

Федеральное государственное бюджетное образовательное учреждение высшего образования «Тамбовский государственный университет имени Г.Р. Державина»
Институт математики, физики и информационных технологий
Кафедра математического моделирования и информационных технологий

УТВЕРЖДАЮ:
Директор института



Н. Л. Королева
«05» июля 2021 г.

РАБОЧАЯ ПРОГРАММА

по дисциплине Б1.О.11 Программирование на Python

Направление подготовки/специальность: 09.03.03 - Прикладная информатика

Профиль/направленность/специализация: Прикладная информатика в
информационной сфере

Уровень высшего образования: бакалавриат

Квалификация: Бакалавр

год набора: 2021

Автор программы:

Кандидат педагогических наук, Скворцов Александр Александрович

Рабочая программа составлена в соответствии с ФГОС ВО по направлению подготовки 09.03.03 - Прикладная информатика (уровень бакалавриата) (приказ Министерства образования и науки РФ от «19» сентября 2017 г. № 922).

Рабочая программа принята на заседании Кафедры математического моделирования и информационных технологий «18» мая 2021 г. Протокол № 9

Рассмотрена и одобрена на заседании Ученого совета Института математики, физики и информационных технологий, Протокол от «05» июля 2021 г. № 5.

СОДЕРЖАНИЕ

1. Цели и задачи дисциплины.....	4
2. Место дисциплины в структуре ОП бакалавра.....	4
3. Объем и содержание дисциплины.....	4
4. Контроль знаний обучающихся и типовые оценочные средства.....	33
5. Методические указания для обучающихся по освоению дисциплины (модуля).....	75
6. Учебно-методическое и информационное обеспечение дисциплины.....	77
7. Материально-техническое обеспечение дисциплины, программное обеспечение, профессиональные базы данных и информационные справочные системы.....	78

1. Цели и задачи дисциплины

1.1 Цель дисциплины – формирование компетенций:

ОПК-7 Способен разрабатывать алгоритмы и программы, пригодные для практического применения

1.2 Типы задач профессиональной деятельности, к которым готовятся обучающиеся в рамках освоения дисциплины:

- научно-исследовательский
- проектный

1.3 Дисциплина ориентирована на подготовку обучающихся к профессиональной деятельности в сфере: 06 Связь, информационные и коммуникационные технологии (в сфере проектирования, разработки, внедрения и эксплуатации информационных систем, управления их жизненным циклом)

1.4 В результате освоения дисциплины у обучающихся должны быть сформированы:

Обобщенные трудовые функции / трудовые функции / трудовые или профессиональные действия (при наличии профстандарта)	Код и наименование компетенции ФГОС ВО, необходимой для формирования трудового или профессионального действия	Индикаторы достижения компетенций
	ОПК-7 Способен разрабатывать алгоритмы и программы, пригодные для практического применения	Имеет адекватное представление об основных языках программирования, современных программных средах разработки информационных систем и технологий

1.5 Согласование междисциплинарных связей дисциплин, обеспечивающих освоение компетенций:

ОПК-7 Способен разрабатывать алгоритмы и программы, пригодные для практического применения

№ п/п	Наименование дисциплин, определяющих междисциплинарные связи	Форма обучения									
		Очная (семестр)					Заочная (семестр)				
		1	2	3	6	8	1	2	3	6	9
1	Алгоритмизация и программирование	+	+				+	+			
2	Научно-исследовательская работа					+					+
3	Объектно-ориентированное программирование			+					+		
4	Ознакомительная практика				+					+	

2. Место дисциплины в структуре ОП бакалавриата:

Дисциплина «Программирование на Python» относится к обязательной части учебного плана ОП по направлению подготовки 09.03.03 - Прикладная информатика.

Дисциплина «Программирование на Python» изучается в 2 семестре.

3. Объем и содержание дисциплины

3.1. Объем дисциплины: 4 з.е.

Очная: 4 з.е.

Заочная: 4 з.е.

Вид учебной работы	Очная (всего часов)	Заочная (всего часов)
Общая трудоёмкость дисциплины	144	144
Контактная работа	64	12
Лекции (Лекции)	16	4
Лабораторные (Лаб. раб.)	48	8
Самостоятельная работа (СР)	44	123
Экзамен	36	9

3.2.Содержание курса:

№ темы	Название раздела/темы	Вид учебной работы, час.						Формы текущего контроля
		Лекции		Лаб. раб.		СР		
		О	З	О	З	О	З	
2 семестр								
1	Переменные. Типы данных. Преобразование типов данных	1	1	4	2	4	8	Собеседование; Тестирование
2	Условные операторы и циклы	1	1	2	-	2	7	Собеседование; Тестирование
3	Строки и двоичные данные	1	-	2	-	2	7	Собеседование; Тестирование
4	Функции и методы для работы со строками	1	-	2	-	2	6	Собеседование; Тестирование
5	Списки. Операции над списками	1	-	2	-	2	6	Собеседование; Тестирование
6	Кортежи, множества и диапазоны.	1	-	2	-	2	6	Собеседование; Тестирование
7	Словари. Операции и методы для работы со словарями	1	-	2	-	2	6	Собеседование; Тестирование
8	Работа с датой и временем	1	-	2	-	2	6	Собеседование; Тестирование
9	Пользовательские функции	1	-	2	-	2	7	Собеседование; Тестирование
10	Обработка исключений	1	-	2	-	2	6	Собеседование; Тестирование
11	Работа с файлами и каталогами	1	-	4	-	2	8	Собеседование; Тестирование
12	Доступ из Python к базам данных SQLite	1	1	4	2	4	8	Собеседование; Тестирование
13	Доступ из Python к базам данных MySQL	1	1	4	2	4	8	Собеседование; Тестирование

14	Работа с графикой	-	-	2	-	2	6	Собеседование; Тестирование
15	Определение класса и создание экземпляра класса	1	-	2	-	2	6	Собеседование; Тестирование
16	Принципы объектно-ориентированного программирования	1	-	2	-	4	6	Собеседование; Тестирование
17	Статические методы и методы класса	-	-	4	-	2	8	Собеседование; Тестирование
18	Основы разработки оконных приложений	1	-	4	2	2	8	Тестирование; Собеседование

Тема 1. Переменные. Типы данных. Преобразование типов данных (ОПК-7)

Лекция.

Введение в Python, возможности Python, синтаксис Python, переменные, типы данных, преобразование типов данных.

Лабораторные работы.

1. Произвести сложение и вычитание a и b. Результат вычислений сохранить в переменных c и d.
2. Исправить, приведенный ниже, блок кода.

```
a:=2
b-a=c
a+b:=c
```

3. Отредактируйте код, что бы он выводил заданный текст.

```
# данный код
a, b = 45, 54
c = a + 1
d = c +
print(d)
# требуемый вывод:
# 56
```

4. Допишите код, чтобы получить требуемый вывод.

```
# данный код
a, b = 45, 54
c = a + 1
d = c +
print(d)
# требуемый вывод:
# 56
```

5. Создайте переменные, чтобы вывести требуемый текст.

```
# данный код
print("Мой стек:", programming_lang_1, programming_lang_2, programming_lang_3)
# требуемый вывод:
# Мой стек: python, javascript, php
```

Задания для самостоятельной работы.

1. Дано целое десятичное число. Выведите его последнюю цифру.

2. Дано целое десятичное число. Найдите число десятков в его десятичной записи.
3. Дано трехзначное число. Найдите сумму его цифр.
4. Пирожок в столовой стоит a рублей и b копеек. Определите, сколько рублей и копеек нужно заплатить за n пирожков.
5. В школе решили набрать три новых математических класса. Так как занятия по математике у них проходят в одно и то же время, было решено выделить кабинет для каждого класса и купить в них новые парты. За каждой партой может сидеть не больше двух учеников. Известно количество учащихся в каждом из трёх классов. Сколько всего нужно закупить парт чтобы их хватило на всех учеников? Программа получает на вход три целых десятичных числа:
количество учащихся в каждом из трех классов.
6. Доработайте код задачи № 3 таким образом, чтобы он запрашивал время начала занятий (минуты и часы отдельно) и номер урока,
а далее также рассчитывал время окончания уроков.
7. Пользователь вводит число и систему счисления этого числа. Программа переводит число в десятичную, двоичную, восьмеричную и шестнадцатеричную системы счисления с использованием стандартных функций.

Тема 2. Условные операторы и циклы (ОПК-7)

Лекция.

Операторы в Python, циклы в python.

Лабораторные работы.

1. Занятия в школе начинаются в 8-30. Урок длится 45 минут, перерывы между уроками – 10 минут. Ввести номер урока и вывести время его окончания.
2. Ввести число, обозначающее размер одной фотографии в Мбайтах. Определить, сколько фотографий поместится на флэш-карту объёмом 2 Гбайта.
3. Разведчики-математики для того, чтобы опознать своих, используют числовые пароли. Услышав число-пароль, разведчик должен возвести его в квадрат и сказать в ответ первую цифру дробной части полученного числа.
Напишите программу, которая по полученному паролю (вещественному числу) вычисляет число-ответ.
4. В игре «Русское лото» из мешка случайным образом выбираются бочонки, на каждом из которых написано число от 1 до 90. Напишите программу, которая выводит наугад первые 5 выигрышных номеров.
5. Напишите программу, которая получает возраст человека (целое число, не превышающее 120) и выводит этот возраст со словом «год», «года» или «лет». Например, «21 год», «22 года», «25 лет».
6. Напишите программу, которая получает с клавиатуры натуральное число и определяет, сколько раз в его десятичной записи встречается цифра 1
7. Напишите программу, которая получает с клавиатуры натуральное число и определяет, есть ли в его десятичной записи одинаковые цифры, стоящие рядом.

Задания для самостоятельной работы.

1. Напишите программу, которая получает с клавиатуры два целых числа и вычисляет их произведение,
используя только операции сложения.
2. Напишите программу, которая получает с клавиатуры натуральное число и вычисляет целый квадратный корень из него – наибольшее число, квадрат которого не больше данного числа.
3. Натуральное число называется числом Армстронга, если сумма цифр числа,

возведенных в N-ную степень (где N – количество цифр в числе) равна самому числу.

4. Вывести на экран циклом пять строк из нулей, причем каждая строка должна быть пронумерована;
5. Найти сумму ряда чисел от 1 до 100. Полученный результат вывести на экран;
6. Даны три числа. Вывести на экран «yes», если среди них есть одинаковые, иначе вывести «ERROR»;
7. Даны три числа. Вывести на экран «yes», если можно взять какие-то два из них и в сумме получить третье;
8. Дано три числа. Найти количество положительных чисел среди них;
9. Вывести на экран все чётные значения в диапазоне от 1 до 497;
10. Написать программу, которая будет складывать, вычитать, умножать или делить два числа. Числа и знак операции вводятся пользователем. После выполнения вычисления

Тема 3. Строки и двоичные данные (ОПК-7)

Лекция.

Строки в Python, двоичные данные в Python

Лабораторные работы.

1. Дана строка, состоящая из слов, разделенных пробелами. Определите, сколько в ней слов. Используйте для решения задачи метод count.
2. Дана строка. Разрежьте ее на две равные части (если длина строки — четная, а если длина строки нечетная, то длина первой части должна быть на один символ больше). Переставьте эти две части местами, результат запишите в новую строку и выведите на экран.
3. Дана строка, состоящая ровно из двух слов, разделенных пробелом. Переставьте эти слова местами. Результат запишите в строку и выведите получившуюся строку.
4. Дана строка, в которой буква h встречается как минимум два раза. Разверните последовательность символов, заключенную между первым и последним появлением буквы h, в противоположном порядке.
5. Дана строка. Замените в этой строке все цифры 1 на слово one.

Задания для самостоятельной работы.

1. Строки в python обозначаются кавычками. Приведите все способы.
2. Какие типы данных можно преобразовать в строку?
3. Опишите синтаксис срезов строк при помощи квадратных скобок.
4. Как применяют операции сложения и умножения к строкам?
5. Что такое двоичные данные?

Тема 4. Функции и методы для работы со строками (ОПК-7)

Лекция.

Функции в Python, методы работы со строками в Python.

Лабораторные работы.

- 1) Что будет выведено в результате выполнения следующих строк кода?

```
String = "hello" + ","
```

```
String = string + "World!"
```

```
Print(string)
```

```
1)string
```

```
2)Hello, World!
```

```
3)Hello+, +World
```

```
4)World
```

Ответ:

- 2) Что возвращает функция len() при передаче в неё строки?

- 1)Размер строки в байтах
- 2)Количество символов в строке
- 3)Количество слов
- 4)Данная функция не предназначена для строк

Ответ:

- 3) Что делает следующая программа?

```
string = "explain your opinion"
```

```
var= string.split()
```

- 1)Разделяет предложение на символы
- 2)Удаляет пробелы из предложения
- 3)Делает все заглавные символы прописными
- 4)Разделяет предложение на слова

Ответ:

- 4) Сколько символов будет содержать строка str1 после запуска программы?

```
Str1 = "abc bc bcaa"
```

```
Str1 = Str1.replace("a", "bc")
```

```
Str1 = Str1.replace("bc", "1")
```

- 1)1
- 2)7
- 3)8
- 4)11

Ответ:

- 5) Сколько символов будет содержать строка str1 после выполнения следующих строк кода?

```
Str1 = "good morning"
```

```
Str1 = str1[2:6] + str1[1:3]
```

- 1)5
- 2)6
- 3)8
- 4)2

Ответ:

Задания для самостоятельной работы.

1. Даны четыре действительных числа: x1, y1, x2, y2. Напишите функцию distance(x1, y1, x2, y2), вычисляющая расстояние между точкой (x1,y1) и (x2,y2).

Считайте четыре действительных числа и выведите результат работы этой функции.

2. Дано действительное положительное число a и целое число n. Вычислите a^n .
3. Напишите функцию search_substr(subst, st), которая принимает 2 строки и определяет, имеется ли подстрока subst в строке st. В случае нахождения подстроки, возвращается фраза «Есть контакт!», а иначе «Мимо!».

Должно быть найдено совпадение независимо от регистра обеих строк.

Тема 5. Списки. Операции над списками (ОПК-7)

Лекция.

Списки в Python и Операции над ними.

Лабораторные работы.

1. Выведите все элементы списка с четными индексами (то есть A[0], A[2], A[4], ...).
2. Дан список чисел. Выведите все элементы списка, которые больше предыдущего элемента.
3. Дан список чисел. Определите, сколько в этом списке элементов, которые больше двух своих соседей,

и выведите количество таких элементов. Крайние элементы списка никогда не учитываются, поскольку у них недостаточно соседей.

4. Дан список, упорядоченный по неубыванию элементов в нем. Определите, сколько в нем различных элементов.

5. В списке все элементы различны. Поменяйте местами минимальный и максимальный элемент этого списка.

Задания для самостоятельной работы.

1) Как создать список?

1) Все варианты верны

2) `L = list(1, 2, 3)`

3) `l = [1, 2, 3]`

4) `l = list[1, 2, 3]`

Ответ:

2) Что выведет этот код:

```
a = [ 1, 342, 223, 'Африка', 'Очки']
```

```
print(a[-3])
```

1) 223

2) 'Африка'

3) 342

4) Ошибку

Ответ:

3) Что выведет этот код:

```
sample = [10, 20, 30]
```

```
sample.append(60)
```

```
sample.insert(3, 40)
```

```
print(sample)
```

1) [10, 20, 30, 40]

2) [10, 20, 30, 40, 60]

3) [10, 20, 30, 60, 40]

4) [60, 10, 20, 30, 40]

Ответ:

4) Что из перечисленного правда?

1) Элементы списка не могут повторяться

2) Все элементы списка должны быть одного типа

3) Мы можем вставить элемент на любую позицию в списке

4) Список не может содержать вложенных списков

Ответ:

5) Как получить ['bar', 'baz'] из списка

```
a = ['foo', 'bar', 'baz', 'qux', 'quux']
```

?

1) `print(a[2:4])`

2) `print(a[1], a[2])`

3) `print(a[1:-2])`

4) `print(a[-4:-3])`

5) `print(a[2:3])`

Ответ:

Кортежи в Python, множества и диапазоны.

Лабораторные работы.

1. Напишите функцию `tpl_sort()`, которая сортирует кортеж, состоящий из целых чисел по возрастанию и возвращает его.

Если хотя бы один элемент не является целым числом, то функция возвращает исходный кортеж.

2. Перед студентом стоит задача: на вход функции `sieve()` поступает список целых чисел.

В результате выполнения этой функции будет получен кортеж уникальных элементов списка в обратном порядке.

3. Дан текст: в первой строке записано число строк, далее идут сами строки. Определите, сколько различных слов содержится в этом тексте.

4. Во входной строке записана последовательность чисел через пробел. Для каждого числа выведите слово YES (в отдельной строке), если это число ранее встречалось в последовательности или NO, если не встречалось.

5. Даны два списка чисел. Найдите все числа, которые входят как в первый, так и во второй список и выведите их в порядке возрастания.

Задания для самостоятельной работы.

1) Что такое множество в Python?

1) Это любая коллекция элементов

2) Это список, содержащий в себе только функции

3) Это контейнер, значения в котором не повторяются

4) Это список, содержащий вложенные списки в себе

Ответ:

2) Каким образом правильно объявляется множество?

1) `a = {}`

2) `a = []`

3) `a = set()`

4) `a = set`

Ответ:

3) Чем отличаются методы `remove()` и `discard()`, применяемые к множеству?

1) Ничем

2) `remove()` удаляет элемент если он есть, но бросает ошибку если элемента нет. `discard()` просто удаляет элемент если он есть

3) `discard()` удаляет элемент если он есть, но бросает ошибку если элемента нет. `remove()` просто удаляет элемент если он есть

4) Метода `discard()` для множеств не существует

Ответ:

4) Что такое `frozenset`?

1) Множество, которое нельзя изменять

2) Множество, которое хранит в себе только неизменяемые объекты

3) Множество, которое используется для хранения констант

4) Выдумка нашей редакции

Ответ:

5) Что это за метод такой, `set.difference(another_set)`

1) Возвращает True, если два множества одинаковые, False если хотя бы один элемент не совпадает

2) Возвращает True, если два множества разные, False если хотя бы один элемент совпадает

3) Возвращает множество из элементов, которые встречаются только в множестве `set`

4) Возвращает множество из элементов, которые встречаются только в множестве `another_set`

Ответ:

Лекция.

Словари в Python, операции и методы словарей

Лабораторные работы.

1. В единственной строке записан текст. Для каждого слова из данного текста подсчитайте, сколько раз оно встречалось в этом тексте ранее.

2. Дан текст: в первой строке задано число строк, далее идут сами строки. Выведите слово, которое в этом тексте чаще встречается.

Если таких слов несколько, выведите то, которое меньше в лексикографическом порядке.

3. Дан список стран и городов каждой страны. Затем даны названия городов. Для каждого города укажите, в какой стране он находится.

4. Во входных данных записано дерево в том же формате, что и в предыдущей задаче. Далее идет число запросов K. В каждой из следующих K строк, содержатся имена двух элементов дерева.

Для каждого такого запроса выведите одно из трех чисел: 1, если первый элемент является предком второго, 2, если второй является предком первого или 0,

если ни один из них не является предком другого.

5. В генеалогическом древе определите для двух элементов их наименьшего общего предка (Lowest Common Ancestor). Наименьшим общим предком элементов A и B является такой элемент C,

что C является предком A, C является предком B, при этом глубина C является наибольшей из возможных. При этом элемент считается своим собственным предком. Для каждого запроса выведите

наименьшего общего предка данных элементов.

Задания для самостоятельной работы.

1) Что означает ошибка `TypeError: unhashable type?`

1) Неверно задано значение

2) Тип данных нельзя использовать в роли ключа

3) Слишком большое значение

4) Ошибка синтаксиса

Ответ:

2) Какие типы данных нельзя использовать в роли ключа?

1) Список, словарь

2) Словарь, кортеж

3) Кортеж, число

4) Число, булево значение

Ответ:

3) Что выдаст этот код?

```
Another_dict = {'a': {'a': ['a']}}
```

```
Print(another_dict.pop('a') == another_dict.clear())
```

1) True

2) False

3) Ошибка

Ответ:

4) Каков будет результат кода?

```
Dict_1 = {'a': 10, 'b': 20}
```

```
Dict_2 = {'b': 20, 'c': 30}
```

```
Dict_1.update(dict_2)
```

```
Print(dict_1)
```

1) {'a': 10}

2) {'a': 10, 'b': 20}

3) {'a': 10, 'b': 20, 'c': 30}

4) {'b': 20, 'c': 30}

Ответ:

5) Что выведет этот код?

```
Old_dict = {'a': 10, 'b': 10}
```

```
New_dict = {}
```

```
For i, j in old_dict.items():
```

```
New_dict[j] = i
```

```
Print(new_dict)
```

1) {'a': 10, 'b': 10}

2) {10: 'a', 10: 'b'}

3) {'a': 10}

4) {10: 'b'}

Ответ:

Тема 8. Работа с датой и временем (ОПК-7)

Лекция.

Дата и время в Python.

Лабораторные работы.

1. По заданному числу n от 1 до 365 определите, на какое число какого месяца приходится день невисокосного года с номером n .

Программа получает на вход целое число n и должна вывести два числа: число месяца (от 1 до 31) и номер месяца (от 1 до 12), на которое приходится данный день.

2. Часы показывают время в формате hh:mm:ss. Определите количество секунд, которое прошло с начала суток.

Программа не должна содержать циклов для решения этой задачи.

3. Часы показывают время в формате hh:mm:ss. На этих часах запустили таймер, который прозвонит через n секунд. Определите время, которое будет на часах, когда прозвонит таймер.

n может принимать значения от 0 до 109. Решение задачи не должно содержать циклов. Постарайтесь также не использовать условную инструкцию.

4. В часах села батарейка и они стали идти вдвое медленней. Когда на часах было время $h1:m1:s1$, точное время было $h2:m2:s2$. Определите,

точное время, когда часы в следующий раз покажут время $h3:m3:s3$.

Решение задачи не должно использовать циклы.

Задания для самостоятельной работы.

1. Как получить текущую дату и время в Python?

2. Как получить текущую дату?

3. Что внутри datetime?

4. Как вывести час, минуту, секунду и микросекунду?

Тема 9. Пользовательские функции (ОПК-7)

Лекция.

Функции в Python.

Лабораторные работы.

1. Реализовать функцию `is_sorted`. Функция принимает на вход список и возвращает `True`, если элементы в нем упорядочены по возрастанию.

2. Реализовать функцию `find_longest`. Функция принимает на вход список строк и возвращает строку с максимальной длиной. Если таких строк несколько, то возвращает первую из них.

3. Реализовать функцию `min_max`. Функция принимает на вход список чисел и возвращает пару: минимальное и максимальное число. Алгоритм должен выполнять не более одного прохода по списку.

4.Реализовать функцию `zip_longest_sum`. Функция принимает на вход два списка и возвращает новый список, полученный сложением соответствующих элементов входных списков.

5.Реализовать функцию `is_prime`. Функция принимает на вход целое число и возвращает `True`, если число простое. Простым называется число, которое делится без остатка только на 1 и на себя.

6.Реализовать функцию `join`. Функция принимает разделитель и список строк, возвращает строку, в которой элементы списка чередуются с разделителем.

Задания для самостоятельной работы.

1) Как можно вызвать метод `func` у следующего класса (выберите все подходящие варианты):

```
Class myclass:
```

```
Def func(self):
```

```
Print('hello')
```

- 1) `myClass.func()`
- 2) `obj = myClass() obj.func()`
- 3) `obj = myClass() myClass.func(obj)`
- 4) `obj = myClass() obj.func`
- 5) ни один из перечисленных

Ответ:

2) Что напечатает следующий код:

```
def func(n):
```

```
    n = n + 1
```

```
print(func(0))
```

- 1)0
- 2)1
- 3)`func(0)`
- 4)`None`
- 5)возникнет ошибка

Ответ:

3) Что выведет следующий код:

```
a = 3
```

```
a = "foo" if a / 2 == 1 else 2
```

```
a = a + a
```

```
print (a)
```

- 1)6
- 2)Возникнет ошибка
- 3)2
- 4)4
- 5)foofoo

Ответ:

4) Что необходимо добавить на место пропущенной строки?

```
def find_max(nums):
```

```
    max_num = float("-inf")
```

```
    for num in nums:
```

```
        if num > max_num:
```

```
            # пропущенная строка
```

```
    return max_num
```

- 1)`max_num = num`
- 2)`num = max_num`
- 3)`max_num += 1`
- 4)`max_num += num`

Ответ:

5) Каким будет результат выполнения кода:

```
a = [1, 2, 3]
```

```
if a[2] < 3:
```

```
print(a[a[1]])
```

```
else:
```

```
print(a[1])
```

1)1

2)2

3)3

4)возникнет ошибка

Ответ:

Тема 10. Обработка исключений (ОПК-7)

Лекция.

Обработка исключений в Python.

Лабораторные работы.

1. Написать программу, которая будет писать в консоль названия нажатых клавиш. Реализовать поддержку enter, space, w, a, s, d, esc.
2. С помощью циклов, используя квадрат 10x10 пикселей и его след, нарисовать рамку 100x100 пикселей.
3. Написать программу, в которой по нажатию клавиш будет двигаться квадрат размером 20x20 пикселей. Учесть, что квадрат не должен выходить за границы экрана.
4. Доработать программу, чтобы квадрат при каждом движении менял свой цвет на случайный.

Задания для самостоятельной работы.

1. Что означает исключение в python?
2. Какие стандартные исключения в Python вы знаете?
3. Блок множественного исключения

Тема 11. Работа с файлами и каталогами (ОПК-7)

Лекция.

Работа с файлами и каталогами в Python.

Лабораторные работы.

1. В текстовый файл построчно записаны фамилия и имя учащихся класса и его оценка за контрольную. Вывести на экран всех учащихся, чья оценка меньше 3 баллов и посчитать средний балл по классу.
2. В текстовом файле посчитать количество строк, а также для каждой отдельной строки определить количество в ней символов и слов.
3. Создать текстовый файл, записать в него построчно данные, которые вводит пользователь. Окончанием ввода пусть служит пустая строка.

Задания для самостоятельной работы.

1. Разработать приложение, которое записывает в файл все строки, введенные пользователем. Признак конца ввода — пустая строка.

После выполнения программы должен появиться файл data.txt, содержащий три строки:

2. Разработать приложение для нумерации строк в файле. Приложение принимает на вход имя файла и выводит его построчно, добавляя к каждой строке ее номер.

Если использовать файл, созданный в предыдущей задаче. Введите имя файла: data.txt

3. Разработать приложение для разделения файла на части. Приложение принимает на вход имя файла для разделения и целое число N.

На выходе у приложения множество файлов, каждый из которых содержит не более, чем N строк из исходного файла.

4. Разработать приложение для объединения файлов. Приложение принимает на вход имена файлов для объединения

(можно использовать файлы, полученные из предыдущего задания) и имя выходного файла.

Тема 12. Доступ из Python к базам данных SQLite (ОПК-7)

Лекция.

Доступ из Python к базам данных SQLite.

Лабораторные работы.

Соединение с базой, получение курсора

Для начала рассмотрим самый базовый шаблон DB-API, который будем использовать во всех дальнейших примерах:

```
# Импортируем библиотеку, соответствующую типу нашей базы данных import sqlite3 # Создаем
соединение с нашей базой данных # В нашем примере у нас это просто файл базы conn =
sqlite3.connect('Chinook_Sqlite.sqlite') # Создаем курсор - это специальный объект который делает
запросы и получает их результаты cursor = conn.cursor() # ТУТ БУДЕТ НАШ КОД РАБОТЫ С
БАЗОЙ ДАННЫХ # КОД ДАЛЬНЕЙШИХ ПРИМЕРОВ ВСТАВЛЯТЬ В ЭТО МЕСТО # Не
забываем закрыть соединение с базой данных conn.close()
```

Задания для самостоятельной работы.

2. Подключение к базам данных

Прежде чем взаимодействовать с любой базой данных через SQL-библиотеку, с ней необходимо связаться. В этом разделе мы рассмотрим, как подключиться из приложения Python к базам данных SQLite, MySQL и PostgreSQL. Рекомендуем сделать собственный .py файл для каждой из трёх баз данных.

Примечание. Для выполнения разделов о MySQL и PostgreSQL необходимо самостоятельно запустить соответствующие серверы. Для быстрого ознакомления с тем, как запустить сервер MySQL, ознакомьтесь с разделом MySQL в публикации [Запуск проекта Django \(англ.\)](#). Чтобы узнать, как создать базу данных в PostgreSQL, перейдите к разделу [Setting Up a Database](#) в публикации [Предотвращение атак SQL-инъекций с помощью Python \(англ.\)](#).

SQLite

SQLite, вероятно, является самой простой базой данных, к которой можно подключиться с помощью Python, поскольку для этого не требуется устанавливать какие-либо внешние модули. По умолчанию стандартная библиотека Python уже содержит модуль sqlite3.

Более того, SQLite база данных не требует сервера и самодостаточна, то есть просто читает и записывает данные в файл. Подключимся с помощью sqlite3 к базе данных:

```
import sqlite3 from sqlite3 import Error def create_connection(path): connection = None try:
connection = sqlite3.connect(path) print("Connection to SQLite DB successful") except Error as e:
print(f"The error '{e}' occurred") return connection
```

Вот как работает этот код:

- Строки 1 и 2 – импорт sqlite3 и класса Error.
- Строка 4 определяет функцию create_connection(), которая принимает путь к базе данных SQLite.
- Строка 7 использует метод connect() и принимает в качестве параметра путь к базе данных SQLite. Если база данных в указанном месте существует, будет установлено соединение. В противном случае по указанному пути будет создана новая база данных и так же установлено соединение.
- В строке 8 выводится состояние успешного подключения к базе данных.

- Строка 9 перехватывает любое исключение, которое может быть получено, если методу `.connect()` не удастся установить соединение.
- В строке 10 отображается сообщение об ошибке в консоли.

`sqlite3.connect(path)` возвращает объект `connection`. Этот объект может использоваться для выполнения запросов к базе данных SQLite. Следующий скрипт формирует соединение с базой данных SQLite:

```
connection = create_connection("E:\\sm_app.sqlite")
```

Выполнив вышеуказанный скрипт, вы увидите, как в корневом каталоге диска E появится файл базы данных `sm_app.sqlite`. Конечно, вы можете изменить местоположение в соответствии с вашими интересами.

2. Подключение к базам данных
Прежде чем взаимодействовать с любой базой данных через SQL-библиотеку, с ней необходимо связаться. В этом разделе мы рассмотрим, как подключиться из приложения Python к базам данных SQLite, MySQL и PostgreSQL. Рекомендуем сделать собственный `.py` файл для каждой из трёх баз данных.

Примечание. Для выполнения разделов о MySQL и PostgreSQL необходимо самостоятельно запустить соответствующие серверы. Для быстрого ознакомления с тем, как запустить сервер MySQL, ознакомьтесь с разделом MySQL в публикации [Запуск проекта Django \(англ.\)](#). Чтобы узнать, как создать базу данных в PostgreSQL, перейдите к разделу [Setting Up a Database](#) в публикации [Предотвращение атак SQL-инъекций с помощью Python \(англ.\)](#).

SQLite

SQLite, вероятно, является самой простой базой данных, к которой можно подключиться с помощью Python, поскольку для этого не требуется устанавливать какие-либо внешние модули. По умолчанию стандартная библиотека Python уже содержит модуль `sqlite3`.

Более того, SQLite база данных не требует сервера и самодостаточна, то есть просто читает и записывает данные в файл. Подключимся с помощью `sqlite3` к базе данных:

```
import sqlite3 from sqlite3 import Error def create_connection(path):    connection = None    try:
connection = sqlite3.connect(path)    print("Connection to SQLite DB successful")    except Error as e:
    print(f"The error '{e}' occurred")    return connection
```

Вот как работает этот код:

- Строки 1 и 2 – импорт `sqlite3` и класса `Error`.
- Строка 4 определяет функцию `create_connection()`, которая принимает путь к базе данных SQLite.
- Строка 7 использует метод `connect()` и принимает в качестве параметра путь к базе данных SQLite. Если база данных в указанном месте существует, будет установлено соединение. В противном случае по указанному пути будет создана новая база данных и так же установлено соединение.
- В строке 8 выводится состояние успешного подключения к базе данных.
- Строка 9 перехватывает любое исключение, которое может быть получено, если методу `.connect()` не удастся установить соединение.
- В строке 10 отображается сообщение об ошибке в консоли.

`sqlite3.connect(path)` возвращает объект `connection`. Этот объект может использоваться для выполнения запросов к базе данных SQLite. Следующий скрипт формирует соединение с базой данных SQLite:

```
connection = create_connection("E:\\sm_app.sqlite")
```

Выполнив вышеуказанный скрипт, вы увидите, как в корневом каталоге диска E появится файл базы данных `sm_app.sqlite`. Конечно, вы можете изменить местоположение в соответствии с вашими интересами.

Тема 13. Доступ из Python к базам данных MySQL (ОПК-7)

Лекция.

MySQL в Python

Лабораторные работы.

Создаем новую базу данных

Чтобы создать новую базу данных, например, с именем `online_movie_rating`, нужно выполнить инструкцию SQL:

```
CREATE DATABASE online_movie_rating;
```

Примечание

MySQL обязывает ставить точку с запятой (;) в конце оператора. Однако MySQL Connector/Python автоматически добавляет точку с запятой в конце каждого запроса.

Чтобы выполнить SQL-запрос, нам понадобится курсор, который абстрагирует процесс доступа к записям базы данных. MySQL Connector/Python предоставляет соответствующий класс `MySQLCursor`, экземпляр которого также называется курсором.

Передадим наш запрос о создании базы данных `online_movie_rating`:

```
try:    with connect(        host="localhost",        user=input("Имя пользователя: "),
password=getpass("Пароль: "),    ) as connection:        create_db_query = "CREATE DATABASE
online_movie_rating"        with connection.cursor() as cursor:        cursor.execute(create_db_query)
except Error as e:    print(e)
```

Запрос `CREATE DATABASE` сохраняется в виде строки в переменной `create_db_query`, а затем передается на выполнение в `cursor.execute()`.

Если база данных с таким именем уже существует на сервере, мы получим сообщение об ошибке. Используя тот же объект `MySQLConnection`, что и ранее, выполним запрос `SHOW DATABASES`, чтобы увидеть все таблицы, хранящиеся в базе данных:

```
try:    with connect(        host="localhost",        user=input("Введите имя пользователя: "),
password=getpass("Введите пароль: "),    ) as connection:        show_db_query = "SHOW DATABASES"
with connection.cursor() as cursor:        cursor.execute(show_db_query)        for db in cursor:
print(db) except Error as e:    print(e)
```

Введите имя пользователя: root Введите пароль: ('information_schema'), ('mysql'), ('online_movie_rating'), ('performance_schema'), ('sys'),

Приведенный код выведет имена всех баз данных, находящихся на нашем сервере MySQL. Команда `SHOW DATABASES` в нашем примере также вывела базы данных, которые автоматически создаются сервером MySQL и предоставляют доступ к метаданным баз данных и настройкам сервера.

Подключение к существующей базе данных

Итак, мы создали базу данных под названием `online_movie_rating`. Чтобы к ней подключиться, просто дополняем вызов `connect()` параметром `database`:

```
try:    with connect(        host="localhost",        user=input("Имя пользователя: "),
password=getpass("Пароль: "),        database="online_movie_rating",    ) as connection:
print(connection) except Error as e:    print(e)
```

Создание, изменение и удаление таблиц

В этом разделе мы рассмотрим, как с помощью Python выполнять некоторые базовые запросы: `CREATE TABLE`, `DROP` и `ALTER`.

Определение схемы базы данных

Начнем с создания схемы базы данных для рейтинговой системы фильмов. База данных будет состоять из трех таблиц:

1. `movies` — общая информация о фильмах:

- `id`
- `title`
- `release year`
- `genre`
- `collection_in_mi`

2. `reviewers` — информация о людях, опубликовавших оценки фильмов:

- 1 `id`

- 2 first_name
 - 3 last_name
3. ratings — информация об оценках фильмов рецензентами:
- 1 movie_id (foreign key)
 - 2 reviewer_id (foreign key)
 - 3 rating

Этих трех таблиц достаточно для целей данного руководства.

Задания для самостоятельной работы.

MySQL

В отличие от SQLite, в Python по умолчанию нет модуля, который можно использовать для подключения к базе данных MySQL. Для этого вам нужно установить драйвер Python для MySQL. Одним из таких драйверов является mysql-connector-python. Вы можете скачать этот модуль Python SQL с помощью pip:

```
pip install mysql-connector-python
```

Обратите внимание, что MySQL – это серверная система управления базами данных. Один сервер MySQL может хранить несколько баз данных. В отличие от SQLite, где соединение равносильно порождению БД, формирование базы данных MySQL состоит из двух этапов:

- 1 Установка соединения с сервером MySQL.
- 2 Выполнение запроса для создания БД.

Определим функцию, которая будет подключаться к серверу MySQL и возвращать объект подключения:

```
import mysql.connector from mysql.connector import Error def create_connection(host_name,
user_name, user_password): connection = None try: connection = mysql.connector.connect(
host=host_name, user=user_name, passwd=user_password ) print("Connection to
MySQL DB successful") except Error as e: print(f"The error '{e}' occurred") return connection
connection = create_connection("localhost", "root", "")
```

В приведенном выше коде мы определили новую функцию create_connection(), которая принимает три параметра:

- 1 host_name
- 2 user_name
- 3 user_password

Модуль mysql.connector определяет метод connect(), используемый в седьмой строке для подключения к серверу MySQL. Как только соединение установлено, объект connection возвращается вызывающей функции. В последней строке функция create_connection() вызывается с именем хоста, именем пользователя и паролем.

Пока мы только установили соединение. Самой базы ещё нет. Для этого мы определим другую функцию – create_database(), которая принимает два параметра:

- 1 Объект connection;
- 2 query – строковый запрос о создании базу данных.

Вот как выглядит эта функция:

```
def create_database(connection, query): cursor = connection.cursor() try:
cursor.execute(query) print("Database created successfully") except Error as e: print(f"The
error '{e}' occurred")
```

Для выполнения запросов используется объект cursor.

Создадим базу данных sm_app для нашего приложения на сервере MySQL:

```
create_database_query = "CREATE DATABASE sm_app" create_database(connection,
create_database_query)
```

Теперь у нас есть база данных на сервере. Однако объект connection, возвращаемый функцией create_connection() подключен к серверу MySQL. А нам необходимо подключиться к базе данных sm_app. Для этого нужно изменить create_connection() следующим образом:

```
def create_connection(host_name, user_name, user_password, db_name):    connection = None    try:
connection = mysql.connector.connect(        host=host_name,        user=user_name,
passwd=user_password,        database=db_name    )    print("Connection to MySQL DB
successful")    except Error as e:        print(f"The error '{e}' occurred")    return connection
```

Функция create_connection() теперь принимает дополнительный параметр с именем db_name. Этот параметр указывает имя БД, к которой мы хотим подключиться. Имя теперь можно передать при вызове функции:

```
connection = create_connection("localhost", "root", "", "sm_app")
```

Скрипт успешно вызывает create_connection() и подключается к базе данных sm_app.

Тема 14. Работа с графикой (ОПК-7)

Лекция.

Графика в Python

Лабораторные работы.

1. Какой командой подключается библиотека, позволяющая работать с черепашьей графикой в Python

- 1)import Черепашка
- 2)import Turtle
- 3)import turtle
- 4)import robot

Ответ:

2. Какая команда позволит Черепашке передвинуться вперед на 50 пикселей

- 1)turtle.forward(50)
- 2)turtle.forward(150)
- 3)turtle.go(50)
- 4)turtle.go(150)

Ответ:

3. Имеется программа. Что она нарисует?

```
Import turtle
Turtle.forward(100)
Turtle.right(90)
Turtle.forward(50)
Turtle.right(90)
Turtle.forward(100)
Turtle.right(90)
Turtle.forward(50)
```

- 1)Прямоугольник
- 2)Квадрат
- 3)Две линии, выходящие из одной точки с углом между ними 90 градусов
- 4)Ромб

Ответ:

4. В начале программы даны описания. Черепашка ориентирована вправо. Какие команды можно использовать, чтобы развернуть Черепашку вверх?

```
Import turtle
Def f(x) :
Turtle.forward(x)
Def lt(g):
Turtle.left(g)
Def rt(g):
Turtle.right(g)
```

```
1)rt(90)
2)lt(90)
3)lt(90)
4)rt(270)
```

Ответ:

5. В начале программы даны описания. Как можно выполнить поворот направо на 90 градусов?

```
Import turtle
Def f(x) :
Turtle.forward(x)
Def lt(g):
Turtle.left(g)
Def rt(g):
Turtle.right(g)
1)rt(90)
2)lt(90)
3)f(90)
```

Ответ:

Задания для самостоятельной работы.

Установка и подключение библиотеки

Нужно скачать файл и положить в ту (именно в ту) директорию, где вы собираетесь далее писать свои программы.

Чтобы импортировать возможности библиотеки `graphics` в вашей программе нужно вставить:

```
import graphics as gr
```

Теперь все объекты этой библиотеки будут вызываться через точку: `gr.КакойНибудьОбъект()`.

Обзор команд модуля `graphics`

Графическое окно — это место, где будут размещаться графические примитивы:

```
# подключение библиотеки под синонимом gr import graphics as gr # Инициализация окна с названием "Russian game" и размером 100x100 пикселей window = gr.GraphWin("Russian game", 100, 100) # Закрытие окна после завершения работы с графикой window.close()
```

Создание графических примитивов:

```
# Создание круга с радиусом 10 и координатами центра (50, 50) my_circle = gr.Circle(gr.Point(50, 50), 10) # Создание отрезка с концами в точках (2, 4) и (4, 8) my_line = gr.Line(gr.Point(2, 4), gr.Point(4, 8)) # Создание прямоугольника у которого диагональ — отрезок с концами в точках (2, 4) и (4, 8) my_rectangle = gr.Rectangle(gr.Point(2, 4), gr.Point(4, 8))
```

Отрисовка примитива в графическом окне производится отдельной командой:

```
# Отрисовка примитивов в окне window my_circle.draw(window) my_line.draw(window) my_rectangle.draw(window)
```

Скрипт выполнится крайне быстро и мы не успеем увидеть результаты нашего труда. Чтобы это исправить, стоит поставить выполнение скрипта на паузу:

```
# Ожидание нажатия кнопки мыши по окну. window.getMouse() # После того как мы выполнили все нужные операции, окно следует закрыть. window.close()
```

Пример программы

Законченный пример:

```
import graphics as gr
window = gr.GraphWin("Jenkslex and Ganzz project", 400, 400)
face = gr.Circle(gr.Point(200, 200), 100)
face.setFill('yellow')
eye1 = gr.Circle(gr.Point(150, 180), 20)
eye2 = gr.Circle(gr.Point(250, 180), 15)
eye1_center = gr.Circle(gr.Point(150, 180), 8)
eye2_center = gr.Circle(gr.Point(250, 180), 7)
eye1.setFill('red')
eye2.setFill('red')
eye1_center.setFill('black')
eye2_center.setFill('black')
eyebrow1 = gr.Line(gr.Point(100, 120), gr.Point(180, 170))
eyebrow2 = gr.Line(gr.Point(220, 170), gr.Point(300, 140))
eyebrow1.setWidth(10)
eyebrow2.setWidth(10)
eyebrow1.setOutline('black')
eyebrow2.setOutline('black')
mouth = gr.Line(gr.Point(150, 260), gr.Point(250, 260))
mouth.setWidth(20)
mouth.setOutline('black')
face.draw(window)
eye1.draw(window)
eye2.draw(window)
eye1_center.draw(window)
eye2_center.draw(window)
eyebrow1.draw(window)
eyebrow2.draw(window)
mouth.draw(window)
window.getMouse()
window.close()
```

Скопируйте код в среду разработки, запустите и посмотрите на результат.

Тема 15. Определение класса и создание экземпляра класса (ОПК-7)

Лекция.

Определение класса и создание экземпляра класса в Python

Лабораторные работы.

1) Какое из представленных слов лучше всего подходит для названия класса?

1) Runner

2) Running

3) Runners

4) Run

5) Параметры функции range могут быть отрицательными целыми числами

Ответ:

2) Какое из следующих утверждений описывает эту строку?

```
xyz = Circle()
```

1) Все не верно

2) создается экземпляр класса

3) переменной присваивается значение класса

4) создается класс

Ответ:

3) Как написать атрибут класса?

1) def some_atr(self): ...

2) self.some_atr = ...

3) some_atr = ...

4) class SomeAtr: ...

Ответ:

4) Как вызвать метод swim(5) у экземпляра hero?

1) hero['swim'] = 5

2) hero.swim(5)

3) hero(swim(5))

4) hero(swim, 5)

Ответ:

5) Что происходит при наследовании Apple от Fruit?

1) Apple перезаписывает класс Fruit

2) В Fruit доступны все атрибуты класса Apple

3) Fruit перезаписывает класс Apple

4) В Apple доступны все атрибуты класса Fruit

Ответ:

Задания для самостоятельной работы.

Доступ к атрибутам

Получите доступ к атрибутам класса, используя оператор `.` после объекта класса. Доступ к классу можно получить используя имя переменной класса:

```
emp1.display_employee() emp2.display_employee() print("Всего сотрудников: %d" %
Employee.emp_count)
```

Теперь, систематизируем все.

```
class Employee:      """Базовый класс для всех сотрудников"""    emp_count = 0    def __init__(self,
name, salary):        self.name = name        self.salary = salary        Employee.emp_count += 1    def
display_count(self):    print('Всего сотрудников: %d' % Employee.emp_count)    def
display_employee(self):    print('Имя: {}. Зарплата: {}'.format(self.name, self.salary))    # Это
создаст первый объект класса Employee    emp1 = Employee("Андрей", 2000)    # Это создаст второй
объект класса Employee    emp2 = Employee("Мария", 5000)    emp1.display_employee()
emp2.display_employee()    print("Всего сотрудников: %d" % Employee.emp_count)
```

При выполнении этого кода, мы получаем следующий результат:

```
Имя: Андрей. Зарплата: 2000 Имя: Мария. Зарплата: 5000 Всего сотрудников: 2
```

Вы можете добавлять, удалять или изменять атрибуты классов и объектов в любой момент.

```
emp1.age = 7 # Добавит атрибут 'age' emp1.age = 8 # Изменит атрибут 'age' del emp1.age # Удалит
атрибут 'age'
```

Вместо использования привычных операторов для доступа к атрибутам вы можете использовать эти функции:

- `getattr(obj, name [, default])` — для доступа к атрибуту объекта.
- `hasattr(obj, name)` — проверить, есть ли в `obj` атрибут `name`.
- `setattr(obj, name, value)` — задать атрибут. Если атрибут не существует, он будет создан.
- `delattr(obj, name)` — удалить атрибут.

```
hasattr(emp1, 'age') # возвращает true если атрибут 'age' существует getattr(emp1, 'age') # возвращает
значение атрибута 'age' setattr(emp1, 'age', 8) #устанавливает атрибут 'age' на 8 delattr(emp1, 'age') #
удаляет атрибут 'age'
```

Встроенные атрибуты класса

Каждый класс Python хранит встроенные атрибуты, и предоставляет к ним доступ через оператор `.`, как и любой другой атрибут:

- `__dict__` — словарь, содержащий пространство имен класса.
- `__doc__` — строка документации класса. `None` если, документация отсутствует.
- `__name__` — имя класса.
- `__module__` — имя модуля, в котором определяется класс. Этот атрибут `__main__` в интерактивном режиме.
- `__bases__` — могут быть пустые tuple, содержащие базовые классы, в порядке их появления в списке базового класса.

Для вышеуказанного класса давайте попробуем получить доступ ко всем этим атрибутам:

```
class Employee:      """Базовый класс для всех сотрудников"""    emp_count = 0    def __init__(self,
name, salary):        self.name = name        self.salary = salary        Employee.empCount += 1    def
display_count(self):    print('Всего сотрудников: %d' % Employee.empCount)    def
display_employee(self):    print('Имя: {}. Зарплата: {}'.format(self.name, self.salary))
print("Employee.__doc__:", Employee.__doc__) print("Employee.__name__:", Employee.__name__)
print("Employee.__module__:", Employee.__module__) print("Employee.__bases__:",
Employee.__bases__) print("Employee.__dict__:", Employee.__dict__)
```

Тема 16. Принципы объектно-ориентированного программирования (ОПК-7)

Лекция.

Принципы объектно-ориентированного программирования в Python

Лабораторные работы.

1) Что относится к основным принципам ООП?

- 1) Инкапсуляция, полиморфизм, наследование, абстракция
- 2) Инкапсуляция, полиморфизм, делегирование, абстракция
- 3) Полиморфизм, разделение интерфейса, наследование, абстракция
- 4) Инкапсуляция, наследование, абстракция, открытость/закрытость

Отвте:

2) Что будет выведено на экран?

Class test:

Test = None

Print(Test.test)

- 1) 0
- 2) False
- 3) None
- 4) Ошибка

Ответ:

3) Какой принцип ООП описывает следующее предложение? Этот принцип является способностью использовать общий интерфейс для нескольких форм (типов данных).

- 1) Инкапсуляция
- 2) Полиморфизм
- 3) Абстракция
- 4) Наследование

Ответ:

4) Какой из перечисленных вариантов является верным объявлением private поля?

- 1) private field = 0
- 2) field = 0
- 3) _field = 0
- 4) __field = 0

Ответ:

5) Как создать конструктор класса А?

- 1) А(параметры конструктора)
- 2) def __init__(параметры конструктора)
- 3) def __A__(параметры конструктора)
- 4) def init(параметры конструктора)

Ответ:

Задания для самостоятельной работы.

Основы ООП на Python для начинающих

Классы Основы ООП на Python для начинающих

Классы

Создавать классы в Python очень просто:

```
class SomeClass(object): # поля и методы класса SomeClass
```

Классы-родители перечисляются в скобках через запятую:

```
class SomeClass(ParentClass1, ParentClass2, ...): # поля и методы класса SomeClass
```

С реализацией наследования разберемся чуть позже.

Свойства классов устанавливаются с помощью простого присваивания:

```
class SomeClass(object): attr1 = 42 attr2 = "Hello, World"
```

Методы объявляются как простые функции:

```
class SomeClass(object): def method1(self, x): # код метода
```

Обратите внимание на первый аргумент – `self` – общепринятое имя для ссылки на объект, в контексте которого вызывается метод. Этот параметр обязателен и отличает метод класса от обычной функции.

Все пользовательские атрибуты сохраняются в атрибуте `__dict__`, который является словарем.

Экземпляры классов

Инстанцировать класс в Python тоже очень просто:

```
class SomeClass(object):    attr1 = 42    def method1(self, x):        return 2*x    obj = SomeClass()
obj.method1(6) # 12 obj.attr1 # 42
```

Можно создавать разные инстансы одного класса с заранее заданными параметрами с помощью инициализатора (специальный метод `__init__`). Для примера возьмем класс `Point` (точка пространства), объекты которого должны иметь определенные координаты:

```
class Point(object):    def __init__(self, x, y, z):        self.coord = (x, y, z)    p = Point(13, 14, 15)
p.coord # (13, 14, 15)
```

Подробнее о других специальных методах жизненного цикла объектов поговорим чуть ниже.

Динамическое изменение

Можно обойтись даже без определения атрибутов и методов:

```
class SomeClass(object):    pass
```

Кажется, этот класс совершенно бесполезен? Отнюдь. Классы в Python могут динамически изменяться после определения:

```
class SomeClass(object):    pass    def squareMethod(self, x):        return x*x    SomeClass.square =
squareMethod    obj = SomeClass()    obj.square(5) # 25
```

Статические и классовые методы

Для создания статических методов в Python предназначен декоратор `@staticmethod`. У них нет обязательных параметров-ссылок вроде `self`. Доступ к таким методам можно получить как из экземпляра класса, так и из самого класса:

```
class SomeClass(object):    @staticmethod    def hello():        print("Hello, world")    SomeClass.hello() #
Hello, world    obj = SomeClass()    obj.hello() # Hello, world
```

Еще есть так называемые методы классов. Они аналогичны методам экземпляров, но выполняются не в контексте объекта, а в контексте самого класса (классы – это тоже объекты). Такие методы создаются с помощью декоратора `@classmethod` и требуют обязательную ссылку на класс (`cls`).

```
class SomeClass(object):    @classmethod    def hello(cls):        print('Hello, класс
{}'.format(cls.__name__))    SomeClass.hello() # Hello, класс SomeClass
```

Статические и классовые методы доступны без инстанцирования.

Специальные методы

Жизненный цикл объекта

С инициализатором объектов `__init__` вы уже знакомы. Кроме него есть еще и метод `__new__`, который непосредственно создает новый экземпляр класса. Первым параметром он принимает ссылку на сам класс:

```
class SomeClass(object):    def __new__(cls):        print("new")        return super(SomeClass,
cls).__new__(cls)    def __init__(self):        print("init")    obj = SomeClass(); # new # init
```

Это обсуждение на [stackoverflow](https://stackoverflow.com) поможет лучше разобраться с инстанцированием классов.

Метод `__new__` может быть очень полезен для решения ряда задач, например, создания имутабельных объектов или реализации паттерна Синглтон:

```
class Singleton(object):    obj = None # единственный экземпляр класса    def __new__(cls, *args,
**kwargs):        if cls.obj is None:            cls.obj = object.__new__(cls, *args, **kwargs)        return cls.obj    single
= Singleton()    single.attr = 42    newSingle = Singleton()    newSingle.attr # 42    newSingle is single # true
```

В Python вы можете поучаствовать не только в создании объекта, но и в его удалении. Специально для этого предназначен метод-деструктор `__del__`.

```
class SomeClass(object):    def __init__(self, name):        self.name = name    def __del__(self):
print('удаляется объект {} класса SomeClass'.format(self.name))    obj = SomeClass("John"); del obj #
удаляется объект John класса SomeClass
```

На практике деструктор используется редко, в основном для тех ресурсов, которые требуют явного освобождения памяти при удалении объекта. Не следует совершать в нем сложные вычисления.

Объект как функция

Объект класса может имитировать стандартную функцию, то есть при желании его можно "вызвать" с параметрами. За эту возможность отвечает специальный метод `__call__`:

```
class Multiplier:
    def __call__(self, x, y):
        return x*y
multiply = Multiplier()
multiply(19, 19) # 361
# то же самое multiply.__call__(19, 19) # 361
```

Имитация контейнеров

Вы знакомы с функцией `len()`, которая умеет вычислять длину списков значений?

```
list = ['hello', 'world']
len(list) # 2
```

Но для объектов вашего пользовательского класса это не пройдет:

```
class Collection:
    def __init__(self, list):
        self.list = list
collection = Collection(list)
len(collection)
```

Этот код выдаст ошибку `object of type 'Collection' has no len()`. Интерпретатор просто не понимает, как ему посчитать длину `collection`.

Решить эту проблему поможет специальный метод `__len__`:

```
class Collection:
    def __init__(self, list):
        self.list = list
    def __len__(self):
        return len(self.list)
collection = Collection([1, 2, 3])
len(collection) # 3
```

Можно работать с объектом как с коллекцией значений, определив для него интерфейс классического списка с помощью специальных методов:

- `__getitem__` – реализация синтаксиса `obj[key]`, получение значения по ключу;
- `__setitem__` – установка значения для ключа;
- `__delitem__` – удаление значения;
- `__contains__` – проверка наличия значения.

Имитация числовых типов

Ваши объекты могут участвовать в математических операциях, если у них определены специальные методы. Например, `__mul__` позволяет умножать объект на число по определенной программистом логике:

```
class SomeClass:
    def __init__(self, value):
        self.value = value
    def __mul__(self, number):
        return self.value*number
obj = SomeClass(42)
print(obj * 100) # 4200
```

Другие специальные методы

В Python существует огромное количество специальных методов, расширяющих возможности пользовательских классов. Например, можно определить вид объекта на печати, его "официальное" строковое представление или поведение при сравнениях. Узнать о них подробнее вы можете в официальной документации языка.

Эти методы могут эмулировать поведение встроенных классов, но при этом они необязательно существуют у самих встроенных классов. Например, у объектов `int` при сложении не вызывается метод `__add__`. Таким образом, их нельзя переопределить. Создавать классы в Python очень просто:

```
class SomeClass(object):
    # поля и методы класса SomeClass
```

Классы-родители перечисляются в скобках через запятую:

```
class SomeClass(ParentClass1, ParentClass2, ...):
    # поля и методы класса SomeClass
```

С реализацией наследования разберемся чуть позже.

Свойства классов устанавливаются с помощью простого присваивания:

```
class SomeClass(object):
    attr1 = 42
    attr2 = "Hello, World"
```

Методы объявляются как простые функции:

```
class SomeClass(object):
    def method1(self, x):
        # код метода
```

Обратите внимание на первый аргумент – `self` – общепринятое имя для ссылки на объект, в контексте которого вызывается метод. Этот параметр обязателен и отличает метод класса от обычной функции.

Все пользовательские атрибуты сохраняются в атрибуте `__dict__`, который является словарем.

Экземпляры классов

Инстанцировать класс в Python тоже очень просто:

```
class SomeClass(object):    attr1 = 42    def method1(self, x):        return 2*x    obj = SomeClass()
obj.method1(6) # 12 obj.attr1 # 42
```

Можно создавать разные инстансы одного класса с заранее заданными параметрами с помощью инициализатора (специальный метод `__init__`). Для примера возьмем класс Point (точка пространства), объекты которого должны иметь определенные координаты:

```
class Point(object):    def __init__(self, x, y, z):        self.coord = (x, y, z)    p = Point(13, 14, 15)    p.coord # (13, 14, 15)
```

Подробнее о других специальных методах жизненного цикла объектов поговорим чуть ниже.

Динамическое изменение

Можно обойтись даже без определения атрибутов и методов:

```
class SomeClass(object):    pass
```

Кажется, этот класс совершенно бесполезен? Отнюдь. Классы в Python могут динамически изменяться после определения:

```
class SomeClass(object):    pass    def squareMethod(self, x):        return x*x    SomeClass.square = squareMethod
obj = SomeClass()    obj.square(5) # 25
```

Статические и классовые методы

Для создания статических методов в Python предназначен декоратор `@staticmethod`. У них нет обязательных параметров-ссылок вроде `self`. Доступ к таким методам можно получить как из экземпляра класса, так и из самого класса:

```
class SomeClass(object):    @staticmethod    def hello():        print("Hello, world")    SomeClass.hello() # Hello, world
obj = SomeClass()    obj.hello() # Hello, world
```

Еще есть так называемые методы классов. Они аналогичны методам экземпляров, но выполняются не в контексте объекта, а в контексте самого класса (классы – это тоже объекты). Такие методы создаются с помощью декоратора `@classmethod` и требуют обязательную ссылку на класс (`cls`).

```
class SomeClass(object):    @classmethod    def hello(cls):        print('Hello, класс {}'.format(cls.__name__))    SomeClass.hello() # Hello, класс SomeClass
```

Статические и классовые методы доступны без инстанцирования.

Специальные методы

Жизненный цикл объекта

С инициализатором объектов `__init__` вы уже знакомы. Кроме него есть еще и метод `__new__`, который непосредственно создает новый экземпляр класса. Первым параметром он принимает ссылку на сам класс:

```
class SomeClass(object):    def __new__(cls):        print("new")        return super(SomeClass, cls).__new__(cls)    def __init__(self):        print("init")    obj = SomeClass(); # new # init
```

Это обсуждение на [stackoverflow](#) поможет лучше разобраться с инстанцированием классов.

Метод `__new__` может быть очень полезен для решения ряда задач, например, создания имутабельных объектов или реализации паттерна Синглтон:

```
class Singleton(object):    obj = None # единственный экземпляр класса    def __new__(cls, *args, **kwargs):        if cls.obj is None:            cls.obj = object.__new__(cls, *args, **kwargs)        return cls.obj    single = Singleton()    single.attr = 42    newSingle = Singleton()    newSingle.attr # 42    newSingle is single # true
```

В Python вы можете поучаствовать не только в создании объекта, но и в его удалении. Специально для этого предназначен метод-деструктор `__del__`.

```
class SomeClass(object):    def __init__(self, name):        self.name = name    def __del__(self):        print('удаляется объект {} класса SomeClass'.format(self.name))    obj = SomeClass("John");    del obj # удаляется объект John класса SomeClass
```

На практике деструктор используется редко, в основном для тех ресурсов, которые требуют явного освобождения памяти при удалении объекта. Не следует совершать в нем сложные вычисления.

Объект как функция

Объект класса может имитировать стандартную функцию, то есть при желании его можно "вызвать" с параметрами. За эту возможность отвечает специальный метод `__call__`:

```
class Multiplier:
    def __call__(self, x, y):
        return x*y
multiply = Multiplier()
multiply(19, 19) # 361
# то же самое multiply.__call__(19, 19) # 361
```

Имитация контейнеров

Вы знакомы с функцией `len()`, которая умеет вычислять длину списков значений?

```
list = ['hello', 'world']
len(list) # 2
```

Но для объектов вашего пользовательского класса это не пройдет:

```
class Collection:
    def __init__(self, list):
        self.list = list
collection = Collection(list)
len(collection)
```

Этот код выдаст ошибку `object of type 'Collection' has no len()`. Интерпретатор просто не понимает, как ему посчитать длину `collection`.

Решить эту проблему поможет специальный метод `__len__`:

```
class Collection:
    def __init__(self, list):
        self.list = list
    def __len__(self):
        return len(self.list)
collection = Collection([1, 2, 3])
len(collection) # 3
```

Можно работать с объектом как с коллекцией значений, определив для него интерфейс классического списка с помощью специальных методов:

- `__getitem__` – реализация синтаксиса `obj[key]`, получение значения по ключу;
- `__setitem__` – установка значения для ключа;
- `__delitem__` – удаление значения;
- `__contains__` – проверка наличия значения.

Имитация числовых типов

Ваши объекты могут участвовать в математических операциях, если у них определены специальные методы. Например, `__mul__` позволяет умножать объект на число по определенной программистом логике:

```
class SomeClass:
    def __init__(self, value):
        self.value = value
    def __mul__(self, number):
        return self.value*number
obj = SomeClass(42)
print(obj * 100) # 4200
```

Другие специальные методы

В Python существует огромное количество специальных методов, расширяющих возможности пользовательских классов. Например, можно определить вид объекта на печати, его "официальное" строковое представление или поведение при сравнениях. Узнать о них подробнее вы можете в официальной документации языка.

Эти методы могут эмулировать поведение встроенных классов, но при этом они необязательно существуют у самих встроенных классов. Например, у объектов `int` при сложении не вызывается метод `__add__`. Таким образом, их нельзя переопределить.

Тема 17. Статические методы и методы класса (ОПК-7)

Лекция.

Статические методы и методы класса в Python

Лабораторные работы.

1. Создать класс, описывающий человека. Должны быть поля для имени, фамилии и возраста. Создать экземпляр и вывести информацию о человеке.
2. Доработать предыдущий класс, чтобы вся информация о человеке была доступна при вызове `str` над экземпляром.
3. Добавить метод `greet`, вызов которого будет выводить в консоль информацию о человеке в формате "Привет! Меня зовут Петров Василий, мне 12 лет".
4. Добавить атрибут `grades`, в котором будет храниться список оценок. Создать список учеников, заполняя оценки случайными числами, и вывести информацию о них в порядке убывания среднего балла. Заполнение оценок и подсчет среднего балла вынести в отдельные методы.

Задания для самостоятельной работы.

1. Создайте класс прямоугольник — `Rectangle`. Метод `__init__` принимает две точки — левый верхний и правый нижний угол.

Каждая точка представлена экземпляром класса Point. Реализуйте методы вычисления площади и периметра прямоугольника.

2. Добавьте в класс Rectangle метод has_point. Метод принимает точку на плоскости и возвращает True,

если точка находится внутри прямоугольника и False в противном случае.

3. Создайте класс фигура — Figure. Метод __init__ принимает число — количество строительных блоков фигуры.

Каждый объект будет состоять из заданного количества строительных блоков. У класса должен быть метод print_figure,

который печатает фигуру. С использованием этого класса реализуйте программу, которая будет “строить стену” из случайного количества строительных блоков.

Тема 18. Основы разработки оконных приложений (ОПК-7)

Лекция.

Основы разработки оконных приложений в Python

Лабораторные работы.

Разработка приложения с использованием ООП и графического интерфейса

2.1 Создание окна, рамки и кнопок

В качестве первого примера создания пользовательского приложения с использованием графической библиотеки tkinter рассмотрим программу guil.py, в которой создается пользовательское окно, а в нем — рамка с тремя кнопками:

```
from tkinter import *

class My_frame (Frame):

    """ Рамка с тремя кнопками """

    def __init__ (self, master):

        super ().__init__ (master)

        self.grid()

        self.create_widgets()

    def create_widgets (self):

        """ Создает три кнопки """

        self.bt1=Button(self, text= 'Кнопка №1' , fg= 'blue' , font= '12' )

        self.bt1.grid()

        self.bt2=Button(self)

        self.bt2.grid()

        self.bt2[ 'text' ]= 'Кнопка №2'
```

```

self.bt2[ 'fg' ]= 'blue'

self.bt2[ 'font' ]= '12'

self.bt3=Button(self)

self.bt3.grid()

self.bt3.config(text= 'Кнопка №3' , fg= 'blue' , font= '12' )

)

root=Tk() #Создание базового окна

root.title( 'Рамка с кнопками' )

root.geometry( '230x95+800+200' )

app=My_frame(root)

root.mainloop()

```

Для запуска программы `gui1.py` на выполнение непосредственно из лабораторной работы используется ссылка: `IE:gui1.pyw`. Для правильной работы ссылки необходимо:

- *применять только браузер Internet Explorer;*
- *использовать специальную запускающую программу `gol.js` –*

```
new ActiveXObject("WScript.shell")).run(PYTHON\\LAB13\\gui1.pyw"
```

- *прописать в ссылке путь к программе `gol.js`;*
- *хранить файлы с программами `gui1.pyw` и `gol.js` в папке "LAB13".*

В программе `gui1.py` подключение модуля `tkinter` осуществляется с использованием формы `from tkinter import *`. При этом имена всех классов и методов модуля помещаются в пространство имен программы и обращение к ним происходит без добавления префикса `tkinter`.

На базе класса `Frame` модуля `tkinter` создается пользовательский класс `My_frame`, который описывает рамку, содержащую три кнопки. При создании конструктора этого класса используется встроенная функция `super()`, которая предназначена для наследования методов родительского класса при их перегрузке. Таким образом, к методам конструктора класса `Frame()` объекта `master` добавляются методы: `grid()` – для размещения рамки, и `create_widgets()` – для создания кнопок в рамке.

Кнопки в виде объектов `bt1`, `bt2` и `bt3` создаются с использованием класса `Button` модуля `tkinter`. Хотя кнопки создаются в этом примере одинаковые, способы задания параметров кнопки: `text` – указывающего название кнопки, `fg` и `font` – задающих соответственно цвет (`blue`) и размеры (12 пикселей) символов, используются разные:

- *для кнопки №1 – непосредственно при ее создании;*

- для кнопки №2 – задаются с помощью ключей словаря экземпляра класса *Button*;
- для кнопки №3 – с помощью метода *config()*.

После описания класса в программе *guil.py*:

- создается пользовательское окно, обычно именуемое *root*;
- указываются имя этого окна – 'Рамка с кнопками' и его размеры и расположение на экране:
- в этом окне создается рамка, свойства которой описаны в классе *My_frame()*;
- запускается цикл обработки событий (хотя в этом примере события не обрабатываются).
- длина – 230 пикселей;
- ширина – 85 пикселей;
- смещение по горизонтали – 800 пикселей;
- смещение по вертикали – 200 пикселей;

При выполнении программы *IE: guil.pyw* на экране появляется окно с заданными графическими объектами.

Задания для самостоятельной работы.

Мы рассмотрим следующие виджеты:

- **Button (Кнопка)** Простая кнопка для вызова некоторых действий (выполнения определенной команды).
- **Canvas (Рисунок)** Основа для вывода графических примитивов.
- **Label (Надпись)** Виджет может показывать текст или графическое изображение.

Проведём нашу работу по шагам, разберем всё подробно. Рассмотрим программный код.

Шаг 1.

from tkinter import* // загружаем графическую библиотеку

root=Tk() // создаём окно *root*

root.mainloop() // запускаем первое окно

Когда мы наберем данный код на Питоне мы получим какое то окошко неопределенного размера и имени. Но это уже все таки окошко.

Перейдём ко второму шагу и укажем нашему окну имя.

Шаг 2.

from tkinter import*

root=Tk()

root.title("Окошко") // указываем заголовок окна

root.mainloop()

Всё готово и дело за размерами нашего окна. Переходим к следующему шагу.

Шаг 3.

from tkinter import*

root=Tk()

root.title("Окошко")

```
root.geometry('800x600') // указываем размер окна
```

```
root.mainloop()
```

И ещё один шаг придадим нашему окну цвет.

Шаг 4.

from tkinter import* // загружаем графическую библиотеку

```
root=Tk() // создаём окно root
```

```
root.title("Окошко") // указываем заголовок окна
```

```
root.geometry('800x600') // указываем размер окна
```

```
root.config(bg='red') // задаём цвет окна
```

```
root.mainloop() // запускаем первое окно
```

Вот всё готово, любуемся творением. Следующим шагом мы создадим надпись(Label) в нашем окне.

Шаг 5.

```
from tkinter import*
```

```
root=Tk()
```

```
root.title("Окошко")
```

```
root.geometry('800x600')
```

```
root.config(bg='red')
```

```
t1=Label(root, text='text in windows', fg='yellow', bg='grey')
```

```
t1.config(font=('Times', 25))
```

```
t1.pack()
```

```
root.mainloop()
```

С надписью разобрались переходим к объекту кнопка(Button).

Шаг 6.

```
from tkinter import*
```

```
root=Tk()
```

```
root.title("Окошко")
```

```
root.geometry('800x600')
```

```
root.config(bg='red')
```

```
t1=Label(root, text='text in windows', fg='yellow', bg='grey')
```

```
t1.config(font=('Times', 25))
```

```
t1.pack()
```

```
b1=Button(root, text='Click my')
```

```
b1.config(width=20, height=2, bg='green', fg='blue')
```

```
b1.pack()
```

```
root.mainloop()
```

И немного забежим вперед. Познакомимся еще с одним объектом – холст.

Теперь мы имеем окошко с именем, размером, и определённым цветом. Но этого нам еще мало для работы.

Для обрисовки графических примитивов необходимо вызвать объект модуля tkinter под названием canvas (холст) – позволяет размещать на себе объекты: изображения, кнопки, текстовые поля и др...

Добавим программный код в нашу программу.

from tkinter import*

```
root=Tk()
```

```

root.title("Окошко")

root.geometry('800x600')

canvas=Canvas(root,width=800, height=600,bg="blue")

canvas.pack()

root.mainloop()
Рисуем вертикальные линии
from tkinter import*

root=Tk()

root.title("Окошко")

root.geometry('800x600')

canvas=Canvas(root,width=800, height=600,bg="blue")

```

```
for y in range(16):
```

```
    k = 50 * y
```

```
    canvas.create_line(10+k,600,10+k,0,width=1,fill="red")
```

```
    canvas.pack()
```

```
root.mainloop()
```

Заключение

Ну что если у Вас тоже всё получилось, поздравляю можно смело кричать ура! Мы достигли результата, а значит, мы научились создавать окна или иначе оконные приложения для Windows. Правда, здорово это мы проделали своими руками, а не пользовались готовыми приложениями.

А теперь перечислим всё чему мы научились.

- 1 Создавать оконные приложения
- 2 Задавать различные свойства окнам (цвет, размер)
- 3 Создавать надписи
- 4 Создавать кнопки

Мы разобрались с данными понятиями и теперь можно смело продвигаться вперед по волнам языка программирования Python.

В добрый путь дорогие начинающие программисты.

4. Контроль знаний обучающихся и типовые оценочные средства

4.1. Распределение баллов:

2 семестр

- посещаемость – 10 баллов
- текущий контроль – 57 баллов

- контрольные срезы – 2 среза: 1 балл, 2 балла
- премиальные баллы – 20 баллов
- ответ на экзамене: не более 30 баллов

Распределение баллов по заданиям:

№ те мы	Название темы / вид учебной работы	Формы текущего контроля / срезы	Мах. кол-во баллов	Методика проведения занятия и оценки
1.	Переменные. Типы данных. Преобразование типов данных	Собеседование	2	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>4 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>3 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	1	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

2.	Условные операторы и циклы	Собеседование(контрольный срез)	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>3 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>2 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

3.	Строки и двоичные данные	Собеседование	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>4 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>3 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>
4.	Функции и методы для работы со строками	Собеседование	1	- рациональность использованных приемов и способов решения поставленной учебной задачи;
		Тестирование(контрольный срез)	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

5.	Списки. Операции над списками	Собеседо вание	2	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>4 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>3 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестиров ание	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

6.	Кортежи, множества и диапазоны.	Собеседование	2	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>4 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>3 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

7.	Словари. Операции и методы для работы со словарями	Собеседование	2	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>4 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>3 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	1	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

8.	Работа с датой и временем	Собеседование	2	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>4 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>3 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>2 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балл - студент правильно отвечает на 25-50% вопросов в тесте.</p>

9.	Пользовательские функции	Собеседование	2	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>4 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>3 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

10.	Обработка исключений	Собеседование	2	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>2 балла - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балл – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>2 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балл - студент правильно отвечает на 25-50% вопросов в тесте.</p>

11.	Работа с файлами и каталогами	Собеседование	2	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>2 балла - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балл – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>2 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балл - студент правильно отвечает на 25-50% вопросов в тесте.</p>

12.	Доступ из Python к базам данных SQLite	Собеседование	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>4 баллов – студент умеет сопоставить полученную при подготовке к практическому занятию информацию, сравнивать разные точки зрения на анализируемую проблему, уметь четко формулировать свои вопросы и отвечать на задаваемые ему вопросы</p> <p>3 баллов - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балла – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

13.	Доступ из Python к базам данных MySQL	Собеседование	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>2 балла - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балл – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>2 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балл - студент правильно отвечает на 25-50% вопросов в тесте.</p>

14.	Работа с графикой	Собеседование	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>2 балла - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балл – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

15.	Определение класса и создание экземпляра класса	Собеседование	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>2 балла - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балл – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

16.	Принципы объектно-ориентированного программирования	Собеседование	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>2 балла - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балл – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

17.	Статические методы и методы класса	Собеседование	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>2 балла - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балл – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
		Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>
18.	Основы разработки оконных приложений	Тестирование	2	<p>Оценка теста по текущему разделу или теме дисциплины</p> <p>3 балла – студент правильно отвечает на 50-100% вопросов в тесте.</p> <p>1 балла - студент правильно отвечает на 25-50% вопросов в тесте.</p>

		Собеседование	1	<p>Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.</p> <p>Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:</p> <ul style="list-style-type: none"> - правильность ответа по содержанию; - полнота и глубина ответа; - сознательность ответа; - логика изложения материала; - рациональность использованных приемов и способов решения поставленной учебной задачи; - своевременность и эффективность использования наглядных пособий и технических средств при ответе; - использование дополнительного материала; - рациональность использования времени, отведенного на задание. <p>2 балла - студент умеет применять полученную при подготовке к практическому занятию информацию, отвечать на большинство вопросов, вести дискуссию .</p> <p>1 балл – студент владеет теоретическим материалом по теме практического занятия, иногда затрудняется при ответе на вопросы, не умеет сформулировать свою точку зрения на обсуждаемую проблему</p> <p>Если студент не владеет проблематикой практического занятия, не может отвечать на вопросы, зачитывает ответ по напечатанному тексту – ответ баллами не оценивается.</p>
19.	Посещаемость		10	<p>7 баллов – стопроцентное посещение занятий студентом</p> <p>5-6 баллов – посещаемость студента составляет не менее 80 % занятий</p> <p>3-4 баллов – посещаемость студента составляет не менее 50 % занятий</p> <p>1-2 балла – посещаемость студента составляет не менее 25 % занятий</p>
20.	Премияльные баллы		20	<p>Дополнительные премияльные баллы могут быть начислены:</p> <ul style="list-style-type: none"> - за проект, выполненный по заказу работодателя и реализованный на практике – 20 баллов; - постоянная активность во время практических занятий – 10 баллов; - полностью подготовленная к публикации статья по тематике в рамках дисциплины – 10 баллов; - участие с докладом во всероссийской олимпиаде по тематике изучаемой дисциплины – 20 баллов; - участие в выставке по тематике изучаемой дисциплины – 20 баллов; - публикация статьи по тематике изучаемой дисциплины в сборнике студенческих работ / материалах всероссийской конференции / журнале из перечня ВАК – 10 / 15 / 20

21.	Ответ на экзамене	30	<p>Оценка «удовлетворительно»- студент имеет достаточный минимальный объем знаний по дисциплине; студентом усвоена основная литература, рекомендованная учебной программой; студент умеет ориентироваться в основных теориях, концепциях и направлениях по дисциплине и давать им оценку; студент умеет делать выводы без существенных ошибок;</p> <p>Оценка «хорошо» – «достаточно полные и систематизированные знания по дисциплине;» умение ориентироваться в основном теориях, концепциях и направлениях дисциплины и давать им критическую оценку; использование научной терминологии, лингвистически и логически правильное изложение ответа на вопросы, умение делать обоснованные выводы; владение инструментарием по дисциплине, умение его использовать в постановке и решении научных и профессиональных задач; усвоение основной и дополнительной литературы, рекомендованной учебной программой по дисциплине; самостоятельная работа на практических занятиях, участие в групповых обсуждениях, высокий уровень культуры исполнения заданий; средний уровень сформированности заявленных в рабочей программе компетенций.</p> <p>- Оценка «отлично» – систематизированные и полные знания по всем разделам дисциплины, а также по основным вопросам, выходящим за пределы учебной программы; точное использование научной терминологии систематически грамотное и логически правильное изложение ответа на вопросы; безупречное владение инструментарием учебной дисциплины, умение его эффективно использовать в постановке научных и практических задач; выраженная способность самостоятельно и творчески решать сложные проблемы и нестандартные ситуации; полное и глубокое усвоение основной и дополнительной литературы, рекомендованной учебной программой по дисциплине; умение ориентироваться в теориях, концепциях и направлениях дисциплины и давать им критическую оценку, используя научные достижения других дисциплин; творческая самостоятельная работа; активное участие в групповых обсуждениях.</p>
22.	Итого за семестр	100	

Итоговая оценка по экзамену выставляется в 100-балльной шкале и в традиционной четырехбалльной шкале. Перевод 100-балльной рейтинговой оценки по дисциплине в традиционную четырехбалльную осуществляется следующим образом:

100-балльная система	Традиционная система
85 - 100 баллов	Отлично
70 - 84 баллов	Хорошо
50 - 69 баллов	Удовлетворительно
Менее 50	Неудовлетворительно

4.2 Типовые оценочные средства текущего контроля

Собеседование

Тема 1. Переменные. Типы данных. Преобразование типов данных

1)Что такое переменная в Python?

- 2) Как указать значение переменной в Python?
- 3) Какие типы данных вы знаете?
- 4) Какие виды числового типа данных вы знаете?
- 5) Как преобразовывать типы данных?

Тема 2. Условные операторы и циклы

1. Какие условные операторы и циклы вы знаете?
2. Что делает оператор цикла выполняет?
3. Для чего предназначены операторы break и continue?
4. Для чего используется конструкция if – elif – else?

Тема 3. Строки и двоичные данные

1. Строки в python обозначаются кавычками. Приведите все способы.
2. Какие типы данных можно преобразовать в строку?
3. Опишите синтаксис срезов строк при помощи квадратных скобок.
4. Как применяют операции сложения и умножения к строкам?
5. Что такое двоичные данные?

Тема 4. Функции и методы для работы со строками

- 4.1. Арифметические выражения.
- 4.2. Логические выражения.
- 4.3. Строковые выражения.

Тема 5. Списки. Операции над списками

1. Перечислите характеристики типа данных «список», которые вы знаете.
2. Как объединить списки?
3. Как умножать списки?
4. Как перевернуть список?
5. В чём разница между append и extend?

Тема 6. Кортежи, множества и диапазоны.

1. Чем список отличается от других структур?
2. Как объединить два списка в список кортежей?
3. Как работает функция range?
4. В каких ситуациях лучше использовать списки, а в каких кортежи, словари или множества?

Тема 7. Словари. Операции и методы для работы со словарями

1. Какими способами можно получить доступ к значению ключа, не изменяя при этом словарь?
2. Что может служить ключом словаря? Перечислите структуры данных и общие требования к именованию.
3. Для чего нужен метод pop()? Какие параметры он может принимать? Приведите пример использования.
4. Охарактеризуйте методы keys(), items(), values(). Что они возвращают, какова специфика результирующих объектов?
5. В чем опасность копирования словаря? Для чего нужно глубокое копирование?

Тема 8. Работа с датой и временем

1. Как получить текущую дату и время в Python?

2. Как получить текущую дату?
3. Что внутри datetime?
4. Как вывести час, минуту, секунду и микросекунду?

Тема 9. Пользовательские функции

1. Что такое пользовательская функция?
2. Чем служит ключевое слово def?
3. Для чего используется слово return?
4. Приведите пример функции Python, которая принимает два параметра, вычисляет сумму и возвращает вычисленное значение.

Тема 10. Обработка исключений

1. Что означает исключение в python?
2. Какие стандартные исключения в Python вы знаете?
3. Блок множественного исключения

Тема 11. Работа с файлами и каталогами

1. Какими способами можно открыть текстовый файл (в формате .txt) в Python (без использования сторонних библиотек)?
2. Опишите основные режимы открытия документа, доступные в функции open().
3. Как используют функцию print() для записи файлов?
4. Охарактеризуйте любые 3 объекта модуля os.

Тема 12. Доступ из Python к базам данных SQLite

1. Как создать базу данных SQLite?
2. Как вставить данные в таблицу?
3. Как редактировать данные?
4. Как удалять данные?
5. Базовые запросы SQL

Тема 13. Доступ из Python к базам данных MySQL

1. Как создать базу данных MySQL?
2. Как вставить данные в таблицу?
3. Как редактировать данные?
4. Как удалять данные?
5. Базовые запросы SQL

Тема 14. Работа с графикой

1. Какая функция используется для задания точки в Python?
2. Какая функция используется для задания отрезка в Python?
3. Какие функция для создания фигур вы знаете
4. Какая команда используется для создания текста в графическом окне в Python?

Тема 15. Определение класса и создание экземпляра класса

1. Как связаны классы и объекты?
2. Для чего необходимо ключевое слово self в классах?
3. Как создаются и для чего нужны статические методы?
4. Как реализуется наследование классов в Python?
5. Что такое дескрипторы данных?

Тема 16. Принципы объектно-ориентированного программирования

1. Что такое ООП?
2. Назовите принципы ООП
3. Зачем нужна абстракция?
4. Для чего нужна инкапсуляция?
5. Для чего нужно наследование?

Тема 17. Статические методы и методы класса

1. Что такое статический метод в Python?
2. Для чего нужны статические методы класса в Python?
3. Как использовать подход `staticmethod():?`

Тема 18. Основы разработки оконных приложений

1. Какой конструктор применяется для создания графического окна?
2. Как установить заголовок окна?
3. Как изменить позицию окна?

Тестирование

Тема 1. Переменные. Типы данных. Преобразование типов данных

1) Чему равно значение z:

```
x, y, z = 32 43 11
```

- 1) 32
- 2) z
- 3) 11
- 4) Код не выполнится из-за ошибки

Ответ:

2) Какой тип данных вернет этот код `print(type(88))`

- 1) integer
- 2) str
- 3) int
- 4) float

Ответ:

3) Что выведет этот код:

```
x = 685.0
```

```
print(x)
```

- 1) Ошибку, переменная должна быть длиннее 3х символов
- 2) 685
- 3) Ошибку, десятичную часть нужно отделять запятой
- 4) 685.0
- 5) "685.0"

4) Как создать переменную с типом float?

- 1) `var = "9846"`
- 2) `var = "9846.499"`
- 3) `var = 9846`
- 4) `var = 9846.495`

5) Какие типы данных из урока могут содержать цифры?

- 1) Все

- 2) bool, str
- 3) int, float, bool
- 4) int, float
- 5) int, float, str

Тема 2. Условные операторы и циклы

1) Что выведет программа?

```
if 4>8:
    print(6)
else:
    print(3)
```

Ответ:

2) Что выведет программа?

```
a = -10
b = -1
if a<b:
    print(a)
else:
    print(b)
```

Ответ:

3) Что выведет программа?

```
a = False
b = False
if a or b:
    print(1)
else:
    print(2)
```

Ответ:

4) Что выведет программа?

```
if True:
    print(1)
else:
    print(0)
```

Ответ:

5) Что выведет программа?

```
a = False
b = True
if a and b:
    print(1)
else:
    print(2)
```

Ответ:

Тема 3. Строки и двоичные данные

1) Что будет выведено в результате выполнения данной программы:

```
def string_length(str1):
    count = 0
    for char in str1:
        count += 1
```

```
return count
```

```
print(string_length('w3resource.com'))
```

1) Длина строки прописью

2) Длина строки цифрой

3) Количество букв в строке

4) Количество цифр в строке

2) Что будет результатом выполнения данной программы:

```
def chars_mix_up(a, b):
```

```
new_a = b[:2] + a[2:]
```

```
new_b = a[:2] + b[2:]
```

```
return new_a + ' ' + new_b
```

```
print(chars_mix_up('abc', 'xyz'))
```

а) Первые буквы строк поменяются местами

б) Последние буквы строк поменяются местами

в) Поменяет строки местами

г) Объединит строки в одну и перемешает

3) Что будет выведено в результате выполнения данной программы:

```
def find_longest_word(words_list):
```

```
word_len = []
```

```
for n in words_list:
```

```
word_len.append((len(n), n))
```

```
word_len.sort()
```

```
return word_len[-1][0], word_len[-1][1]
```

```
result = find_longest_word(["PHP", "Exercises", "Backend"])
```

```
print("\nLongest word: ", result[1])
```

```
print("Length of the longest word: ", result[0])
```

а) Длина строки цифрами и прописью

б) Длина самого длинного слова строки и само слово

в) Длина самого короткого слова и само слово

г) Самое длинное слово и длина строки

Тема 4. Функции и методы для работы со строками

1. Компонент Combobox-это

1) упорядоченная совокупность взаимосвязанных элементов, являющихся текстовыми строками

2) контейнер, в котором можно размещать другие элементы управления

3) многострочный редактор

4) переключатель с зависимой фиксацией

2. Для работы с комбинированным списком в Delphi служит компонент

1) Combobox

2) ListBox

3) Radiogroup

4) Stringgid

5) Checklistbox

3. Свойство Style типа TComboboxstyle определяет

1) внешний вид и поведение комбинированного списка

2) свойства комбинированного списка

3) стиль работы с комбинированным списком

4) цвет комбинированного списка

5) число записей в комбинированном списке

4. Свойство DropDownCount типа Integer определяет

1) количество строк, одновременно отображающиеся в раскрывающемся списке

2) номер выделенного элемента

3) число выделенных компонентов на форме

4) размеры выделенного компонента

5) раскрыт ли список

5. Свойство DroppedDown типа Boolean позволяет определить

1) раскрыт ли список

2) выделен ли список

3) число выделенных компонентов на форме

4) размеры выделенного компонента

5) количество удаленных записей

6. Отсчет элементов списка начинается с

1) с нуля

2) с единицы

3) с минус единицы

4) с двух

5) с минус двух

7. Какое свойство представляет собой массив строк

и определяет количество элементов списка и их содержимое

1) items

2) count

3) index

4) font

5) hint

8. Свойство count типа integer

1) задает число элементов в списке

2) определяет номер выделенного элемента

3) определяет количество удаленных записей

4) задает число выделенных компонентов на форме

5) определяет внешний вид и поведение комбинированного списка

9. Процедура Insert(Index:Integer; const S:String)

1) вставляет строку S на позицию с номером, заданным параметром Index

2) добавляет в конец списка строку

3) удаляет элемент с номером, заданным параметром Index

4) нет верного ответа

5) очищает список, удаляя все его элементы

10. Функция Add (const S:string)

1) вставляет строку S на позицию с номером, заданным параметром Index

2) добавляет в конец списка строку

3) удаляет элемент с номером, заданным параметром Index

4) нет верного ответа

5) очищает список, удаляя все его элементы

11.Процедура Delete(Index:Integer; const S:String)

1) вставляет строку S на позицию с номером, заданным параметром Index

2) добавляет в конец списка строку

3) удаляет элемент с номером, заданным параметром Index

4) нет верного ответа

5) очищает список, удаляя все его элементы

12.Процедура Clear

1) вставляет строку S на позицию с номером, заданным параметром Index

2) добавляет в конец списка строку

3) удаляет элемент с номером, заданным параметром Index

4) нет верного ответа

5) очищает список, удаляя все его элементы

13. Процедура IndexOf(const S:string):integer

1) определяет, содержится ли в списке строка S

2) вставляет строку S на позицию с номером, заданным параметром Index

3) добавляет в конец списка строку

4) удаляет элемент с номером, заданным параметром Index

5) нет верного ответа

Тема 5. Списки. Операции над списками

1)Как создать список?

1)Все варианты верны

2)L = list(1, 2, 3)

3)l = [1, 2, 3]

4)l = list[1, 2, 3]

Ответ:

2) Что выведет этот код:

```
a = [ 1, 342, 223, 'Африка', 'Очки']
```

```
print(a[-3])
```

1)223

2)'Африка'

3)342

4)Ошибку

Ответ:

3) Что выведет этот код:

```
sample = [10, 20, 30]
```

```
sample.append(60)
```

```
sample.insert(3, 40)
```

```
print(sample)
```

1)[10, 20, 30, 40]

2)[10, 20, 30, 40, 60]

3)[10, 20, 30, 60, 40]

4)[60, 10, 20, 30, 40]

Ответ:

4) Что из перечисленного правда?

1)Элементы списка не могут повторяться

2)Все элементы списка должны быть одного типа

3)Мы можем вставить элемент на любую позицию в списке

4)Список не может содержать вложенных списков

Ответ:

5) Как получить ['bar', 'baz'] из списка

```
a = ['foo', 'bar', 'baz', 'qux', 'quux']
```

?

1)print(a[2:4])

2)print(a[1], a[2])

3)print(a[1:-2])

4)print(a[-4:-3])

5)print(a[2:3])

Ответ:

Тема 6. Кортежи, множества и диапазоны.

1)Что такое множество в Python?

1)Это любая коллекция элементов

2)Это список, содержащий в себе только функции

3)Это контейнер, значения в котором не повторяются

4)Это список, содержащий вложенные списки в себе

Ответ:

2) Каким образом правильно объявляется множество?

1)a = {}

2)a = []

3)a = set()

4)a = set

Ответ:

3) Чем отличаются методы remove() и discard(), применяемые к множеству?

1)Ничем

2)remove() удаляет элемент если он есть, но бросает ошибку если элемента нет. discard() просто удаляет элемент если он есть

3)discard() удаляет элемент если он есть, но бросает ошибку если элемента нет. remove() просто удаляет элемент если он есть

4)Метода discard() для множеств не существует

Ответ:

4) Что такое frozenset?

1)Множество, которое нельзя изменять

2)Множество, которое хранит в себе только неизменяемые объекты

3)Множество, которое используется для хранения констант

4)Выдумка нашей редакции

Ответ:

5) Что это за метод такой, set.difference(another_set)

1)Возвращает True, если два множества одинаковые, False если хотя бы один элемент не совпадает

2)Возвращает True, если два множества разные, False если хотя бы один элемент совпадает

3)Возвращает множество из элементов, которые встречаются только в множестве set

4)Возвращает множество из элементов, которые встречаются только в множестве another_set

Ответ:

Тема 7. Словари. Операции и методы для работы со словарями

1) Что означает ошибка TypeError: unhashable type?

1)Неверно задано значение

2)Тип данных нельзя использовать в роли ключа

3)Слишком большое значение

4)Ошибка синтаксиса

Ответ:

2) Какие типы данных нельзя использовать в роли ключа?

1)Список, словарь

2)Словарь, кортеж

3)Кортеж, число

4)Число, булево значение

Ответ:

3) Что выдаст этот код?

```
Another_dict = {'a': {'a': ['a']}}
```

```
Print(another_dict.pop('a') == another_dict.clear())
```

1)True

2)False

3)Ошибка

Ответ:

4) Каков будет результат кода?

```
Dict_1 = {'a': 10, 'b': 20}
```

```
Dict_2 = {'b': 20, 'c': 30}
```

```
Dict_1.update(dict_2)
```

```
Print(dict_1)
```

1){'a': 10}

2){'a': 10, 'b': 20}

3){'a': 10, 'b': 20, 'c': 30}

4){ 'b': 20, 'c': 30}

Ответ:

5) Что выведет этот код?

```
Old_dict = {'a' : 10, 'b' :10}
```

```
New_dict = {}
```

```
For i, j in old_dict.items():
```

```
New_dict[j] = i
```

```
Print(new_dict)
```

1){'a': 10, 'b': 10}

2){10: 'a', 10: 'b'}

3){'a': 10}

4){10: 'b'}

Ответ:

Тема 8. Работа с датой и временем

1) Определите что будет выведено после выполнения данного кода:

```
from datetime import datetime, timedelta
```

```
given_date = datetime(2020, 2, 25)
```

```
print("Given date")
```

```
print(given_date)
```

```
days_to_subtract = 7
```

```
res_date = given_date - timedelta(days=days_to_subtract)
```

```
print("New Date")
```

```
print(res_date)
```

а)Все даты прошлой недели

б)Все дни прошлой недели

в)Дата неделю назад

г)День недели неделю назад

2) Определите что будет выведено после выполнения данного кода:

```
from datetime import datetime
```

```
given_date = datetime(2020, 7, 26)
```

```
print(given_date.today().weekday())
```

```
print(given_date.strftime('%A'))
```

а)Проверка заданной даты на четность

б)День недели заданного числа

в)Проверка является ли заданная дата выходным

г)Является ли четной неделя, в которой находится заданная дата

3) Определите что будет выведено после выполнения данного кода:

```
from datetime import datetime
```

```
given_date = datetime(2020, 2, 25)
```

```
string_date = given_date.strftime("%Y-%m-%d %H:%M:%S")
```

```
print(string_date)
```

а)Заданная дата в измененном формате (другой внешний вид)

б)Заданная дата в типе str

в)Заданная дата через неделю

г)Данная дата в типе int для проведения вычислений

4) Определите что будет выведено в результате выполнения данного кода:

```

from datetime import datetime
date_1 = datetime(2020, 2, 25).date()
date_2 = datetime(2020, 9, 17).date()
delta = None
if date_1 > date_2:
    print("date_1 is greater")
    delta = date_1 - date_2
else:

```

```

    print("date_2 is greater")
    delta = date_2 - date_1

```

```

print("Difference is", delta.days, "days")

```

а)Количество дней между сегодняшним днем и указанными датами

б)Количество дней между указанными днями

в)Какая дата "старше" (какое число дальше от начала календаря)

г)Какая дата "младше" (какое число ближе к началу календаря)

****За начало календаря принимать Рождество Христово**

5) Определите что будет выведено в результате выполнения данного кода:

```

from datetime import datetime, timedelta
given_date = datetime(2020, 3, 22, 10, 00, 00)
print("Given date")
print(given_date)
days_to_add = 7
res_date = given_date + timedelta(days=days_to_add, hours=12)
print("New Date")
print(res_date)

```

а)Указанная дата с добавлением 1 недели и 12 часов

б)Указанная дата с добавлением 12 часов

в)Указанная дата с вычетом 1 недели и 12 часов

г)Указанная дата с вычетом 12 часов и добавлением одной недели

Тема 9. Пользовательские функции

1) Как можно вызвать метод func у следующего класса (выберите все подходящие варианты):

```

Class myclass:

```

```

    Def func(self):

```

```

        Print('hello')

```

1) myClass.func()

2) obj = myClass() obj.func()

3) obj = myClass() myClass.func(obj)

4) obj = myClass() obj.func

5) ни один из перечисленных

Ответ:

2) Что напечатает следующий код:

```

def func(n):

```

```

    n = n + 1

```

```

print(func(0))

```

1)0

2)1

3)func(0)

4)None

5) возникнет ошибка

Ответ:

3) Что выведет следующий код:

`a = 3`

`a = "foo" if a / 2 == 1 else 2`

`a = a + a`

`print (a)`

1) 6

2) Возникнет ошибка

3) 2

4) 4

5) foofoo

Ответ:

4) Что необходимо добавить на место пропущенной строки?

`def find_max(nums):`

`max_num = float("-inf")`

`for num in nums:`

`if num > max_num:`

`# пропущенная строка`

`return max_num`

1) `max_num = num`

2) `num = max_num`

3) `max_num += 1`

4) `max_num += num`

Ответ:

5) Каким будет результат выполнения кода:

`a = [1, 2, 3]`

`if a[2] < 3:`

`print(a[a[1]])`

`else:`

`print(a[1])`

1) 1

2) 2

3) 3

4) возникнет ошибка

Ответ:

Тема 10. Обработка исключений

1. Какое исключение возникнет при вводе кода:

`>>> 1/0`

1) `TypeError`

2) `ZeroDivisionError` +

3) `ValueError`

4) `NameError`

2. Какое выражение позволяет получить исключение и повторно поднять его?

1) `try`

2) `raise` +

3) `expect`

4) `traceback`

3. Что выведет данный код?

```
my_dict = {"a":1, "b":2, "c":3}
```

```
try:
```

```
    value = my_dict["a"]
```

```
except KeyError:
```

```
    print("A KeyError occurred")
```

```
else:
```

```
    print("No error occurred")
```

```
finally:
```

```
    print("The finally statement ran")
```

1) No error occurred

The finally statement ran +

2) A KeyError occurred

3) The finally statement ran

4) ValueError

4. Что выведет данный код при $a = 10$, $b = 5$?

```
try:
```

```
    a = input("Введите число: ")
```

```
    b = input("Введите еще одно число: ")
```

```
    a = int(a)
```

```
    b = int(b)
```

```
    print (a / b)
```

```
except ZeroDivisionError:
```

```
    print("b не может быть нулем!")
```

```
except ValueError:
```

```
    print("Ошибка ввода числа")
```

1) b не может быть нулем

2) NameError

3) 2 +

4) Ошибка ввода числа

5. Что выведет данный код при $a = 1$, $b = 0$?

```
try:
```

```
    print(a/b)
```

```
    print("Это не будет напечатано")
```

```
    print('10'+10)
```

```
except TypeError:
```

```
    print("Вы сложили значения несовместимых типов")
```

```
except ZeroDivisionError:
```

```
    print("Деление на 0")
```

1) Деление на 0 +

2) Вы сложили значения несовместимых типов

3) NameError

4) Это не будет напечатано

Тема 11. Работа с файлами и каталогами

Вопрос 1. Какими способами можно открыть текстовый файл (в формате .txt) в Python (без использования сторонних библиотек)?

1) `for`

2) open +

3) File

4) String

Вопрос 2. Дан файл «article.txt» со следующим содержимым:

Вечерело

Жужжали мухи

Светил фонарик

Кипела вода в чайнике

Венера зажглась на небе

Деревья шумели

Тучи разошлись

Листва зеленела

Что будет выведено в результате выполнения следующего кода:

```
def read_last(lines, file):
```

```
    if lines > 0:
```

```
        with open(file, encoding='utf-8') as text:
```

```
            file_lines = text.readlines()[-lines:]
```

```
        for line in file_lines:
```

```
            print(line.strip())
```

```
    else:
```

```
        print('Количество строк может быть только целым положительным')
```

```
read_last(-5, 'article.txt')
```

1) Деревья шумели

2) Тучи разошлись

3) Количество строк может быть только целым положительным +

4) Листва зеленела

Вопрос 3. Функция open() позволяет как читать, так и записывать данные в файл. Существует несколько режимов, которые передаются вторым параметром.

Для чего используется параметр "a":

1) одновременные режимы чтения и записи;

2) возможность дозаписывать содержимое документа;+

3) возможность записи в файл (все старые данные будут уничтожены), а если его не существует, он то предварительно будет создан;

4) открытие документа в байтовом режиме.

Вопрос 4. Документ «article.txt» содержит следующий текст:

Вечерело

Жужжали мухи

Светил фонарик

Кипела вода в чайнике

Венера зажглась на небе

Деревья шумели

Тучи разошлись

Листва зеленела

Что будет выведено в результате выполнения следующего кода:

```
def longest_words(file):
```

```
    with open(file, encoding='utf-8') as text:
```

```
        words = text.read().split()
```

```
        max_length = len(max(words, key=len))
```

```
        sought_words = [word for word in words if len(word) == max_length]
```

```

if len(sought_words) == 1:
    return sought_words[0]
return sought_words
print(longest_words('article.txt'))

```

1) разошлись +

2) Жужжали

3) водаТучи

4) зажглась

Вопрос 5. Какая команда используется для записи в файл?

1) fout

2) write +

3) writes

4) PrintWriter

Тема 12. Доступ из Python к базам данных SQLite

1).Какие типы данных поддерживает SQLite?

1. Null, integer, strings, boolean

2. Integer, float, strings, text

3. Null, text, real, integer, blob +

4. Blob, real, strings, boolean

2).Какая команда создает объект, который позволяет нам взаимодействовать с базой данных и добавлять записи?

1. conn.commit

2. cursor.execute +

3. cursor.executable

4. conn.cursor

3). Что выполняет данная часть кода?

```
import sqlite3
```

```
from sqlite3 import Error
```

```
def create_connection(path):
```

```
    connection = None
```

```
    try:
```

```
        connection = sqlite3.connect(path)
```

```
        print("Connection to SQLite DB successful")
```

```
    except Error as e:
```

```
        print(f"The error '{e}' occurred")
```

```
    return connection
```

1. Импорт базы данных в SQLite

2. Ищет путь к базе данных на ПК

3. Подключает к SQLite базе данных +

4. Данный код не выполняет никаких задач

4). База данных, помещенная в оперативную память ПК хранится:

1. Сутки

2. До перезагрузки ПК или смерти процесса +

3. Бессрочно

4. Пока не будет перенесена на сервер

5).Какая команда используется для создания базы данных в python?

1) sqlite3.conn

2) sqlite3.connect +

- 3) connect
- 4) sqllite3.connect

Тема 13. Доступ из Python к базам данных MySQL

1. Что выполняет следующий код?

```
import mysql.connector
mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    password="yourpassword",
    database="mydatabase"
)
mycursor = mydb.cursor()
sql = "SELECT * FROM customers WHERE address LIKE '%way%'"
mycursor.execute(sql)
myresult = mycursor.fetchall()
for x in myresult:
    print(x)
```

- +1) Поиск в таблице address данных имеющих подстроку way и выводит их на экран
- 2) Поиск в таблице address данных имеющих подстроку %way%
- 3) Замена в таблице address всех данных на строку way
- 4) Изменение названия таблицы address на way

2. Что выполняет следующий код?

```
>>> show_db_query = "SHOW DATABASES"
>>> with connection.cursor() as cursor:
...     cursor.execute(show_db_query)
...     for db in cursor:
...         print(db)
```

- 1) Выводит имена таблиц базы данных находящихся на вашем сервере MySQL
- +2) Выводит имена всех баз данных находящихся на вашем сервере MySQL
- 3) Данные таблиц из выбранных баз данных
- 4) Выводит все таблицы имеющиеся на сервере MySQL

3. Что выполняет следующий код?

```
from getpass import getpass
from mysql.connector import connect, Error
try:
    with connect(
        host="localhost",
        user=input("Enter username: "),
        password=getpass("Enter password: "),
    ) as connection:
        create_db_query = "CREATE DATABASE online_movie_rating"
        with connection.cursor() as cursor:
            cursor.execute(create_db_query)
except Error as e:
    print(e)
```

- 1) Создает таблицу с названием online_movie_rating
- +2) Создает базу данных с названием online_movie_rating
- 3) Изменяет существующую базу данных с названием online_movie_rating

4) Удаляет базу данных с названием online_movie_rating

4. Что выполняет следующий код?

```
import mysql.connector
mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    password="yourpassword",
    database="mydatabase"
)
mycursor = mydb.cursor()
mycursor.execute("SELECT * FROM customers")
myresult = mycursor.fetchall()
for x in myresult:
    print(x)
```

1) Удаляет все данные из базы данных customers

+2) Выводит все данные из базы данных customers

3) Удаляет все значения с сохранением таблицы базы данных customers

4) Создает базу данных с названием customers

5. Что выполняет следующий код?

```
import mysql.connector
mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    password="yourpassword",
    database="mydatabase"
)
mycursor = mydb.cursor()
sql = "SELECT * FROM customers WHERE address ='Sovetskaya 38'"
mycursor.execute(sql)
myresult = mycursor.fetchall()
for x in myresult:
    print(x)
```

+1) Выводит все данные из таблицы, в которых адрес Sovetskaya 38

2) Удаляет все данные из таблицы, в которых адрес Sovetskaya 38

3) Если адрес Sovetskaya 38, то данные удаляются и присваивается значению none

4) Удаляет базу customers и создает новую с названием address

Тема 14. Работа с графикой

1. Какой командой подключается библиотека, позволяющая работать с черепашкой графикой в Python

1) import Черепашка

2) import Turtle

3) import turtle

4) import robot

Ответ:

2. Какая команда позволит Черепашке передвинуться вперед на 50 пикселей

1) turtle.forward(50)

2) turtle.forward(150)

3) turtle.go(50)

4) turtle.go(150)

Ответ:

3. Имеется программа. Что она нарисует?

```
Import turtle
```

```
Turtle.forward(100)
```

```
Turtle.right(90)
```

```
Turtle.forward(50)
```

```
Turtle.right(90)
```

```
Turtle.forward(100)
```

```
Turtle.right(90)
```

```
Turtle.forward(50)
```

1) Прямоугольник

2) Квадрат

3) Две линии, выходящие из одной точки с углом между ними 90 градусов

4) Ромб

Ответ:

4. В начале программы даны описания. Черепашка ориентирована вправо. Какие команды можно использовать, чтобы развернуть Черепашку вверх?

```
Import turtle
```

```
Def f(x) :
```

```
Turtle.forward(x)
```

```
Def lt(g):
```

```
Turtle.left(g)
```

```
Def rt(g):
```

```
Turtle.right(g)
```

1) rt(90)

2) lt(90)

3) lt(90)

4) rt(270)

Ответ:

5. В начале программы даны описания. Как можно выполнить поворот направо на 90 градусов?

```
Import turtle
```

```
Def f(x) :
```

```
Turtle.forward(x)
```

```
Def lt(g):
```

```
Turtle.left(g)
```

```
Def rt(g):
```

```
Turtle.right(g)
```

1) rt(90)

2) lt(90)

3) f(90)

Ответ:

Тема 15. Определение класса и создание экземпляра класса

1) Какое из представленных слов лучше всего подходит для названия класса?

1) Runner

2) Running

3) Runners

4) Run

5) Параметры функции range могут быть отрицательными целыми числами

Ответ:

2) Какое из следующих утверждений описывает эту строку?

`xyz = Circle()`

1) Все не верно

2) создается экземпляр класса

3) переменной присваивается значение класса

4) создается класс

Ответ:

3) Как написать атрибут класса?

1) `def some_atr(self): ...`

2) `self.some_atr = ...`

3) `some_atr = ...`

4) `class SomeAtr: ...`

Ответ:

4) Как вызвать метод `swim(5)` у экземпляра `hero`?

1) `hero['swim'] = 5`

2) `hero.swim(5)`

3) `hero(swim(5))`

4) `hero(swim, 5)`

Ответ:

5) Что происходит при наследовании `Apple` от `Fruit`?

1) `Apple` перезаписывает класс `Fruit`

2) В `Fruit` доступны все атрибуты класса `Apple`

3) `Fruit` перезаписывает класс `Apple`

4) В `Apple` доступны все атрибуты класса `Fruit`

Ответ:

Тема 16. Принципы объектно-ориентированного программирования

1) Что относится к основным принципам ООП?

1) Инкапсуляция, полиморфизм, наследование, абстракция

2) Инкапсуляция, полиморфизм, делегирование, абстракция

3) Полиморфизм, разделение интерфейса, наследование, абстракция

4) Инкапсуляция, наследование, абстракция, открытость/закрытость

Отвте:

2) Что будет выведено на экран?

`Class test:`

`Test = None`

`Print(Test.test)`

1) 0

2) False

3) None

4) Ошибка

Ответ:

3) Какой принцип ООП описывает следующее предложение? Этот принцип является способностью использовать общий интерфейс для нескольких форм (типов данных).

1) Инкапсуляция

2) Полиморфизм

3) Абстракция

4) Наследование

Ответ:

4) Какой из перечисленных вариантов является верным объявлением `private` поля?

1) `private field = 0`

2) `field = 0`

3) `_field = 0`

4) `__field = 0`

Ответ:

5) Как создать конструктор класса `A`?

1) `A(параметры конструктора)`

2) `def __init__(параметры конструктора)`

3) `def __A__(параметры конструктора)`

4) `def init(параметры конструктора)`

Ответ:

Тема 17. Статические методы и методы класса

Вопрос 1. С помощью какого атрибута можно получить доступ к атрибутам класса и возможности менять состояние самого класса?

1) `self`

2) `classmethod`

3) `self.__class__ +`

4) `class.self`

Вопрос 2. С помощью чего декларируются статические методы?

1) `self`

2) `classmethod`

3) `staticmethod +`

4) `staticselfmethod`

Вопрос 3. Что будет выведено в результате работы данного кода:

```
from datetime import date
```

```
class Person:
```

```
    def __init__(self, name, age):
```

```
        self.name = name
```

```
        self.age = age
```

```
@classmethod
```

```
    def from_birth_year(cls, name, year):
```

```
        return cls(name, date.today().year - year)
```

```
@staticmethod
```

```
    def is_adult(age):
```

```
        return age > 18
```

```
person1 = Person('Sarah', 25)
```

```
person2 = Person.from_birth_year('Roark', 1994)
```

```
>>> person2.name, person2.age
```

1) `True`

2) `Roark 24 +`

3) `False`

4) `Sarah 25`

Вопрос 4. Что будет выведено в результате работы данного кода:

```
class Soda:
```

```
    def __init__(self, ingredient):
```

```

if isinstance(ingredient, str):
    self.ingredient = ingredient
else:
    self.ingredient = None
def show_my_drink(self):
    if self.ingredient:
        print(f'Газировка и {self.ingredient}')
    else:
        print('Обычная газировка')
drink1 = Soda('малина')
drink1.show_my_drink()

```

- 1) Обычная газировка
- 2) Газировка и малина +
- 3) None
- 4) малина

Вопрос 5. Укажите набор атрибутов, которые считаются общедоступными, для экземпляров следующего класса:

```

class Example:
    def __init__(self, x, y):
        xy = x, y
        self.position = xy
        self._length = self.__len(x, y)
    def __len__(self, x, y):
        return abs(x) + abs(y)
    def getlen(self):
        return self._length

```

- 1) getlen, _length, position, __len, xy
- 2) getlen, position +
- 3) getlen, _length, position
- 4) position

Тема 18. Основы разработки оконных приложений

7) Основы разработки оконных приложений

1) Что будет выведено в заголовке окна приложения? (3)

```
from tkinter import *
```

```

window = Tk()
window.title("Приложение PythonRu")
lbl = Label(window, text="Привет, пользователь!")
lbl = Label(window, text="Добро пожаловать в приложение PythonRu!")
lbl.grid(column=0, row=0)
window.mainloop()

```

Ответ: 1)Привет, пользователь! 2)Приложение PythonRu 3)Добро пожаловать в приложение PythonRu!

2) Как правильно установить окно шириной 400x255? (4)

Ответ: 1)window.title("400x250") 2)btn.grid(column=400, row=255) 3)lbl.grid(column=400, row=255)
4)window.geometry('400x250')

3) Перечислите, правильные варианты ответа. (1,2)

```
from tkinter import *
```

```

window = Tk()
window.title("Добро пожаловать в приложение PythonRu")
window.geometry('400x250')
menu = Menu(window)
new_item = Menu(menu)
new_item.add_command(label='Новый')
new_item.add_separator()
menu.add_cascade(label='Файл', menu=new_item)
window.config(menu=menu)
window.mainloop()

```

Ответ: 1)Задаст панели меню название "Файл" 2)Создаст панель "Меню" 3)Изменит название панели меню с "Файл" на "Новый"

4)Какие библиотеки используются для разработки графических пользовательских интерфейсов? (1)

Ответы: 1) Tkinter, PyQt 2) Django, Tkinter 3) CPython, Jython 4) Pyforms, PyPy

5)Укажите правильный вариант ответа: (3)

```

from tkinter import*
root=Tk()
root.title("Мое первое приложение")
root.geometry('800x600')
root.config(bg='red')
t1=Label(root, text='text in windows', fg='yellow', bg='grey')
t1.config(font=('Times', 25))
t1.pack()
b1=Button(root, text='Click my')
b1.config(width=20, height=2, bg='green', fg='blue')
b1.pack()
root.mainloop()

```

Ответы:

1)Заголовок окна приложения будет зеленым

2)Будет создана кнопка с цветом "Желтый"

3)Размер окна приложения будет 800x600

4)Название кнопки будет "Click me"

4.3 Промежуточная аттестация по дисциплине проводится в форме экзамена

Типовые вопросы экзамена (ОПК-7)

1. Основные характеристики и критерии оценки алгоритмов. Данные. Понятие типа данных.
2. Понятие типа данных. Константы. Переменные.
3. Основные характеристики и критерии оценки алгоритмов. Целочисленные типы данных.
4. Вещественные типы данных.
5. Основные характеристики и критерии оценки алгоритмов. Символьные типы данных.
6. Булевские типы данных.
7. Определение новых типов данных.
8. Основные характеристики и критерии оценки алгоритмов. Перечисляемые типы данных.
9. Интервальные типы данных.
10. Временной тип данных.
11. Операции. Выражения. Арифметические операции.

- 12 Операции. Операции отношения.
- 13 Операции. Булевские операции.
- 14 Основные характеристики и критерии оценки алгоритмов. Оператор присваивания.
- 15 Оператор ветвления if.
- 16 Оператор ветвления case.
- 17 Операторы повтора — циклы. Оператор повтора for.
- 18 Операторы повтора — циклы. Оператор повтора repeat
- 19 Операторы повтора — циклы. Оператор повтора while
20. Процедуры. Понятие. Свойства. Параметры.
21. Функции. Понятие. Свойства. Параметры.
22. Параметры процедур и функций.
23. Рекурсивные подпрограммы.
24. Строковые переменные.
25. Операции над строками.
26. Форматы кодирования символов.
27. Стандартные процедуры и функции для работы со строками.
28. Объявление массива.
29. Работа с массивами.
30. Понятие файла.
31. Работа с файлами.
32. Стандартные подпрограммы управления файлами.

Типовые задания для экзамена (ОПК-7)

Не предусмотрено

4.4. Шкала оценивания промежуточной аттестации

Оценка	Компетенции	Дескрипторы (уровни) – основные признаки освоения (показатели достижения результата)
«отлично» (85 - 100 баллов)	ОПК-7	Имеет адекватное представление об основных языках программирования, современных программных средах разработки информационных систем и технолог
«хорошо» (70 - 84 баллов)	ОПК-7	Имеет представление об основных языках программирования, современных программных средах разработки информационных систем и технолог
«удовлетворительно» (50 - 69 баллов)	ОПК-7	Имеет представление об языках программирования, современных программных средах разработки информационных систем и технологий
«неудовлетворительно» (менее 50 баллов)	ОПК-7	Не имеет представление об основных языках программирования, современных программных средах разработки информационных систем и технолог

5. Методические указания для обучающихся по освоению дисциплины (модуля)

5.1 Методические указания по организации самостоятельной работы обучающихся:

Приступая к изучению дисциплины, в первую очередь обучающимся необходимо ознакомиться содержанием рабочей программы дисциплины (РПД), которая определяет содержание, объем, а также порядок изучения и преподавания учебной дисциплины, ее раздела, части.

Для самостоятельной работы важное значение имеют разделы «Объем и содержание дисциплины», «Учебно-методическое и информационное обеспечение дисциплины» и «Материально-техническое обеспечение дисциплины, программное обеспечение, профессиональные базы данных и информационные справочные системы».

В разделе «Объем и содержание дисциплины» указываются все разделы и темы изучаемой дисциплины, а также виды занятий и планируемый объем в академических часах.

В разделе «Учебно-методическое и информационное обеспечение дисциплины» указана рекомендуемая основная и дополнительная литература.

В разделе «Материально-техническое обеспечение дисциплины, программное обеспечение, профессиональные базы данных и информационные справочные системы» содержится перечень профессиональных баз данных и информационных справочных систем, необходимых для освоения дисциплины.

5.2 Рекомендации обучающимся по работе с теоретическими материалами по дисциплине

При изучении и проработке теоретического материала необходимо:

- просмотреть еще раз презентацию лекции в системе MOODLe, повторить законспектированный на лекционном занятии материал и дополнить его с учетом рекомендованной дополнительной литературы;
- при самостоятельном изучении теоретической темы сделать конспект, используя рекомендованные в РПД источники, профессиональные базы данных и информационные справочные системы;
- ответить на вопросы для самостоятельной работы, по теме представленные в пункте 3.2 РПД.
- при подготовке к текущему контролю использовать материалы фонда оценочных средств (ФОС).

5.3 Рекомендации по работе с научной и учебной литературой

Работа с основной и дополнительной литературой является главной формой самостоятельной работы и необходима при подготовке к устному опросу на семинарских занятиях, к дебатам, тестированию, экзамену. Она включает проработку лекционного материала и рекомендованных источников и литературы по тематике лекций.

Конспект лекции должен содержать реферативную запись основных вопросов лекции, в том числе с опорой на размещенные в системе MOODLe презентации, основных источников и литературы по темам, выводы по каждому вопросу. Конспект может быть выполнен в рамках распечатки выдачи презентаций лекций или в отдельной тетради по предмету. Он должен быть аккуратным, хорошо читаемым, не содержать не относящуюся к теме информацию или рисунки.

Конспекты научной литературы при самостоятельной подготовке к занятиям должны содержать ответы на каждый поставленный в теме вопрос, иметь ссылку на источник информации с обязательным указанием автора, названия и года издания используемой научной литературы. Конспект может быть опорным (содержать лишь основные ключевые позиции), но при этом позволяющим дать полный ответ по вопросу, может быть подробным. Объем конспекта определяется самим студентом.

В процессе работы с основной и дополнительной литературой студент может:

- делать записи по ходу чтения в виде простого или развернутого плана (создавать перечень основных вопросов, рассмотренных в источнике);
- составлять тезисы (цитирование наиболее важных мест статьи или монографии, короткое изложение основных мыслей автора);
- готовить аннотации (краткое обобщение основных вопросов работы);
- создавать конспекты (развернутые тезисы).

5.4. Рекомендации по подготовке к отдельным заданиям текущего контроля

Собеседование предполагает организацию беседы преподавателя со студентами по вопросам практического занятия с целью более обстоятельного выявления их знаний по определенному разделу, теме, проблеме и т.п. Все члены группы могут участвовать в обсуждении, добавлять информацию, дискутировать, задавать вопросы и т.д.

Устный опрос может применяться в различных формах: фронтальный, индивидуальный, комбинированный. Основные качества устного ответа подлежащего оценке:

- правильность ответа по содержанию;
- полнота и глубина ответа;
- сознательность ответа;
- логика изложения материала;
- рациональность использованных приемов и способов решения поставленной учебной задачи;

- своевременность и эффективность использования наглядных пособий и технических средств при ответе;
- использование дополнительного материала;
- рациональность использования времени, отведенного на задание.

Устный опрос может сопровождаться презентацией, которая подготавливается по одному из вопросов практического занятия. При выступлении с презентацией необходимо обращать внимание на такие моменты как:

- содержание презентации: актуальность темы, полнота ее раскрытия, смысловое содержание, соответствие заявленной темы содержанию, соответствие методическим требованиям (цели, ссылки на ресурсы, соответствие содержания и литературы), практическая направленность, соответствие содержания заявленной форме, адекватность использования технических средств учебным задачам, последовательность и логичность презентуемого материала;
- оформление презентации: объем (оптимальное количество), дизайн (читаемость, наличие и соответствие графики и анимации, звуковое оформление, структурирование информации, соответствие заявленным требованиям), оригинальность оформления, эстетика, использование возможности программной среды, соответствие стандартам оформления;
- личностные качества: ораторские способности, соблюдение регламента, эмоциональность, умение ответить на вопросы, систематизированные, глубокие и полные знания по всем разделам программы;
- содержание выступления: логичность изложения материала, раскрытие темы, доступность изложения, эффективность применения средств ИКТ, способы и условия достижения результативности и эффективности для выполнения задач своей профессиональной или учебной деятельности, доказательность принимаемых решений, умение аргументировать свои заключения, выводы.

6. Учебно-методическое и информационное обеспечение дисциплины

6.1 Основная литература:

1. Лубашева, Т. В., Железко, Б. А. Основы алгоритмизации и программирования : учебное пособие. - 2022-08-04; Основы алгоритмизации и программирования. - Минск: Республиканский институт профессионального образования (РИПО), 2016. - 379 с. - Текст : электронный // IPR BOOKS [сайт]. - URL: <http://www.iprbookshop.ru/67689.html>
2. Агафонов Е. Д., Ващенко Г. В. Прикладное программирование : учебное пособие. - Красноярск: Сибирский федеральный университет, 2015. - 112 с. - Текст : электронный // ЭБС «Университетская библиотека онлайн» [сайт]. - URL: <http://biblioclub.ru/index.php?page=book&id=435640>
3. Белоцерковская И. Е., Галина Н. В., Катаева Л. Ю. Алгоритмизация. Введение в язык программирования C++. - 2-е изд., испр.. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 197 с. - Текст : электронный // ЭБС «Университетская библиотека онлайн» [сайт]. - URL: <http://biblioclub.ru/index.php?page=book&id=428935>
4. Колокольникова А. И., Таганов Л. С. Информатика: 630 тестов и теория : пособие. - Москва: Директ-Медиа, 2014. - 429 с. - Текст : электронный // ЭБС «Университетская библиотека онлайн» [сайт]. - URL: <http://biblioclub.ru/index.php?page=book&id=236489>

6.2 Дополнительная литература:

1. Седжвик Р. Алгоритмы на C++. - 2-е изд., испр.. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 1773 с. - Текст : электронный // ЭБС «Университетская библиотека онлайн» [сайт]. - URL: <http://biblioclub.ru/index.php?page=book&id=429164>
2. Сеницын, С. В., Хлытчиев, О. И. Основы разработки программного обеспечения на примере языка C. - 2021-01-23; Основы разработки программного обеспечения на примере языка C. - Москва: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. - 211 с. - Текст : электронный // IPR BOOKS [сайт]. - URL: <http://www.iprbookshop.ru/73700.html>

3. Алексеев Е. Р., Злобин Г. Г., Костюк Д. А., Чеснокова О. В., Чмыхало А. С. Программирование на языке С++ в среде Qt CreaTo. - 2-е изд., испр.. - Москва: Национальный Открытый Университет «ИНТУИТ», 2016. - 716 с. - Текст : электронный // ЭБС «Университетская библиотека онлайн» [сайт]. - URL: <http://biblioclub.ru/index.php?page=book&id=428929>

6.3 Иные источники:

1. Портал "Гуманитарное образование" - <http://www.humanities.edu.ru/>
2. Федеральный портал "Российское образование" - <http://www.edu.ru/>
3. Федеральное хранилище «Единая коллекция цифровых образовательных ресурсов» - <http://school-collection.edu.ru/>
4. Электронная библиотека социологического факультета МГУ имени М.В. Ломоносова - <http://lib.socio.msu.ru/l/library>
5. Электронная версия «Социологического журнала», издаваемого Российской академией наук Институтом социологии РАН - www.nir.ru/socio/scipubl/socjour.htm
6. Журнал «Социологические исследования» - <http://socis.isras.ru/>

7. Материально-техническое обеспечение дисциплины, программное обеспечение, профессиональные базы данных и информационные справочные системы

Для проведения занятий по дисциплине необходимо следующее материально-техническое обеспечение: учебные аудитории для проведения занятий лекционного и семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, помещения для самостоятельной работы.

Учебные аудитории и помещения для самостоятельной работы укомплектованы специализированной мебелью и техническими средствами обучения, служащими для представления учебной информации большой аудитории.

Помещения для самостоятельной работы укомплектованы компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа в электронную информационно-образовательную среду Университета.

Для проведения занятий лекционного типа используются наборы демонстрационного оборудования, обеспечивающие тематические иллюстрации (проектор, ноутбук, экран/ интерактивная доска).

Лицензионное и свободно распространяемое программное обеспечение:

Операционная система "Альт Образование"

Microsoft Windows 10

Профессиональные базы данных и информационные справочные системы:

1. Электронный каталог Фундаментальной библиотеки ТГУ. – URL: <http://biblio.tsutmb.ru/elektronnyij-katalog>
2. Университетская библиотека онлайн: электронно-библиотечная система. – URL: <https://biblioclub.ru>
3. Консультант студента. Гуманитарные науки: электронно-библиотечная система. – URL: <https://www.studentlibrary.ru>
4. Научная электронная библиотека eLIBRARY.ru. – URL: <https://elibrary.ru>
5. Российская государственная библиотека. – URL: <https://www.rsl.ru>
6. Российская национальная библиотека. – URL: <http://nlr.ru>
7. Президентская библиотека имени Б.Н. Ельцина. – URL: <https://www.prilib.ru>
8. Научная электронная библиотека Российской академии естествознания. – URL: <https://www.monographies.ru>
9. Электронная библиотека РФФИ. – URL: <https://www.rfbr.ru/rffi/ru/library>

https://auth.tsutmb.ru/authorize?response_type=code&client_id=moodle&state=xyz

Взаимодействие преподавателя и студента в процессе обучения осуществляется посредством мультимедийных, гипертекстовых, сетевых, телекоммуникационных технологий, используемых в электронной информационно-образовательной среде университета.