм 2

В. Алгоритм 3

9. Почему рекурсивный алгоритм поиска обычно не применяют на практике?

А. Из-за того, что дроби в компьютере используются лишь конечной точности, числа считаются с погрешностью

Б. Из-за больших обращений функции к самой себе время значительно увеличивается

В. Может переполниться массив, в котором мы храним предыдущие значения вызовов функции

10. Поиск индекса. Дан упорядоченный массив элементов и значение некоторого элемента. Имплементируйте бинарный поиск для нахождения индекса элемента n в заданном массиве. После нахождения индекса, выведите его на экран. Если элемента в массиве нет, выведите None.

```
n = int(input())
arr = list(map(int, input().split()))

def binary_search(lst:list, item:int) -> int:
    first = 0
    last = len(arr)-1
    index = -1
    while (first <= last) and (index == -1):
        mid = (first+last)//2
        if lst[mid] == item:
            index = mid
        else:
            if item<lst[mid]:
                last = mid -1
            else:
                first = mid +1
```

```
    return index
print(binary_search(arr, n))
```

7.2.3 Примерный перечень заданий для решения прикладных задач

1. Класс и его инициализатор 1

Создайте класс **Motorbike**, имеющий инициализатор **__init__**, который принимает объем двигателя и год выпуска мотоцикла в качестве аргументов и сохраняет их в атрибутах engine и year.

Объем двигателя и год выпуска мотоцикла передаются на вход программы пользователем.

```
# не изменяйте код ниже, он нужен для проверки
v = float(input())
y = int(input())
bike = Motorbike(v, y)
print(bike.engine, bike.year)
```

2. Класс и его инициализатор 2

Создайте класс **Car**, имеющий инициализатор **__init__**, который принимает и инициализирует атрибуты класса: brand, model, price, а также атрибут name — строку следующего вида: <Брэнд> <Модель>.

Брэнд, модель и цена автомобиля передаются на вход программы пользователем.

```
# не изменяйте код ниже, он нужен для проверки
b = input()
m = input()
p = int(input())
tesla = Car(b, m, p)
print(tesla.price)
print(tesla.name)
```

3. Хоккеист. Создайте класс **HockeyPlayer**, у которого есть:

- **инициализатор __init__**, принимающий в качестве аргументов first_name и last_name, то есть имя и фамилию хоккеиста. Также во время инициализации объекта необходимо создать два атрибута-счетчика: goals и assists. Эти атрибуты будут хранить информацию о забитых голах и передачах каждого хоккеиста. Оба атрибута необходимо проинициализировать нулями;
- метод **add_goals**, который добавляет количество голов в счетчик игрока. По умолчанию добавляет один гол;
- метод **add_assists**, который добавляет количество передач в счетчик игрока. По умолчанию добавляет одну передачу;
- метод **statistics**, который считает результативность игрока и возвращает строку вида:

<Имя> <Фамилия> - <голы> + 0.5*<передачи>

Фамилия, имя, количество голов и передач хоккеиста передаются на вход программы пользователем.

```
# не изменяйте код ниже, он нужен для проверки
name = input()
surname = input()
goals = int(input())
assists = int(input())
player = HockeyPlayer(name, surname)
player.add_goals(goals)
player.add_assists(assists)
print(player.statistics())
```

4. Раскраска зебры. Создайте класс **Zebra** с методом **which_stripe**, который поочередно печатает фразы "Полоска белая", "Полоска черная" и так далее начиная с фразы "Полоска белая".

```
# не изменяйте код ниже, он нужен для проверки
z1 = Zebra()
z1.which_stripe()
z1.which_stripe()
z1.which_stripe()
```

5. Линейный поиск в массиве. В программе задан массив, состоящий из некоторого количества символьных элементов. На вход программы передается символ x. Используя линейный поиск (полный обход массива) подсчитайте количество вхождений символа x в заданный массив.

Выведите результат на экран.

```
arr = ['a', 'b', 'c', 'a', 'a', 'b', 'b', 'a']
result = 0
x = input()
for i in range(0, len(arr)):
    if (arr[i] == x):
        print(i)

print(result)
```

6. Первые и последние элементы в массиве. В программе задан массив, состоящий из некоторого количества символьных элементов. На вход программы передается символ x. Используя линейный поиск (полный обход массива) подсчитайте количество элементов между самым первым и самым последним вхождениями элемента x в массив.

Выведите результат на экран.

Если x встречается в массиве всего 1 раз или не встречается вовсе — выведите -1.

```
arr = ['a', 'b', 'c', 'c', 'a', 'b', 'b', 'a']
min_pos, max_pos = len(arr), 0
x = input()
# ваш код здесь

print(max_pos - min_pos - 1 if min_pos<=max_pos else -1 )
```

7. Индекс последнего элемента в массиве. В программе задан массив, состоящий из некоторого количества целочисленных элементов. На вход

программы передается целое число x. Используя линейный поиск (полный обход массива) найдите индекс последнего элемента в массиве, который больше x и является четным.

Выведите найденный индекс на экран.

Если индекс не найден, верните -1.

```
arr = [1, 12, 4, 7, 11, 4, 8]
result = -1
x = int(input())
# ваш код здесь

print(result)
```

8. **Простая сортировка**. Дан неупорядоченный массив элементов. Реализуйте простую сортировку вставкой и выведите получившийся массив в качестве ответа.

```
arr = list(map(int, input().split()))
def insertion_sort(arr):
    for i in range(1, len(arr)):
        value = arr[i]      #Создаем временную переменную для хранения сравниваемого элемента массива
        j = i - 1           #берем отсортированный элемент
        while j >= 0 and value < arr[j]:    #Задаем цикл - индекс следующий за отсортированным и значение больше
            arr[j + 1] = arr[j]      # Перемещаем отсортированный элемент вперед
            j -= 1          #   Уменьшаем индекс
        arr[j + 1] = value     # Присваиваем значение
    return arr     # Возвращаем список
print("The sorted list1 is:", insertion_sort(arr))
```

9. **Сортировка пузырьком**. Программа принимает на вход строку, состоящую из целочисленных значений разделенных пробелом и преобразует их в массив. Напишите свою функцию, которая принимает на вход в качестве аргумента массив, реализует сортировку пузырьком по возрастанию, а затем возвращает отсортированный массив. В качестве проверки программа сверяет результат сортировки.

```
    def bubble_sort(arr):
    for i in range(len(arr) - 1):
        for j in range(len(arr) - i - 1):
            if arr[j] > arr[j + 1]:
                buff = arr[j]
                arr[j] = arr[j + 1]
                arr[j + 1] = buff
    return arr

arr = list(map(int, input().split()))
arr_sort = bubble_sort(arr)
print(arr_sort==sorted(arr))
```

10. **Подсчет инверсий**. Инверсией называется пара чисел **x**, **y**, таких что **x < y**, при этом **pos[x] > pos[y]** (то есть **x** стоит правее **y** в массиве).

Программа принимает на вход строку, состоящую из целочисленных значений разделенных пробелом и преобразует их в массив. Напишите свою функцию, которая принимает на вход в качестве аргумента массив, реализует подсчет количества инверсий, а затем возвращает результат подсчета. В качестве проверки программа сверяет результат сортировки.

```
def inversion_count(arr):
    # ваш код здесь

# Не удаляйте код ниже, он нужен для проверки
arr = list(map(int, input().split()))
inv_cnt = inversion_count(arr)
print(inv_cnt)
```

7.2.4 Примерный перечень вопросов для подготовки к зачету

Не предусмотрено учебным планом

7.2.5 Примерный перечень заданий для подготовки к экзамену

1.Встроенный тип str. Методы объекта str. print() и форматирование вывода.

2.Работа с файловой системой средствами Python. Работа с файлами. Методы open(), close(), read(), write().

3.Модуль re. Синтаксис регулярных выражений, метасимволы. Методы compile(), match(),search(),findall(),split(),sub(),subn(). Нумерованные и именованные группы в шаблонах поиска.

4.Встроенные типы последовательностей list, tuple, range и их методы.

5.Встроенный объект dict и его методы. Множества. Встроенные типы set и frozenset.

6.Встроенные типы чисел — int, float, complex. Машинное представление чисел с плавающей точкой и целых. Преобразование типов при сравнении чисел. Рациональные числа. Модуль fractions. Двоичное представление чисел. Неассоциативность операций в арифметике с плавающей запятой. Целые числа с произвольной точностью.

7.Инструкции и синтаксис. Составные конструкции. Инструкции if/else/elif, логические операторы и выражения сравнения

8.Циклы while и for в Python

9.Функции в Python. Основные понятия Передача аргументов в функцию. Специальные режимы сопоставления аргументов.

10.Парадигма объектно-ориентированного программирования. Поддержка в Python функционального программирования.

11.Объекты. Динамическая типизация. Инкапсуляция.

12.Области видимости и пространство имен в Python. Генерация объекта class. Новое пространство имен. Объект экземпляр класса.

13.Атрибуты класса. Атрибуты данных. Атрибуты-методы. Параметр self. Добавление атрибутов к классу во время исполнения программы.

14.Специальные методы и атрибуты классов. Методы __init__() и ___del__() в Python. Декораторы функций и декораторы классов. Инструменты интроспекции в Python. Метаклассы.

15.Абстрактные методы в Python. Классические классы и классы нового стиля.

16.Наследование. Базовый и производный класс. Построение производного класса.

17.Агрегация. Контейнеры. Иерархия наследования.

18.Полиморфизм. Подмена методов в производном классе. Доступ к методам базового класса.

19.Обработка исключений. Инструкция try… exept… else… finaly. Объект Менеджер контекста и конструкция with…as. Классы встроенных исключений. Пользовательские исключения. Генерация заданного исключения с помощью Отладочные проверки assert и возбуждение исключения AssertionError.raise. Поиск ошибок программирования на стадиях разработки и тестирования.

20.Устойчивость объектов. Время жизни объектов. Их сохранение для следующего запуска программы и/или передачи на другой компьютер. Модуль pickle для хранения представлений объектов в виде байтовых последовательностей и их последующего восстановления (сериализация и десериализация). Модуль shelve – БД для объектов. Независимая от языка программирования сериализация и XML. Создание документов XML и обработка готовых документов средствами Python

21.Модули и пакеты. Библиотеки сторонних разработчиков.

22.Взаимодействие Python с Интернетом. Структура и функционирование сети Интернет. Архитектура клиент-сервер. Пакетная передача данных

23.Математическая статистика на Python – основные методы

24.Построение графиков на Python – библиотека Mathplotlib

25.Нейронные сети на Python. Библиотека keras.

26. Словари. Создание словаря. Операции над словарями.

27. Методы для работы над словарями. Генераторы словарей.

28. Определение функции и ее вызов. Необязательные параметры функций и сопоставление по ключам.

29. Переменное число параметров в функции.

30. Анонимные функции.

31. Функции-генераторы.

32. Декораторы функций.

33. Глобальные и локальные переменные.

34. Рекурсивные функции. Вложенные функции.

35. Модули. Понятие модуля. Подключение модуля: инструкции import и from.

36. Пути поиска модулей. Повторная загрузка модулей.

37. Пакеты. Понятие пакета. Работа с пакетами.

38. Понятие класса, атрибута и метода.

39. Определение класса и создание экземпляра класса. Конструкторы и деструкторы.

40. Наследование. Множественное наследование.

41. Понятие примесей и их использование.

42. Специальные методы классов. Перегрузка операторов.

43. Статические методы и методы классов.

44. Абстрактные методы.

45. Ограничение доступа к идентификаторам внутри класса.

46. Свойства классов. Декораторы классов.

7.2.6. Методика выставления оценки при проведении промежуточной аттестации

Зачет с оценкой проводится по тест-билетам, каждый из которых содержит 10 вопросов и задачу. Каждый правильный ответ на вопрос в тесте оценивается 1 баллом, задача оценивается в 10 баллов (5 баллов верное решение и 5 баллов за верный ответ). Максимальное количество набранных баллов – 20.

1. Оценка «Неудовлетворительно» ставится в случае, если студент набрал менее 6 баллов.

2. Оценка «Удовлетворительно» ставится в случае, если студент набрал от 6 до 10 баллов

3. Оценка «Хорошо» ставится в случае, если студент набрал от 11 до 15 баллов.

4. Оценка «Отлично» ставится, если студент набрал от 16 до 20 баллов.

7.2.7 Паспорт оценочных материалов

№ п/п	Контролируемые разделы (темы) дисциплины	Код контролируемой компетенции	Наименование оценочного средства
1	Введение в Python	ОПК-6, ОПК-8, ОПК-10	Тест, контрольная работа, защита лабораторных работ, защита реферата, требования к курсовому проекту….
2	Условные операторы функции	ОПК-6, ОПК-8, ОПК-10	Тест, контрольная работа, защита лабораторных работ, защита реферата, требования к курсовому проекту….
3	Работа со строками	ОПК-6, ОПК-8, ОПК-10	Тест, контрольная работа, защита лабораторных работ, защита реферата, требования к курсовому проекту….
4	Программирование разветвляющихся и циклических процессов на языке Python	ОПК-6, ОПК-8, ОПК-10	Тест, контрольная работа, защита лабораторных работ, защита реферата, требования к курсовому

			проекту….
5	Массивы	ОПК-6, ОПК-8, ОПК-10	Тест, контрольная работа, защита лабораторных работ, защита реферата, требования к курсовому проекту….
6	Объектно-ориентированное программирование на языке Python	ОПК-6, ОПК-8, ОПК-10	Тест, контрольная работа, защита лабораторных работ, защита реферата, требования к курсовому проекту….

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

Тестирование осуществляется, либо при помощи компьютерной системы тестирования, либо с использованием выданных тест-заданий на бумажном носителе. Время тестирования 30 мин. Затем осуществляется проверка теста экзаменатором и выставляется оценка согласно методики выставления оценки при проведении промежуточной аттестации.

Решение стандартных задач осуществляется, либо при помощи компьютерной системы тестирования, либо с использованием выданных задач на бумажном носителе. Время решения задач 30 мин. Затем осуществляется проверка решения задач экзаменатором и выставляется оценка, согласно методики выставления оценки при проведении промежуточной аттестации.

Решение прикладных задач осуществляется, либо при помощи компьютерной системы тестирования, либо с использованием выданных задач на бумажном носителе. Время решения задач 30 мин. Затем осуществляется проверка решения задач экзаменатором и выставляется оценка, согласно методики выставления оценки при проведении промежуточной аттестации.

Защита курсовой работы, курсового проекта или отчета по всем видам практик осуществляется согласно требованиям, предъявляемым к работе, описанным в методических материалах. Примерное время защиты на одного студента составляет 20 мин.

8 УЧЕБНО МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ)

8.1 Перечень учебной литературы, необходимой для освоения дисциплины

1. Никульчев, Е. В. Системы сбора и предобработки данных. Методы статистического анализа с использованием Google Colab: учебное пособие / Е. В. Никульчев, А. С. Алексеенко, Д. Ю. Ильин. — Москва: РТУ МИРЭА, 2023. — 121 с. — ISBN 978-5-7339-1948-5. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/382739 (дата обращения: 00.00.0000). — Режим доступа: для авториз. пользователей.

2. Кочетыгов, А. А. Основы программирования на языке Python: учебное пособие / А. А. Кочетыгов. — Тула: ТулГУ, 2024. — 272 с. — ISBN 978-5-7679-5380-6. — Текст: электронный // Лань: электронно-библиотечная

система. — URL: https://e.lanbook.com/book/427316 (дата обращения: 00.00.0000). — Режим доступа: для авториз. пользователей.

3. Косицин, Д. Ю. Язык программирования Python: учебно-методическое пособие / Д. Ю. Косицин. — Минск: БГУ, 2019. — 136 с. — ISBN 978-985-566-746-0. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/180546 (дата обращения: 00.00.0000). — Режим доступа: для авториз. пользователей.

4. Язык программирования PYTHON: учебно-методическое пособие / составители М. А. Артемов [и др.]. — Воронеж: ВГУ, 2021 — Часть 1 — 2021. — 93 с. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/454721 (дата обращения: 00.00.0000). — Режим доступа: для авториз. пользователей.

5. Барулина, М. А. Основы языка программирования Python для применения в математическом моделировании и обработке медицинских данных: учебно-методическое пособие / М. А. Барулина. — Самара:, 2024. — 77 с. — Текст: электронный // Лань: электронно-библиотечная система. — URL: https://e.lanbook.com/book/429698 (дата обращения: 00.00.0000). — Режим доступа: для авториз. пользователей.

6. Широбокова, С. Н. Программирование на языке Python для лабораторных занятий: учебное пособие / С. Н. Широбокова, А. А. Кацупеев, А. В. Сулыз. — Новочеркасск: ЮРГПУ, 2020. — 104 с. — ISBN 978-5-9997-0725-3. — Текст: электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/180938 (дата обращения: 00.00.0000). — Режим доступа: для авториз. пользователей.

7. Прикладная информатика. Магистратура. Учебный комплект для студентов магистратуры по направлению подготовки 09.04.03 «Прикладная информатика» всех форм обучения: учебно-методическое пособие: в 2 частях / составители С. И. Сенашов [и др.] ; под редакцией С. И. Сенашова. — Красноярск: СибГУ им. академика М. Ф. Решетнёва, 2023 — Часть 1 — 2023. — 96 с. — Текст: электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/400598 (дата обращения: 00.00.0000). — Режим доступа: для авториз. пользователей.

8.2 Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень лицензионного программного обеспечения, ресурсов информационно-телекоммуникационной сети «Интернет», современных профессиональных баз данных и информационных справочных систем:

1. Справочная правовая система КонсультантПлюс http://www.consultant.ru/
2. Межвузовская электронная библиотека (МЭБ) https://icdlib.nspu.ru/
3. Национальная электронная библиотека https://rusneb.ru/
4. Adobe Acrobat Reader. reader.html?promoid=81G55Y1C&mv=other). (https://acrobat.adobe.com/us/en/acrobat/pdf2.
5. Бесплатная интегрированная среда разработки Anaconda.

6. Система электронного обучения https://elearning.utmn.ru/

9 МАТЕРИАЛЬНО-ТЕХНИЧЕСКАЯ БАЗА, НЕОБХОДИМАЯ ДЛЯ ОСУЩЕСТВЛЕНИЯ ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА

Компьютерный класс 2303 в составе:

- Рабочие станции –10 комплектов;
- Принтер лазерный -1 комплект;
- Комплект сетевого оборудования для организации ЛВС и доступа к ресурсам сети ВГТУ (в том числе к нейрокомпьютеру);
- Мультимедиапроектор и экран;
- Программы: Google Colab, PyCharm, postgreSQL.

Автоматизированные обучающие системы для изучения прикладных программных продуктов, тестирующий комплекс контроля качества обучения, интегрированная система мониторинга хода учебного процесса кафедры.

10. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ (МОДУЛЯ)

По дисциплине «Алгоритмы и структуры данных (продвинутый Python)» читаются лекции, проводятся лабораторные работы, выполняется курсовой проект.

Основой изучения дисциплины являются лекции, на которых излагаются наиболее существенные и трудные вопросы, а также вопросы, не нашедшие отражения в учебной литературе.

Лабораторные работы выполняются на лабораторном оборудовании в соответствии с методиками, приведенными в указаниях к выполнению работ.

Методика выполнения курсового проекта изложена в учебно-методическом пособии. Выполнять этапы курсового проекта должны своевременно и в установленные сроки.

Контроль усвоения материала дисциплины производится проверкой курсового проекта, защитой курсового проекта.

Вид учебных занятий	Деятельность студента
Лекция	Написание конспекта лекций: кратко, схематично, последовательно фиксировать основные положения, выводы, формулировки, обобщения; помечать важные мысли, выделять ключевые слова, термины. Проверка терминов, понятий с помощью энциклопедий, словарей, справочников с выписыванием толкований в тетрадь. Обозначение вопросов, терминов, материала, которые вызывают трудности, поиск ответов в рекомендуемой литературе. Если самостоятельно не удается разобраться в материале, необходимо сформулировать вопрос и задать преподавателю на лекции или на практическом занятии.
Лабораторная работа	Лабораторные работы позволяют научиться применять

	теоретические знания, полученные на лекции при решении конкретных задач. Чтобы наиболее рационально и полно использовать все возможности лабораторных для подготовки к ним необходимо: следует разобрать лекцию по соответствующей теме, ознакомится с соответствующим разделом учебника, проработать дополнительную литературу и источники, решить задачи и выполнить другие письменные задания.
Самостоятельная работа	Самостоятельная работа студентов способствует глубокому усвоения учебного материала и развитию навыков самообразования. Самостоятельная работа предполагает следующие составляющие: - работа с текстами: учебниками, справочниками, дополнительной литературой, а также проработка конспектов лекций; - выполнение домашних заданий и расчетов; - работа над темами для самостоятельного изучения; - участие в работе студенческих научных конференций, олимпиад; - подготовка к промежуточной аттестации.
Подготовка к промежуточной аттестации	Готовиться к промежуточной аттестации следует систематически, в течение всего семестра. Интенсивная подготовка должна начаться не позднее, чем за месяц-полтора до промежуточной аттестации. Данные перед зачетом с оценкой три дня эффективнее всего использовать для повторения и систематизации материала.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

№ п/п	Перечень вносимых изменений	Дата внесения изменений	Подпись заведующего кафедрой, ответственной за реализацию ОПОП