ОГЛАВЛЕНИЕ

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 7. Алгоритмы сортировки и поиска. Миронова И.В. 2020**](#_t4igwuqgz7k) **4**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.15. Работа со словарем. Миронова И.В. 2020**](#_s9h44l31cumx) **6**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.4. Условный оператор. Миронова И.В. 2020**](#_htf7m3ibr0o3) **9**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.8. Методы строк. Миронова И.В. 2020**](#_rhdi5pjs8ymv) **11**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.1. Атрибуты. Миронова И.В. 2020**](#_f3i4jqordnlx) **12**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.3. Функции с произвольным числом параметров. Миронова И.В. 2020**](#_xfojdak7cqbi) **16**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 5. Элементы функционального программирования. 5.2. Сортировка с параметром key. Миронова И.В. 2020**](#_ta0alaiomsyb) **18**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 4. Использование библиотек. Миронова И.В. 2020**](#_gea2xpb1759u) **19**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.4. Наследование. Миронова И.В. 2020**](#_2h4tezygmjo1) **22**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.1. Арифметические выражения и операции. Миронова И.В. 2020**](#_25qe7k5453n6) **26**

[Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.1. Передача параметров в функцию. Миронова И.В. 2020](#_g4y71sm1auw) **28**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.3. Методы. Миронова И.В. 2020**](#_csqkii9bte0r) **30**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.9. Операции над списками. Миронова И.В. 2020**](#_bliy9w6lawxy) **32**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.4. Инструкция import. Миронова И.В. 2020**](#_en1ri8611dk2) **34**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.2. Ввод-вывод, форматирование. Миронова И.В. 2020**](#_kis4p0q2ljtk) **36**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.6. Циклы. Миронова И.В. 2020**](#_43t11wlpw9u9) **38**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 1. Языки и технологии программирования. Миронова И.В. 2020**](#_c4trpylobcl0) **40**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 5. Элементы функционального программирования. 5.1. Функция в качестве параметра. Миронова И.В. 2020**](#_eh7371p9n0lz) **42**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.5. Функции range() и enumerate(). Миронова И.В. 2020**](#_8eex86i6wecx) **45**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 8. Структуры данных. Миронова И.В. 2020**](#_h1y7f3ud6g3u) **46**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.11. Преобразования между списками и строками. Миронова И.В. 2020**](#_m1dyzb93fj61) **48**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.5. Термины. Миронова И.В. 2020**](#_cgplrgjysay4) **50**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.12. Кортежи. Миронова И.В. 2020**](#_o9fbsmwchufl) **51**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 5. Элементы функционального программирования. 5.3. Функции map(), filter(), reduce(). Миронова И.В. 2020**](#_t925zjtrfwbp) **53**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.3. Логические выражения. Миронова И.В. 2020**](#_6cw00vkzt4b4) **54**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.16. Множества. Миронова И.В. 2020**](#_hg5j0l8k1zj8) **56**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 5. Элементы функционального программирования. 5.4. Декораторы, итераторы, генераторы. Миронова И.В. 2020**](#_kb4jtok6e3f) **58**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.13. Создание словарей. Миронова И.В. 2020**](#_aqhhrxnas4zm) **60**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.2. Список в качестве параметра функции. Миронова И.В. 2020**](#_jfey0yhiq5mk) **62**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.14. Изменение словарей. Миронова И.В. 2020**](#_t998vfz3wdou) **65**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.10. Методы списков. Миронова И.В. 2020**](#_jgi88j5dpak8) **67**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.2. Свойства. Миронова И.В. 2020**](#_nl7ro885293u) **68**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.7. Операции для строк. Миронова И.В. 2020**](#_a04kh0a3z3l8) **72**

[**Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.5. Исключения. Миронова И.В. 2020**](#_3tasnq88gue) **74**

[**ПМ экзамен**](#_fbild3488z98) **77**

### 

### 

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 7. Алгоритмы сортировки и поиска. Миронова И.В. 2020

Имеется список [7, 8, 11, 5, 1]. После выполнения одного прохода алгоритма сортировки выбором список будет иметь вид …

[5, 7, 8, 11, 1]

[5, 1, 7, 11, 8]

[1, 8, 5, 11, 7]

[7, 8, 5, 1, 11]

[7, 8, 1, 5, 11]

Временная сложность алгоритма быстрой сортировки:

в среднем 

всегда 

в среднем 

зависит от выбора длины шага

в наихудшем случае 

При сортировке не меняют взаимного расположения элементов с одинаковыми значениями алгоритмы:

сортировки вставками

сортировки Шелла

сортировки выбором

пузырьковой сортировки

быстрой сортировки

Имеется список [7, 8, 11, 5, 1]. После выполнения одного прохода алгоритма быстрой сортировки с разделяющим элементом 7 список будет иметь вид …

[1, 8, 5, 11, 7]

[7, 8, 5, 1, 11]

[5, 1, 7, 11, 8]

[5, 7, 8, 11, 1]

[7, 8, 1, 5, 11]

Сортировка, требующая определения значения, на основании которого массив разделяется на два подмассива, для которых выполняется некоторое свойство относительно этого числа, – это …

пузырьковая сортировка

сортировка Шелла

сортировка вставками

сортировка выбором

быстрая сортировка

Временная сложность алгоритма сортировки Шелла:

зависит от выбора длины шага

в наихудшем случае 

всегда 

в среднем 

всегда 

Имеется список [7, 8, 11, 5, 1]. После выполнения одного прохода алгоритма пузырьковой сортировки список будет иметь вид ……

[5, 1, 7, 11, 8]

[7, 8, 5, 1, 11]

[1, 8, 5, 11, 7]

[7, 8, 1, 5, 11]

[5, 7, 8, 11, 1]

Имеется список [7, 8, 11, 5, 1]. После выполнения одного прохода алгоритма сортировки Шелла с шагом 2 список будет иметь вид …

[7, 8, 1, 5, 11]

[7, 8, 5, 1, 11]

[5, 1, 7, 11, 8]

[1, 5, 7, 8, 11]

[5, 7, 8, 11, 1]

Имеется список [7, 8, 11, 5, 1]. После выполнения одного прохода алгоритма сортировки вставками список будет иметь вид …

[7, 8, 5, 1, 11]

[5, 1, 7, 11, 8]

[1, 8, 5, 11, 7]

[5, 7, 8, 11, 1]

[7, 8, 1, 5, 11]

Сортировка, требующая определения шага, который используется для формирования серий, внутри которых каждое значение отстоит от своих соседей на заданную длину шага, – это …

быстрая сортировка

сортировка Шелла

сортировка выбором

пузырьковая сортировка

сортировка вставками

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.15. Работа со словарем. Миронова И.В. 2020

После выполнения операторов

d = dict([[2, 11], [7, 14]])

s = 0

for v in d.items():

s += v

print(s)

на экран будет выведено …

9

34

25

0

сообщение об ошибке

После выполнения операторов

d = {8:2, 1:10, 3:12}

d1 = sorted(d)

print(d1)

на экран будет выведено …

[(1, 10), (2, 12), (8, 2)]

[2, 10, 12]

{1:10, 3:12, 8:2}

сообщение об ошибке

[1, 3, 8]

После выполнения операторов

d = dict([[3, 1], [5, 12]])

s = 0

for v in d:

s += v

print(s)

на экран будет выведено

8

После выполнения операторов

d1 = {'x': 1, 'y': 2}

d2 = {'x': 3, 'z': 4, 't': 5}

print(d2.keys() - d1.keys())

на экран будет выведено …

{'z': 4, 't': 5}

{'x': 2, 'z': 4, 't': 5}

сообщение об ошибке

{'z', 't'}

['z', 't']

После выполнения операторов

d = {1:11, 2:22}

s = 0

for v in d:

s += d.pop(v,1)

print(s)

на экран будет выведено …

3

сообщение об ошибке

33

23

12

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.4. Условный оператор. Миронова И.В. 2020

После выполнения фрагмента программы

a = '1'

if 0 <= a <= 10:

print("да")

else:

print("нет")

на экран будет выведено …

нет

сообщение об ошибке

ничего

None

да

В результате выполнения следующего фрагмента

a = b = [1]

c = [1]

print (a is b, a is c, a == c)

на экран будет выведено …

True False True

False False True

True True False

сообщение об ошибке

True True True

После выполнения фрагмента программы

a=1

if (a): print (a)

на экран будет выведено …

a

сообщение об ошибке

1

ничего

0

В результате выполнения следующего фрагмента

a = 1

b = 2

print (0 if a!=b else 1)

на экран будет выведено …

False

0

True

сообщение об ошибке

1

После выполнения фрагмента программы

a = 7; b = 3

if a > 5:

if b < 0:

print(a)

else:

print(b)

на экран будет выведено …

None

сообщение об ошибке

3

ничего

7

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.8. Методы строк. Миронова И.В. 2020

s = '456 45 345'

print(s.find('45', 2))

будет напечатано …

8

-1

0

4

сообщение об ошибке

После выполнения операторов

s = '\n\t ## '

print(f"1{s.strip()}0")

будет напечатано …

1 ## 0

сообщение об ошибке

1\n\t ## 0

1\n\t ##0

1##0

Имеется строка s = 'a 100 abc'. При работе с этой строкой правильными являются операторы:

s2 = s.replace('a', 'd', 2)

s1 = s.append(' 2')

s5 = s.count('2')

s4 = s.lower()

s3 = s.remove('0')

После выполнения операторов

s = '1011000'

s1 = s.remove('0', 2)

будет напечатано …

11100

сообщение об ошибке

10110

111000

101100

После выполнения операторов

s = '1.00000'

print(s.replace('00', '0'))

будет напечатано …

1.0

1.00

1.000

1.00000

1.0000

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.1. Атрибуты. Миронова И.В. 2020

Имеется класс и объект этого класса

class Class1:

def \_\_init\_\_(self):

self.\_\_v = 0

def pr(self):

print(self.\_\_v)

a = Class1()

…

a.pr()

Чтобы изменить значение атрибута \_\_v, вместо многоточия нужно вставить оператор …

a.\_\_v = 10

этот атрибут недоступен вне класса

Class1.\_\_v = 10

a.Class1.\_\_v = 10

a.\_Class1\_\_v = 10

После выполнения операторов

class Class1:

def \_\_init\_\_(self):

self.\_\_v = 0

def pr(self):

print(self.\_\_v)

a = Class1()

a.\_\_v = 10

a.pr()

на экран будет выведено …

0

10

ничего

None

сообщение об ошибке

После выполнения операторов class

class Class1:

b = 1

def \_\_init\_\_(self):

b = 2

el1 = Class1()

el2 = Class1()

el2.b = 3

Class1.b = 4

print(el1.b, el2.b, Class1.b)

на экран будет выведено …

сообщение об ошибке

4 3 4

2 3 4

1 3 4

4 4 4

После выполнения операторов

class Class1:

b = 3

def \_\_init\_\_(self):

self.b = 4

el = Class1()

Class1.b = 0

print(el.b)

на экран будет выведено …

0

None

4

сообщение об ошибке

3

После выполнения операторов

class Class1:

b = 5

def \_\_init\_\_(self):

b= 0

el = Class1()

print(el.b)

на экран будет выведено …

сообщение об ошибке

0

5

ничего

None

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.3. Функции с произвольным числом параметров. Миронова И.В. 2020

После выполнения операторов

def f(x, \*\*y):

print(x,y)

d = {'x':3, 'y':2}

f(5, \*\*d)

на экран будет выведено …

5 {'x': 3, y': 2}

сообщение об ошибке

3 {'x': 3, 'y': 2}

5 {'y': 2}

3 {'y': 2}

После выполнения операторов

def f(\*x, \*\*y):

print(x, y)

f(x=5, y=0)

на экран будет выведено …

(5) {'y': 0}

5 {'y': 0}

() {'x': 5, 'y': 0}

(1, 2) {}

сообщение об ошибке

Имеется функция

def f(\*x, y, z=0):

print(x, y + z)

Корректными вызовами функции являются:

f( z=3, y=4, 1, 2)

f(1, 2, z=3, y=4)

f(1, 2, y=3)

f(1, 2, 3)

f(1, 2, z=3)

После выполнения операторов

def f(x=10, \*y, z=10):

print(x, y, z)

ls = [5, 4, 3]

f(\*ls)

на экран будет выведено …

10 (5, 4, 3) 10

сообщение об ошибке

5 (4, 3) 10

5 4 3

10 (5, 4) 3

Имеется функция

def f(\*x, \*\*y):

print(x, y)

Корректными вызовами функции являются:

f(a=2, 3, 4)

f(1, 2, 3)

f(a=1)

f()

f(x=1, y=1)

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 5. Элементы функционального программирования. 5.2. Сортировка с параметром key. Миронова И.В. 2020

После выполнения операторов

arr = ['5.11.1999', '1.8.2006', '14.6.1980']

arr.sort(key=lambda x: x.split('.')[1])

print(arr)

на экран будет выведено …

сообщение об ошибке

['5.11.1999', '14.6.1980', '1.8.2006']

['14.6.1980', '5.11.1999', '1.8.2006']

['1.8.2006', '5.11.1999', '14.6.1980']

['14.6.1980', '1.8.2006', '5.11.1999']

После выполнения операторов

arr = [('яблоки', 95, 1), ('молоко', 77, 1), ('чипсы', 85, 2)]

arr.sort(key=lambda x: -x[1]\*x[2])

print(arr)

на экран будет выведено …

[('молоко', 77, 1), ('яблоки', 95, 1), ('чипсы', 85, 2)]

[('молоко', 77, 1), ('чипсы', 85, 2), ('яблоки', 95, 1)]

[('яблоки', 95, 1), ('чипсы', 85, 2), ('молоко', 77, 1)]

сообщение об ошибке

[('чипсы', 85, 2), ('яблоки', 95, 1), ('молоко', 77, 1)]

После выполнения операторов

arr = [(3, 0, 5), (1, 2, 3), (0, 3, 2)]

arr.sort(key=lambda x, y, z: x + y + z)

print(arr)

на экран будет выведено …

[(0, 3, 2), (3, 0, 5), (1, 2, 3)]

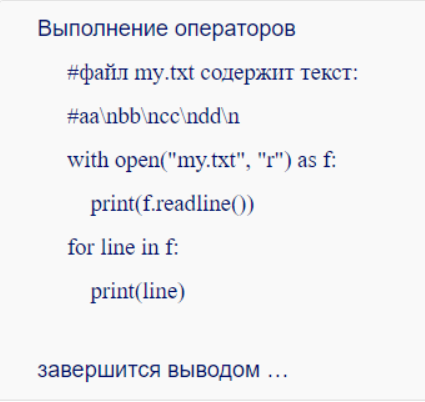
[(0, 3, 2), (1, 2, 3), (3, 0, 5)]

[(1, 2, 3), (3, 0, 5), (0, 3, 2)]

сообщение об ошибке

[(3, 0, 5), (1, 2, 3), (0, 3, 2)]

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 4. Использование библиотек. Миронова И.В. 2020



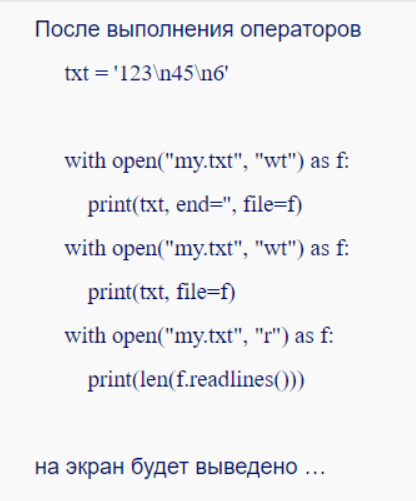
сообщения об ошибке

трех строк

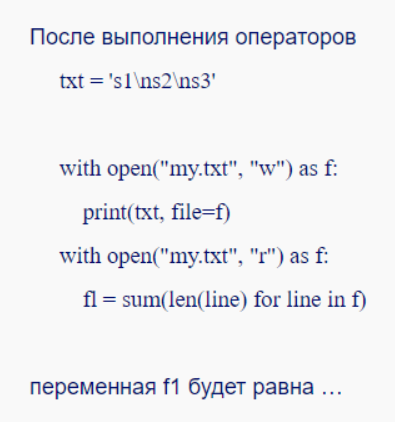
пяти строк

четырех строк

первой строки



3



9



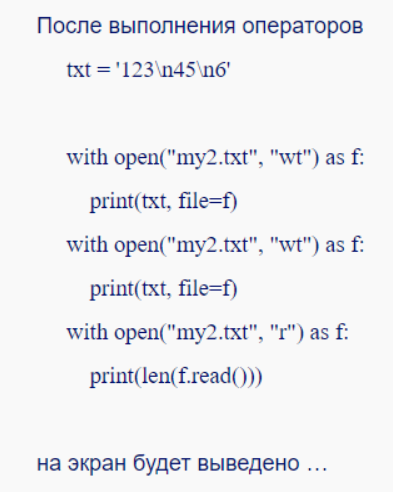
это словарь

содержит все пути для поиска модулей

это список

нельзя изменять из программы

можно изменять из программы с помощью методов



9



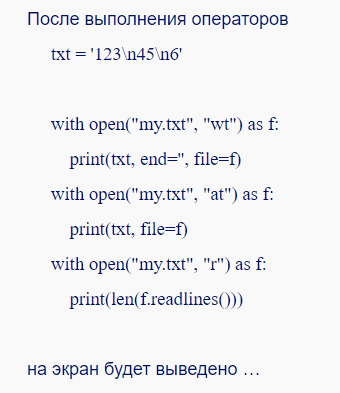
random()

rmdir()

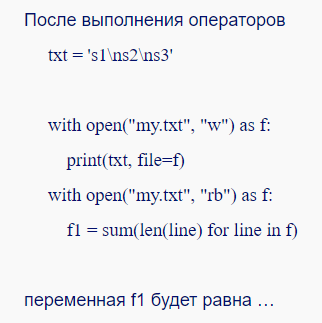
rename()

deepcopy()

chdir()



5



12



degrees()

round()

choice()

pow()

pi



SciPy

SymPy

Pandas

Matplotlib

NumPy

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.4. Наследование. Миронова И.В. 2020

После выполнения операторов

class Base:

def \_\_init\_\_(self):

self.a = 0

self.b = 1

def pr(self):

print(self.a, self.b)

class Child(Base):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.a = 5

x = Child()

x.pr()

на экран будет выведено …

5 1

0 None

0 1

сообщение об ошибке

5 None

Имеются классы

class Base:

b = 1

def \_\_init\_\_(self):

self.b = 0

class Child(Base):

def \_\_init\_\_(self, val):

…

Чтобы изменить значение атрибута b класса Base из конструктора класса Child, нужно вместо многоточия написать оператор …

super(self).b = val

super().b = val

Base.b = val

b = val

self.b = val

После выполнения операторов

class Base:

def \_\_init\_\_(self):

self.a = 0

self.b = 1

def pr(self):

print(self.a, self.b)

class Child(Base):

def \_\_init\_\_(self):

self.a = 5

x = Child()

x.pr()

на экран будет выведено …

сообщение об ошибке

5 None

0 1

5 1

0 None

Имеются классы

class Base:

def \_\_init\_\_(self):

self.\_\_v = 0

class Child(Base):

def \_\_init\_\_(self):

super().\_\_init\_\_()

…

Чтобы изменить значение атрибута \_\_v в базовом классе из конструктора производного класса, нужно вместо многоточия в методе \_\_init\_\_() написать оператор …

Base.\_\_v = 10

self.\_\_v = 10

self.\_Base\_\_v = 10

\_\_v = 10

атрибут в Child недоступен

После выполнения операторов

class Base:

def \_\_init\_\_(self):

self.a = 0

self.\_\_b = 1

def pr(self):

print(self.a, self.\_\_b)

class Child(Base):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.\_\_b = 5

x = Child()

x.pr()

на экран будет выведено …

сообщение об ошибке

0 1

None 5

0 5

None 1

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.1. Арифметические выражения и операции. Миронова И.В. 2020

После выполнения операторов

a = 2

b = a \* 2 % 3 + a \*\* 2

Значение переменной b равно

5

После выполнения операторов

a = 3

b = (5 - a % 4) \* a \*\* a // 3

Значение переменной b равно

18

После выполнения операторов

a = 4

b = 4 \* a \*\* 0.5 % 5

Значение переменной b равно

3

После выполнения операторов

a = 2

b = (a + a % 3) \* a \*\* a // 3

Значение переменной b равно

5

После выполнения операторов

a = 5

b = 1 - (-a // 3)

Значение переменной b равно

3

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.1. Передача параметров в функцию. Миронова И.В. 2020

После выполнения операторов

def m\_sum(x=1, y=1, z=1):

return 2 \* (x + y + z)

print(m\_sum(\*'123'))

на экран будет выведено …

123123

1231112311

сообщение об ошибке

250

12

Имеется функция

def my\_f(x, y='\*', z=0):

print(x, y, z) x = [1, 2]

Корректными вызовами этой функции являются:

my\_f()

my\_f(\*x)

my\_f(\*x, '7')

my\_f(10, \*x)

my\_f(\*\*x)

Имеется программа

def m\_sum(x=1, y=1, z=1):

return 2 \* (x + y + z)

ls = (1, 4)

d = {'x':2, 'z':2, 'v':2}

Корректными вызовами функции являются:

m\_sum(\*ls)

m\_sum()

m\_sum(y=5, x=0)

m\_sum(\*\*d)

m\_sum(, 2)

Имеется функция

def my\_f(x, y='\*', z=0):

print(x, y, z)

z = {'x':1, 'z':2}

Корректными вызовами этой функции являются:

my\_f(\*\*z, '\*')

my\_f(1, 2, 3)

my\_f(0, \*\*z)

my\_f(\*z)

my\_f(\*\*z)

Имеется функция

def my\_f(x, y=’\*’, z='2'):

print(a,b,c)

Корректными вызовами этой функции являются:

my\_f('...')

my\_f(y=7, z='+')

my\_f(z=2, x=3)

my\_f(1, 2, 3)

my\_f()

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.3. Методы. Миронова И.В. 2020

Имеется класс

class Class1:

def add1(self, p):

return p + 1

def pr(self, x):

a = …

print(a)

Чтобы вычислить значение a с помощью метода add1, нужно использовать оператор …

a = self.add1(x)

a = \_Class1\_\_add1(x)

a = add1(self, x)

a = add1(x)

a = Class1.add1(x)

Для метода \_\_repr\_\_ справедливы следующие утверждения. Метод:

должен возвращать строку

не имеет параметров (даже self)

автоматически вызывается при выводе объекта в интерактивной оболочке

преобразует объект в строку

вызывается при использовании функции str для объекта

Для метода \_\_init\_\_ справедливы следующие утверждения. Метод:

часто называют конструктором по аналогии с другими языками

вызывается автоматически при создании экземпляра класса

не имеет параметра self

обязательно должен быть реализован в классе

создает все атрибуты экземпляра класса и присваивает им значения

Имеется класс и объект этого класса

class Class1:

def met1(self, p):

print(p)

def met2 (p):

print(p)

@staticmethod

def met3(p):

print(p)

def \_met4(self, p):

print(p)

def \_\_met5(self, p):

print(p)

c = Class1()

Корректно выполнятся инструкции:

c.met2(1)

c.met1(1)

c.\_\_met5(1)

c.met3(1)

c.\_met4(1)

Для специальных (магических) методов справедливы следующие утверждения. Специальные методы:

всегда вызываются автоматически

не имеют параметра self

всегда имеют реализацию по умолчанию, которую можно не переопределять

имеют имена, которые начинаются и заканчиваются двумя символами подчеркивания

используются для реализации операторов языка

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.9. Операции над списками. Миронова И.В. 2020

После выполнения операторов

ls1 = ls2 = [0, 1]

ls1.insert(0, 3)

print(ls1 is ls2)

будет напечатано

True

После выполнения операторов

lst = [2, 3, 4, 5, 6, 7, 8, 9]

lst[5:7] = [0, 1]

print(lst)

будет напечатано …

[2, 3, 4, 5, 0, 1, 8, 9]

сообщение об ошибке

[2, 3, 4, 5, 6, 0, 1]

[2, 3, 4, 5, 6, 0, 1, 9]

[2, 3, 4, 0, 1, 8, 9]

После выполнения фрагмента

ls = [] \* 3

ls[1] = 1

print (ls)

будет напечатано …

сообщение об ошибке

[1, 1, 1]

[[], 1, []]

[[1], [1], [1]]

[1]

После выполнения фрагмента

ls = [[]] \* 3

ls[2].append(3)

ls[0] = 0

переменная A будет равна …

[0, 0, [3]]

[0, [], [3]]

[0, 0, 0]

[[3], [3], [3]]

[0, [3], [3]]

После выполнения операторов

ls1 = ls2 = [1, 2]

ls2 = ls2 + ls1

print(ls1 is ls2)

будет напечатано

False

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.4. Инструкция import. Миронова И.В. 2020

Операторы

x = [[1, 2], 3]

y= deepcopy(x)

выполнятся корректно, если предварительно выполнить одну из инструкций:

from copy import deepcopy

from copy import \*

import copy

import copy as \*

import deepcopy

Оператор

a = rd.randint(1, 10)

выполнится корректно, если предварительно выполнить инструкцию …

import random as rd

from random import \*

import \* from random as rd

from random as rd import random

import random

Оператор

a = math.factorial(5)

выполнится корректно, если предварительно выполнить одну из инструкций:

import math

import factorial from math

from math import factorial

import math as \*

from math import \*

Оператор

y = random.choice('0123456789')

выполнится корректно, если предварительно выполнить одну из инструкций:

import random from \*

from random import \*

from random import choice

import random

import random as \*

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.2. Ввод-вывод, форматирование. Миронова И.В. 2020

Строка "a = a \* 0.67" может быть выведена на экран с помощью инструкции …

print('a = a \* {:2f}'.format(2/3))

print('a = a \* {:3f}.format(2/3)')

print(f'a = a \* {2 / 3:.2}')

print(f'a = a \* {2 / 3:2.f}')

print('a = a \* ', 2 / 3)

После выполнения оператора

print("({:5})".format(4/3))

на экран будет выведено …

сообщение об ошибке

(1.33333)

(1.333)

(1.3333333333333333)

1.33333

После выполнения оператора

print(f"({1.11:6})")

на экран будет выведено …

(1.11 )

сообщение об ошибке

( 1.11)

(1.110000)

(1.1100)

После выполнения оператора

print(f"({'#':5}{4:6})")

на экран будет выведено …

( # 4)

( #4 )

сообщение об ошибке

(# 4 )

(# 4)

Переменная b равна 2.25. Тогда строка

2.25, 6.75

Могла быть выведена на экран с помощью операторов:

print(b, 3\*b, sep=', ')

print(b, ',', b\*3)

print(f'{b}, {3\*b}')

print(b, 3\*b, end=' ,')

print(b, 3\*b)

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.6. Циклы. Миронова И.В. 2020

В следующем фрагменте кода

a = [1, 3, 5, 7, 9]

for b in a:

print(b)

a.pop()

цикл выполнится … раз.

3

После выполнения программы

lst = []

for s in 'ab ':

lst.append(s)

else:

lst.append('\*')

произойдет следующее – …

переменная lst будет иметь значение ['a', 'b', ' ', '\*']

В следующем фрагменте кода

a = [0, 1, 2]

for b in a:

a.insert(0, b + 2)

цикл выполнится следующее количество раз – …

бесконечное количество (программа зациклится)

После выполнения программы

lst = []

for i in range(-2, 3):

lst.append(i)

if i == 0:

break

else:

lst.append(5)

переменная lst будет иметь значение …

[-2, -1, 0]

После выполнения программы

i = 1

s = 0

while i <= 5:

i \*= 2

if i%2 != 1:

continue

else:

s += i

else:

s += 5

переменная s будет иметь значение …

5

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 1. Языки и технологии программирования. Миронова И.В. 2020

Программа, которая переводит текст программы, написанный на языке программирования, в эквивалентную программу на машинно-ориентированном языке, – это

компилятор

Программа, которая построчно анализирует, обрабатывает и выполняет исходный код программы, – это …

компоновщик

эмулятор

компилятор

интерпретатор

виртуальная машина

Первым объектно-ориентированным языком считается …

Prolog

Pascal

Simula

Java

C++

Первым функциональным языком был …

Lisp

Понятие «полиморфизм» используется в … программировании.

объектно-ориентированном

декларативном

логическом

императивном

функциональном

Для функционального программирования характерно использование понятий:

чистая функция

инструкция

объект

полиморфизм

итератор

Для императивного программирования характерно использование понятий:

подпрограмма

присваивание

объект

переменная

чистая функция

Объектно-ориентированное программирование поддерживают языки программирования:

C#

Python

Java

SQL

C

Объектно-ориентированное программирование НЕ поддерживают язык программирования:

Паскаль

Функциональное программирование НЕ поддерживается в языках программирования:

Pascal

Haskell

Java

Python

Visual Basic

Язык программирования Java поддерживает:

процедурное программирование

императивное программирование

объектно-ориентированное программирование

функциональное программирование

логическое программирование

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 5. Элементы функционального программирования. 5.1. Функция в качестве параметра. Миронова И.В. 2020

Имеется программа

def f1(a, b, f):

print(f(a, b))

x=1

y=2

Корректными вызовами функции f1 являются:

f1(3, 10, max)

f1(x, y, lambda: x+y)

f1(2, 5)

f1( a=2, b=3, f=sum)

f1([2, 3, 5], y, list.count)

Имеются функции

def f1(a, f):

print(f(a), f(a))

def f2(x):

return x+1

Корректными вызовами функции f1 являются:

f1(math.pi, math.cos())

f1('123', len)

f1({3, 5}, str)

f1(5, lambda x: str(x-1))

f1(11, f2(3))

Имеется функция

def f1(a=0, f=str):

print(f(a))

Корректными вызовами функции f1 являются:

f1(input())

f1(f=math.sin())

f1([3, 10, 2], max())

f1([2, 3, 5])

f1()

Имеются функции

def f1(f, a=''):

print(f(a))

def f2(x):

return str(x)

Корректными вызовами функции f1 являются:

f1(lambda x: x+1, 5)

f1(f2([]))

f1(lambda x: f2(x))

f1(f2(), [])

f1(f2, [])

Имеется программа

import math

import random

def f1(f):

print(f() + 1)

def f2(x=0):

return x+1

Корректными вызовами функции f1 являются:

f1(f2(2))

f1(f2)

f1(random.random)

f1(math.pi)

f1()

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.5. Функции range() и enumerate(). Миронова И.В. 2020

Следующий код

a=[[i+j for j in range(2)] for i in range(2)]

print (a)

выведет …

[[0, 1, 2], [1, 2, 3], [2, 3, 4]]

[[1]]

[[0, 1], [1, 2]]

[[0, 1], [2, 3]]

[[0], [1], [1], [2]]

После выполнения цикла

ls = [1, 5, 7]

for i, j in enumerate(ls, 3):

print(2\*i+j)

последним будет выведено значение

17

После выполнения цикла

ls = [8, 1]

for i, j in enumerate(ls, 2):

print(j, i, sep='', end='')

будет выведено значение

8213

Цикл

for i in range(2, 10, -2):

print(i)

выполнится … раз.

0

Оператор

print(sum(range(10, -4, -4)))

выведет значение

16

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 8. Структуры данных. Миронова И.В. 2020

Структура данных, в которой доступ к элементам организован по принципу FIFO, – это …

дек

список

стек

куча

очередь

Для двоичного дерева поиска выполняются следующие условия:

глубина всех листьев отличается не более чем на 1

у всех вершин правого поддерева произвольной вершины X значения больше либо равны значения в X

у всех вершин левого поддерева произвольной вершины X значения меньше значения в X

значение в любой вершине не меньше, чем значения её потомков

последний слой заполняется слева направо без «дырок»

Структура данных, состоящая из узлов, каждый из которых содержит данные и ссылку на следующий или первый узел, – это …

дек

кольцевой односвязный список

односвязный список

двусвязный список

кольцевой двусвязный список

Структура данных, состоящая из узлов, каждый из которых содержит данные и ссылку на следующий или предыдущий узел это

Связный список

Список [None, 3, 8, 4, 9, 16, 8, 5, 10, 18, 17, 32] представляет собой линейную запись двоичной кучи (элемент с индексом 0 не является членом кучи). Самый правый элемент уровня 2 в этой куче имеет значение …

5

В двоичном дереве поиска данные извлекаются в порядке убывания, когда обрабатывается …

левое поддерево, затем – текущая вершина, затем – правое поддерево

текущая вершина, затем – ее левое поддерево, затем – ее правое поддерево

левое поддерево, затем – правое поддерево, затем – текущая вершина

правое поддерево, затем – левое поддерево, затем – текущая вершина

правое поддерево, затем – текущая вершина, затем – левое поддерево

При обратном обходе дерева обрабатывается …

текущая вершина, затем – ее левое поддерево, затем – ее правое поддерево

левое поддерево, затем – текущая вершина, затем – правое поддерево

левое поддерево, затем – правое поддерево, затем – текущая вершина

правое поддерево, затем – текущая вершина, затем – левое поддерево

правое поддерево, затем – левое поддерево, затем – текущая вершина

Для хеш-таблицы справедливы следующие утверждения:

позволяет реализовать быстрый алгоритм обхода всех хранимых пар в порядке возрастания или убывания ключей

позволяет хранить пары (ключ, значение)

добавление, поиск и удаление элементов выполняется в среднем за время 

при реализации хеш-таблицы используется массив

добавление, поиск и удаление элементов выполняется в худшем случае за время 

Хеш-функция удовлетворяет следующим условиям:

возвращает конечное количество значений любого типа

для разных значений аргумента может возвращать одинаковый результат

возвращает целое число

сложность вычисления функции зависит от количества элементов в хеш-таблице

для заданного значения ключа всегда должна возвращать одно и то же значение

Преимуществами связных списков являются:

не требуется хранить какую-либо информацию кроме самих данных

легкость вычисления адреса элемента

одинаковое время доступа ко всем элементам

динамическое добавление и удаление элементов

добавление и удаление элементов выполняются за постоянное время

Преимуществами массивов являются:

легкость вычисления адреса элемента

одинаковое время доступа ко всем элементам

динамическое добавление и удаление элементов

добавление и удаление элементов выполняются за постоянное время

не требуется хранить какую-либо информацию кроме самих данных

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.11. Преобразования между списками и строками. Миронова И.В. 2020

Преобразование строки s = 'abcaada' к виду s = 'bcd', можно выполнить с помощью операторов:

s = ''.join(s.split('a'))

s = ''.join(for x in s if x!='a')

s = ''.join(list(s).remove('a'))

s = s.replace('a', '')

s = ''.join(list(s).remove('a'))

Преобразование строки s = '5 7 9' в список L, имеющий значение [5, 7, 9], могло быть выполнено с помощью оператора …

L = [int(i) for i in s]

L = [i for i in s.split( ' ' )]

L = int(s.split( ' ' ))

L = map(int, list(s.split( ' ' )))

L = list(map(int, s.split( ' ' )))

После выполнения операторов

s = ':'.join(map(str, range(10, 40, 10)))

print(s)

будет напечатано …

10:20:30

После выполнения операторов

s = "15 12 1 22 44 3 "

a = s.split()

print (a[1] + 3)

будет напечатано …

15

18

4

3

сообщение об ошибке

Преобразование строки s = '123' в список lst, имеющий значение ['1', '2, '3'], можно выполнить с помощью операторов:

lst = list(map(int, s))

lst = [x for x in s]

lst = list(map(str, s))

lst = s.split()

lst = list(s)

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.5. Термины. Миронова И.В. 2020

Инкапсуляция позволяет …

ограничивать доступ к переменным и методам класса

работать с различными реализациями классов через один интерфейс

использовать одни классы для создания других

создавать новые классы из существующих

разным объектам одного класса иметь разное поведение

Утиная типизация в Python позволяет …

при использовании обращать внимание только на интерфейс объекта, а не на его класс

создавать объект, являющийся одновременно представителем нескольких классов

делать классы-наследники несвязанными с классами-родителями

изменять тип объекта во время выполнения программы

использовать в разных классах одни и те же имена

Класс:

используется для создания объекта

всегда имеет конструктор

описывает атрибуты и методы объекта

хранит данные экземпляра объекта в памяти

в Python является объектом

Наследование в Python позволяет:

создать новый класс из существующего, переопределив в нем часть методов

при создании нового класса использовать существующие классы

создавать классы с одинаковым интерфейсом

создать новый класс из существующего, удалив из него часть методов или атрибутов

создать новый класс из существующего, добавив в него новые методы или атрибуты

Полиморфизм позволяет …

ограничивать доступ к некоторым переменным и методам

скрывать реализацию от пользователя класса

переименовывать методы в дочерних классах

одной и той же программе работать с объектами разных классов (работать с различными реалезациями классов через один интерфейс)

разным объектам одного класса иметь разные атрибуты

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.12. Кортежи. Миронова И.В. 2020

Имеется кортеж

t = ('1', '3', '4')

Сообщение об ошибке будет выведено при выполнении следующих операторов:

t1 = t.pop()

a = '.'.join(t)

t2 = sorted(t)

t = t + ('2')

t2 = t[::-1]

Имеются переменные

t = (1, 3, 4, 5, 6)

a = 1

c = 7

Правильными являются следующие операторы присваивания:

t = a, c, \*t

t1 = t, c

a, c, \*t = t

t2 =a, {t}, c

\*a, \*b, c = t

Имеются переменные

t = (3, 2, 1)

x = 10

Сообщение об ошибке будет выведено при выполнении следующих операторов:

del t[0]

a = t[2]

t1 = t + (x,)

t1 = 2 \* t

t[0] = 5

Для кортежей определены следующие методы:

find()

count()

copy()

index()

replace()

Правильными операторами, в которых переменной T в качестве значения присваивается кортеж, являются:

T = (10)

T = ({1:1}, )

T = ( )

T = (x)

T = [],

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 5. Элементы функционального программирования. 5.3. Функции map(), filter(), reduce(). Миронова И.В. 2020

После выполнения операторов

ls = [(1, 0), (2, 3), (3, -1), (4, 0), (3, 10)]

f = lambda x: bool(x[1])

a = sum(map(f, filter(f, ls)))

значение a равно

3

После выполнения операторов

from functools import reduce

lst = ["++\*", "\*##", "%$#"]

print(reduce(lambda a, b: a + lst[b][b] , range(3), ""))

будет напечатано

+##

После выполнения операторов

lst = [5, 1, -1, 2]

b = sum(map(lambda x: 2 \* x, filter(lambda x: 0 < x < 3, lst)))

значение b равно

6

После выполнения оператора

a = sum(map(lambda x: x if x>0 else -x, range(-2, 2)))

значение a равно

4

После выполнения операторов

from functools import reduce

lst = [-3, 1, -5, 0]

b = reduce(lambda a, b: a + b if (b > 0) else a, lst )

значение b равно

-2

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.3. Логические выражения. Миронова И.В. 2020

Выражение

x>0 and x<5 or x<-2

истинно для значений x:

7

1

-1

0

-5

Выражение, имеющее значения True, когда переменная X не равна 1, 4 и любому значению меньше 0, – это …

not(X == 1 and X == 4 and X < 0)

X == 1 or X == 4 or X < 0

X != 1 and X != 4 and X < 0

X != 1 and X != 4 and X >= 0

X == 1 and X == 4 and X < 0

Выражение, имеющее значения True, когда переменная X не равна 3, 5 и любому значению больше 10, – это

X != 3 and x != 5 and x<=10

Выражение

bool(x and y)

истинно для значений x, y:

x = 1, y = ''

x = 0, y = ''

x = [0], y = {}

x = 1, y = '1'

x = [0], y = '1'

Когда значение переменной X не равно 2 и 4, выражение … имеет значение True.

X != 2 and X != 4

not(X == 2 and X == 4)

X == 2 and X == 4

X != 2 or X != 4

X == 2 or X == 4

Любой объект в логическом контексте может интерпретироваться как True или False. Значение False имеют следующие выражения:

bool('0')

bool(0.0)

bool({0:0})

bool(0)

bool([])

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.16. Множества. Миронова И.В. 2020

После выполнения операторов

s1 = set([1, 3, 7, 1]) | set(range(4)) & {2, 3, 10}

print(len(s1))

на экран будет выведено

4

После выполнения операторов

s1 = set([3, 7, 3, 1]) ^ set(range(3))

s1.pop()

print(len(s1))

на экран будет выведено

3

После выполнения операторов

s1 = set([3, 7, 3, 1]) - set(range(3))

s1.pop()

print(len(s1))

на экран будет выведено

1

После выполнения операторов

s = {0}

s.add(1)

s.remove(1)

s.discard(1)

print(s.pop())

на экран будет выведено …

сообщение об ошибке

set()

None

0

{0}

После выполнения операторов

s1 = set([2, 3, 4, 2]) - set(range(4)) & {2, 7}

print(len(s1))

на экран будет выведено …

0

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 5. Элементы функционального программирования. 5.4. Декораторы, итераторы, генераторы. Миронова И.В. 2020

Функция-генератор:

обязательно содержит инструкцию, которая возбуждает исключение StopIteration

не сохраняет значения переменных между вызовами

не может содержать инструкцию return

обязательно содержит хотя бы одну инструкцию yield

при повторном обращении продолжает своё исполнение с места, на котором была приостановлена

Декоратор – это…

объект, который позволяет поочередно перебирать элементы

функция, которая позволяет расширить функциональность другой функции без непосредственного изменения её кода

функция, содержащая выражение yield

функция, в которой есть вложенная функция

класс, который используется для создания итерируемого объекта

После выполнения операторов

ls1 = [0, 2, 0, 1]

ls2 = [7, -2, 0, -1]

b = any(sum(a) for a in zip(ls1, ls2))

значение b равно …

случайное число из 0 и 7

False

True

7

-0

Для выражений-генераторов справедливы следующие утверждения:

после прохождения по выражению оно остается пустым

может быть бесконечным

можно распечатать элементы функцией print()

нельзя писать без скобок

можно получить длину функцией len()

После выполнения операторов

ls1 = [-5, 7, 10, 0]

ls2 = [2, 0, -2, 1]

b = all(max(a) for a in zip(ls1, ls2)

значение b равно …

[2, 7, 10, 1]

2

10

False

True

Итераторами являются функции:

open()

zip()

reduce()

max()

enumerate()

Справедливы следующие утверждения:

метод \_\_iter\_\_() должен возвращать итератор

параметром функции next() является итерируемый объект

параметром функции next() является итератор

параметром функции iter() является итератор

параметром функции iter() является итерируемый объект

После выполнения операторов

ls1 = [2, 3]

ls2 = [1, 4]

b = sum((list(a) for a in zip(ls1, ls2)), [])

значение b равно …

[3, 7]

[[2, 3], [1, 4], []]

10

[5, 5]

[2, 1, 3, 4]

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.13. Создание словарей. Миронова И.В. 2020

После выполнения операторов

d1 = {'a': 2, 'b': 3}

d2 = dict(d1)

d1['a'] = 10

d2['b'] = 20

print(d1['b'] + d2['a'])

на экран будет выведено …

Ответ 5, но тест не засчитывает

Правильными операторами, в которых переменной d в качестве значения присваивается словарь, являются:

d = { }

d = {(1, 'a'): 'a', (2, 'b'): 'ab'}

d = {'a': 'a', 2: 'b'}

После выполнения операторов

d1 = dict(zip('ijk', range(2)))

print(d1.get('k'))

на экран будет выведено …

None

После выполнения операторов

d1 = {'a': [10, 3], 'b': [20, 5]}

d2 = dict(d1)

d1['a'] = [1, 4]

d2['b'][0] = 7

print(d1['a'] [0] + d2['a'][0] + d1['b'][0])

на экран будет выведено…

18

После выполнения операторов

d1 = d2 = {'a': 1, 'b': 5}

d1['a'] = 2

d2['b'] = 3

print(d1['b'] + d2['a'])

на экран будет выведено …

5

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.2. Список в качестве параметра функции. Миронова И.В. 2020

После выполнения операторов

def data(d):

d.insert(0, 1)

return d

a=[5]

b=data(a)

c=data(a)

print(a,b,c)

на экран будет выведено …

[1, 1, 5] [1, 5] [1, 1, 5]

сообщение об ошибке

[5] [1, 5] [1, 5]

[1, 5] [1, 5] [1, 1, 5

[1, 1, 5] [1, 1, 5] [1, 1, 5]

После выполнения операторов

def data(x):

x += [0]

return x

ls1 = [1, 2]

ls2 = data(ls1)

print(ls1 == ls2, ls1 is ls2)

будет напечатано …

True True

False False

сообщение об ошибке

False True

True False

После выполнения операторов

def data(d):

del d[-1]

return d

a = [1, 2, 3]

b = data(a)

c = data(a)

print(b, c)

на экран будет выведено …

[1] []

[1, 2] [1]

сообщение об ошибке

[1, 2] [1, 2]

[1] [1]

После выполнения операторов

def data(x):

x = x + [0]

return x

ls1 = [1, 2]

ls2 = data(ls1)

print(ls1 == ls2, ls1 is ls2)

будет напечатано …

сообщение об ошибке

True True

False False

False True

True False

После выполнения операторов

def data(d=[1]):

d.insert(0, 1)

return d

a=data()

b=data()

c=data()

print(b,c)

на экран будет выведено …

сообщение об ошибке

[1, 1] [1, 1]

[1, 1, 1] [1, 1, 1]

[1, 0] [1, 0]

[1, 1, 1, 1] [1, 1, 1, 1]

После выполнения операторов

def data(d=[]):

d.insert(0, 1)

return d

a=data()

b=data()

c=data([5])

print(b,c)

на экран будет выведено …

[1,1] [1,5]

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.14. Изменение словарей. Миронова И.В. 2020

Имеется словарь d = {'m': {3: 'март', 6: 'июнь'}}. Удалить элемент 6: 'июнь', чтобы d стал иметь вид {'m': {3: 'март'}}, можно следующими способами:

del d['m'][6]

d['m'].pop(6)

d['m'].del(6)

d['m'].pop('июнь')

d['m'].remove(6)

Имеется словарь d = {1:{'a': 0}}. Изменить значение 0 на 10 так, чтобы словарь стал иметь вид {1:{'a': 10}}, можно с помощью операторов:

d[1]['a'] = 10

d.update(1).update(a=10)

d.update({'a':10})

d[1].update(a=10)

d[1].update({'a':10})

Имеется словарь x = {"товары": []}. Добавить в список элемента "товары" строку "мяч" можно следующими способами:

x["товары"].insert(0,"мяч")

x["товары"][] = "мяч"

x["товары"] += ["мяч"]

x["товары"] = "мяч"

x["товары"].append("мяч")

После выполнения операторов

d = {'a': 5}

x = d.pop('a') + 2

y = d.get('a', 1) + 3

print(x + y)

на экран будет выведено

11

Имеется словарь d = {1: 21, 2: 22}. Добавить в словарь еще один элемент 3:24 можно с помощью операторов:

d.update(3, 24)

d.update([[3, 24]])

d.update(3=4)

d.update({3:24})

d.update([3, 24])

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.10. Методы списков. Миронова И.В. 2020

Имеется список arr = [1, 2, 3]. Чтобы изменить его на arr = [1, 3], можно использовать операторы:

arr.pop(1)

arr[1] = None

arr.remove(2)

del arr[2]

arr.replace(2, None)

Имеется список x = [3, 2, 1]. Оператор

print(x == y, x is y)

выведет на экран

True False

если y был создан с помощью одного из операторов:

y = x.copy()

y = copy(x)

y = x[:]

y = x

y = list(x)

После выполнения операторов

ar1 = [1, 2]

ar2 = ar1[:]

ar2.append(3)

ar3 = ar2

del ar3[1]

списки ar1, ar2, ar3 будут равны …

ar1 = [1, 2], ar2 = [1, 2, 3], ar3 = [2, 3]

ar1 = [1, 2], ar2 = [2, 3], ar3 = [2, 3]

ar1 = [1, 2], ar2 = [1, 2, 3], ar3 = [1, 3]

ar1 = [2, 3], ar2 = [2, 3], ar3 = [2, 3]

ar1 = [1, 2], ar2 = [1, 3], ar3 = [1, 3]

Имеется список arr = [1, 2]. Чтобы изменить его на arr = [1, 1, 2], можно использовать операторы:

arr.insert(0, 1)

arr = [1] + arr

arr.extend(1)

arr.insert(1, 1)

arr.append(1)

После выполнения операторов

ls = [2, 3, 4, 2, 3]

ls.remove(2)

значение ls равно …

[4, 2, 3]

[3, 4, 2, 3]

[2, 3, 2, 3]

[2, 3, 4, 3]

[3, 4, 3]

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 6. Объектно-ориентированное программирование. 6.2. Свойства. Миронова И.В. 2020

После выполнения операторов

class Class1:

def \_\_init\_\_(self, a=1):

self.\_\_v = a

def rv(self):

return self.\_\_v+2

v = property(rv)

b=Class1(5)

b.v = 0

print(b.v)

на экран будет выведено …

1

0

сообщение об ошибке

5

7

После выполнения операторов

class Class1:

def \_\_init\_\_(self, a=1):

self.\_\_v = a

def rv(self):

return self.\_\_v+2

v = property(rv)

b=Class1(5)

print(b.v)

на экран будет выведено …

сообщение об ошибке

5

None

7

1

После выполнения операторов

class Class1:

def \_\_init\_\_(self, t=0):

self.\_v = t

@v.getter

def v(self):

return self.\_v+2

b=Class1(5)

print(b.v)

на экран будет выведено …

сообщение об ошибке

5

None

2

7

После выполнения операторов

class Class1:

def \_\_init\_\_(self, a=1):

self.\_\_v = a

v = property(rv)

def rv(self):

return self.\_\_v+2

b=Class1(5)

print(b.v)

на экран будет выведено …

None

7

5

сообщение об ошибке

1

После выполнения операторов

class Class1:

def \_\_init\_\_(self, t=0):

self.\_v = t

@property

def v(self):

return self.\_v+2

b=Class1(5)

print(b.v)

на экран будет выведено …

сообщение об ошибке

5

2

None

7

### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 2. Типы данных и операции. 2.7. Операции для строк. Миронова И.В. 2020

После выполнения операторов

a = 'да'

b = 'задача'

print(a is b, a in b)

на экран будет выведено …

False False

True False

сообщение об ошибке

True True

False True

После выполнения операторов

s1 = 'aabbbcc'

s2 = s3 = 'bbb'

s4 = s1[2:-2]

значение True будут иметь выражения:

s2 == s4

s3 in s4

s3 is s4

s2 is s3

s2 in s3

После выполнения операторов

s='12345'

s1 = s[-1::] + s[-1::-1]

print(s1)

будет напечатано …

554321

После выполнения операторов

ls = [4,3,2,1]

a = 3 \* str(ls)[1] + str(ls)[-2]

print(a)

будет напечатано

4441

После выполнения операторов

a = "Технология"

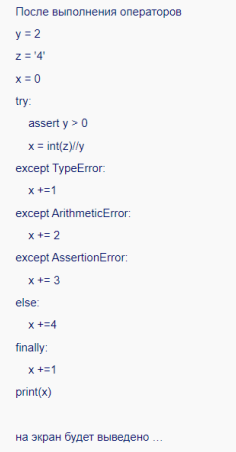
b = a[7:4:-1]

print(b)

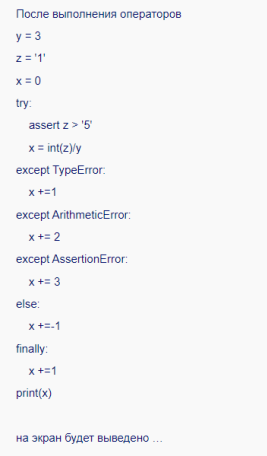
будет напечатано

гол

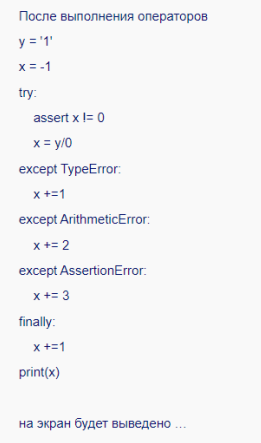
### Самоподготовка. Алгоритмы и структуры данных в языке Python. Тема 3. Функции, модули, исключения. 3.5. Исключения. Миронова И.В. 2020



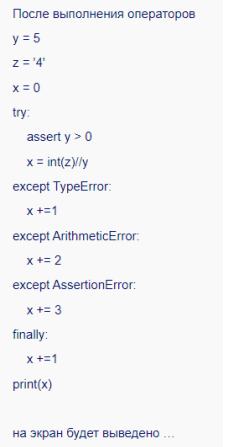
7



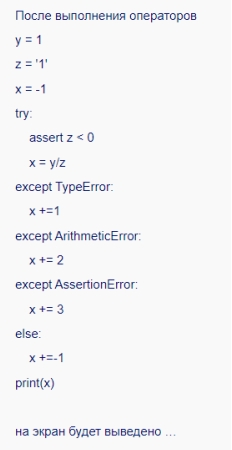
4



1



1



0

### ПМ экзамен

Вопрос 1

Язык программирования C++ поддерживает:

1. процедурное программирование

2. функциональное программирование

3. декларативное программирование

4. объектно-ориентированное программирование

5. логическое программирование

Ответ: 14

Вопрос 2

Для функционального программирования характерно использование понятий:

1. чистая функция

2. объект

3. инструкция

4. рекурсия

5. инкапсуляция

Ответ: 14

Вопрос 3

После выполнения операторов

a = 3

b = ( a\*\*2 % (a + a) + 10) // a

Значение переменной b равно …

Ответ: 4

Вопрос 4

Значение переменной a равно 5. Напишите оператор, который вычисляет и выводит значение квадрата этого числа и квадратного корня из этого числа в одной строке через запятую (без пробела) с двумя значащими цифрами после запятой …

Ответ: print(str(a\*\*2) + ',' + str(round(a\*\*0.5, 2)))

Вопрос 5

Переменная x может иметь значения от 0 до 5 и от 15 до 25. Напишите логическое выражение для проверки этого условия …

Ответ: if 0 <= x <= 5 or 15 <= x <= 25:

Вопрос 6

В результате выполнения следующего фрагмента

a = с = 1

b = d = 1.0

print (a is b and c is a and a == d)

на экран будет выведено …

Ответ: False

Вопрос 7

Цикл

for i in range(1, 10, -2):

print(i)

выполнится … раз.

Ответ: 0

Вопрос 8

В следующем фрагменте кода

a = [1, 2, 3, 4, 5]

for b in a:

print(b)

a.pop()

цикл выполнится … раза.

Ответ: 3

Вопрос 9

Имеется строка s = '123456789'. Получить из нее строку '987654321' можно с помощью среза …

Ответ: s[::-1]

Вопрос 10

Имеется строка s = 'ABCDEF'. При работе с этой строкой правильными являются операторы …

1. s1 = s.append('G')

2. s2 = s.replace('A', 'a', 2)

3. s3 = s.remove('C')

4. s4 = s.lower()

5. s5 = s.count('x')

Ответ: 245

Вопрос 11

После выполнения операторов

s = '5++++5'

print(s.replace('++', '\*'))

Ответ: 5\*\*5

Вопрос 12

После выполнения операторов

ls1 = ls2 = [1, 2]

ls1.append(3)

print(ls1 is ls2)

будет напечатано …

Ответ: True

Вопрос 13

После выполнения операторов

A = [1, 3, 1, 1, 7]

A.remove(1)

значение A равно …

Ответ: [3, 1, 1, 7]

Вопрос 14

Имеется список x = [3, 2, 1]. Для создания копии этого списка можно использовать операторы:

1. y = x

2. y = list(x)

3. y = copy(x)

4. y = x[:]

5. y = x.copy()

Ответ: 245

Вопрос 15

Преобразование строки s = '345' в список lst, имеющий значение ['3', '4', '5'], можно выполнить с помощью операторов:

1. lst = list(s)

2. lst = s.split()

3. lst = s.split('')

4. lst = [x for x in s]

5. lst = list(map(int, s))

Ответ: 14

Вопрос 16

Правильными операторами, в которых переменной T в качестве значения присваивается кортеж, являются:

1. T = 1,

2. T = (1)

3. T = ( )

4. T = ([1], )

5. T = (, 1)

Ответ: 134

Вопрос 17

После выполнения операторов

d1 = d2 = {'a': 2, 'b': 3}

d1['a'] = 11

d2['b'] = 9

print(d1['b'] + d2['a'])

на экран будет выведено…

Ответ: 20

Вопрос 18

Имеется словарь d = {1: 2, 5: 7}. Добавить в словарь еще одно значение 3:4 можно с помощью операторов:

1. d.update([3, 4])

2. d.update([[3, 4]])

3. d.update(3,4)

4. d.update({3:4})

5. d.update(3=4)

Ответ: 24

Вопрос 19

После выполнения операторов

d = {'a': 7}

x = d.pop('a') + 1

y = d.get('a', 1) + 1

print(x + y)

на экран будет выведено …

Ответ: 10

Вопрос 20

Имеется словарь D. Создайте список всех ключей словаря

Ответ: keys = list(D.keys())

Вопрос 21

Имеется список чисел L. Создайте словарь D, в котором элементы списка L будут значениями. Ключами будут целые числа 1, 2, 3 и т.д.

Ответ:

d = {}

for i in range (len(L)):

d.update({i + 1:L[i]})

Вопрос 22

После выполнения операторов

s1 = set([1, 3, 5, 3]) - set(range(4)) & {3, 10}

print(len(s1))

на экран будет выведено …

Ответ: 0

Вопрос 23

Имеется функция

def my\_f(a, b=1, c='2'):

print(a,b,c)

Корректными вызовами этой функции являются:

1. my\_f()

2. my\_f(10)

3. my\_f(b=2, a=3)

4. my\_f(b=4, c='1')

5. my\_f(1, 3, 5)

Ответ: 235

Имеется функция

def my\_f(a, b=1, c='2'):

print(a,b,c)

p = [10,20]

my\_f(\*p,’2’)

my\_f(\*p)

my\_f(2,\*p)

Вопрос 24

После выполнения операторов

def data(d):

d.append(1)

return d

a=[0]

b=data(a)

c=data(a)

print(a,b,c)

на экран будет выведено …

1. [0] [0, 1] [0, 1]

2. [0, 1] [0, 1] [0, 1, 1]

3. [0, 1, 1] [0, 1] [0, 1, 1]

4. [0, 1, 1] [0, 1, 1] [0, 1, 1]

5. сообщение об ошибке

Ответ: 4

Вопрос 25

Имеется функция

def prog(\*a, \*\*b):

print(a,b)

Корректными вызовами функции являются:

1. prog(a=1, b=2)

2. prog(11, 12, 13)

3. prog(d=1)

4. prog()

5. prog(a=1, 4, 2)

Ответ: 1234

Вопрос 26

Оператор

a = math.sqrt(2) или a = math.radians(45)

выполнится корректно, если предварительно выполнить одну из инструкций:

1. import math

2. import math as \*

3. from math import sqrt

4. from math import \*

5. import sqrt from math

Ответ: 1

Вопрос 27

После выполнения операторов

y = '1'

x = -1

try:

assert x > 0

х = y/0

except TypeError:

x +=1

except ArithmeticError:

х += 2

except AssertionError:

x += 3

finally:

x +=1

print(x)

на экран будет выведено …

Ответ: 3

Вопрос 28

Для построения графиков и диаграмм в Python можно использовать библиотеку …

1. SymPy

2. NumPy

3. Matplotlib

4. SciPy

5. Pandas

Ответ: 3

Вопрос 29

В модуле math находятся функции:

1. sqrt()

2. random()

3. factorial()

4. pow()

5. choice()

Ответ: 134

Вопрос 30

Переменная f содержит ссылку на открытый для чтения текстовый файл. Прочитайте из этого файла 1 символ.

Ответ: f.read(1)

Вопрос 31

Имеются функции

def f1(a, b, f):

print(f(a)+f(b))

def f2(x):

return x-1

Корректными вызовами функции f1 являются:

1. f1(10, 5, f2)

2. f1('a', 'b', len)

3. f1(0, 3, math.sin())

4. f1(1, 5, str())

5. f1(10, 5, lambda x: x\*\*2)

Ответ: 125

Вопрос 32

Напишите анонимную функцию с одним аргументом x, которая вычисляет значение (x+1)/2. Сохраните ссылку на функцию в переменной f1

Ответ: f1 =lambda x : ((x+1)/2)

Вопрос 33

Имеется список:

arr=[('Иванов', 25, 4), ('Петров', 5, 10), ('Сидоров', 11, 2)]

Отсортируйте этот список по возрастанию последнего значения в кортеже.

Ответ: arr.sort(key=lambda x: x[2])

Вопрос 34

После выполнения операторов

li1 = [1, 0, 2]

li2 = [10, 20, 30]

b = max(map(lambda x, y: x\*y, li1, li2))

значение b равно …

Ответ: 60

Вопрос 35

После выполнения операторов

from functools import reduce

ls = [1, 0, 5, 2]

b = reduce(lambda x, y: x\*y, filter(None, ls), 3)

значение b равно …

Ответ: 30

Вопрос 36

Функция, которая принимает функцию в качестве единственного аргумента и возвращает новую функцию, включающую эту функцию с дополнительными функциональными возможностями – это …

1. декоратор

2. итератор

3. генератор

4. замыкание

5. вложенная функция

Ответ: 1

Вопрос 37

Функция-генератор:

1. не может иметь параметров

2. обязательно содержит инструкцию yield

3. может содержать более одной инструкции yield

4. сохраняет значения переменных между вызовами

5. выполняется только при явном использовании функции next()

Ответ: 2345

Выражение-генератор:

Вопрос 38

После выполнения операторов

class Class1:

i = 1

def \_\_init\_\_(self):

i = 2

el = Class1()

i = el.i

print(i)

на экран будет выведено …

1. 0

2. 1

3. 2

4. None

5. сообщение об ошибке

Ответ: 2

Вопрос 39

Имеются классы

class Base:

def \_\_init\_\_(self):

self.\_\_x = 1

class Child(Base):

def \_\_init\_\_(self):

super().\_\_init\_\_()

…

Чтобы в классе Child получить доступ к атрибуту \_\_x, нужно использовать выражение…

1. \_\_x

2. self.\_\_x

3. Base.\_\_x

4. self.\_Base\_\_x

5. атрибут в Child недоступен

Ответ: 5

Вопрос 40

После выполнения операторов

class Class1:

def \_\_init\_\_(self, n=0):

self.\_\_x = n

def gx(self):

return self.\_\_x+2

x = property(gx)

c = Class1(2)

c.x = 5

print(c.x)

на экран будет выведено …

1. 0

2. 2

3. 5

4. 7

5. сообщение об ошибке

Ответ: 5

Вопрос 41

Имеется класс и объект этого класса

class Class1:

def dbl1(self, param):

return param \* 2

def dbl2 (param):

return param \* 2

@staticmethod

def dbl3(param):

return param \* 2

def \_dbl4(self, param):

return param \* 2

def \_\_dbl5(self, param):

return param \* 2

c = Class1()

Инструкция c.metod(2) корректно выполнится для методов:

1. dbl1

2. dbl2

3. dbl3

4. \_dbl4

5. \_\_dbl5

Ответ: 13

Вопрос 42

Имеется класс

class Class1:

def dbl1(self, param):

return param \* 2

def pr(self, x):

a = …

print(a)

Чтобы вычислить значение a с помощью метода dbl1, нужно использовать оператор …

1. a = dbl1(x)

2. a = dbl1(self, x)

3. a = self.dbl1(x)

4. a = Class1.dbl1(x)

5. a = \_Class1\_\_dbl1(x)

Ответ: 3

Вопрос 43

После выполнения операторов

class Base:

def \_\_init\_\_(self):

self.x = 0

self.y = 7

def pr(self):

print(self.x, self.y)

class Child(Base):

def \_\_init\_\_(self):

self.x = 3

el1 = Child()

el1.pr()

на экран будет выведено …

1. 0 7

2. 3 7

3. 0 None

4. 3 None

5. сообщение об ошибке

Ответ: 5

Вопрос 44

Имеется базовый класс Base и производный класс Child(Base). Чтобы вызвать в конструкторе класса Child конструктор базового класса, можно воспользоваться инструкциями:

1. Base.\_\_init\_\_()

2. Base.\_\_init\_\_(self)

3. super().\_\_init\_\_()

4. super().\_\_init\_\_(self)

5. super(self).\_\_init\_\_()

Ответ: 23

Вопрос 45

При наследовании класса нельзя …

1. добавить новый атрибут

2. реализовать новый конструктор

3. добавить новый метод

4. удалить атрибут базового класса

5. использовать методы базового класса

Ответ: 4

Вопрос 46

Имеется список [15, 17, 31, 10, 7]. После выполнения одного прохода алгоритма сортировки вставками список будет иметь вид …

1. [15, 17, 10, 7, 31]

2. [15, 17, 7, 10, 31]

3. [10, 15, 17, 31, 7]

4. [7, 17, 10, 31, 15]

5. [10, 7, 15, 31, 17]

Ответ: 3

Вопрос 47

Для алгоритма пузырьковой сортировки справедливы следующие утверждения:

1. меняет взаимное расположение элементов с одинаковыми значениями

2. при работе требуется дополнительная память

3. имеет среднюю и максимальную временные сложности O(n^2)

4. имеет временную сложность O(n log(n))

5. при обработке уже упорядоченных или частично упорядоченных данных алгоритм работает быстрее

Ответ: 135

Вопрос 48

Структура данных, в которой доступ к элементам организован по принципу LIFO, – это…

1. список

2. стек

3. очередь

4. дек

5. куча

Ответ: 2

Вопрос 49

В двоичном дереве поиска данные извлекаются в порядке возрастания, когда обрабатывается …

1. текущая вершина, затем – ее левое поддерево, затем – ее правое поддерево

2. левое поддерево, затем – текущая вершина, затем – правое поддерево

3. левое поддерево, затем – правое поддерево, затем – текущая вершина

4. правое поддерево, затем – левое поддерево, затем – текущая вершина

5. правое поддерево, затем – текущая вершина, затем – левое поддерево

Ответ: 2

Вопрос 50

Хеш-функция должна удовлетворять условиям:

1. быстрое вычисление

2. для разных входных данных выдавать разные значения

3. минимальное количество коллизий

4. параметром функции должно быть число

5. реализует однозначное соответствие между входными и выходными данными

Ответ: 13

Вопрос 51

Имеется список arr = [1, 2, 3] .Чтобы изменить его на arr = [1, 2, 3, 4], можно использовать операторы:

1. arr = arr + [4] ☑
2. arr.extend(4) ❌
3. arr.insert(3, 4) ☑
4. arr.append(4) ☑
5. arr.add(4) ❌

Ответ: 134

Вопрос 52

t = (1, 3, 4, 5, 6)

a = 1

c = 7

1. t2 = [a], {c}, a ☑
2. a, \*b, c = t ☑
3. t1 = a, c ☑
4. a, c, \_ = t ❌
5. a, b = b, a ❌

Ответ: 123

Вопрос 53

После выполнения операторов

1. d1 = {"a":[2, 3], "b":[4, 5]}
2. d2 = dict(d1)
3. d1["a"] = [10, 20]
4. d2["b"][0] = 1
5. print(d1["a"][0] + d2["a"][0] + d1["b"][0])

Ответ: 13

Вопрос 54

Имеется словарь d = {1:{"b":2}}. Изменить значение 2 на 22 так, чтобы словарь стал иметь вид {1:{"b":22}}, можно с помощью операторов:

1. d[1].update(b=22) ☑
2. d[1]["b"] = 22 ☑
3. d.update(1).update(b=22) ❌
4. d.update({"b":22}) ❌
5. d[1].update({"b":22}) ☑

Ответ: 125

Вопрос 55

После выполнения операторов

d = {"a": 4}

x = d.pop("a") +1

y = d.get("a", 1) + 1

print(x+y)

Ответ: 7

Вопрос 56

После выполнения операторов

d = {5:1, 3:15, 2:7}

d1 = d.sort()

print(d1)

Ответ: сообщение об ошибке

Вопрос 57

После выполнения операторов

d = {5:1, 3:15, 2:7}

d1 = sorted(d)

print(d1)

Ответ: [2, 3, 5]

Вопрос 58

После выполнения операторов

s1 = set([1, 3, 5, 3]) - set(range(4))

s1.pop()

print(len(s1))

Ответ: 0

Вопрос 59

После выполнения операторов

def m\_sum(x=1, y=1, z =1):

return x + 2\*y +3\*z

print(m\_sum(\*"ab"))

Ответ: сообщение об ошибке

Вопрос 60

Имеются функции

def f1(a, b, f):

print(f(a, b))

def f2(x, y):

return x + y

Корректными вызовами функции f1 являются:

1. f1(1, 2, f2) ☑
2. f1("abc", "c", str.find) ☑
3. f1(p1, p2, max) ❌
4. f1(p1, p2, lambda : p1-p2) ❌
5. f1("a", "b", f2()) ❌

Ответ: 12

Вопрос 61

Имеются функции

def f1(f, a):

print(f(a))

def f2(x):

return x + 1

Корректными вызовами функции f1 являются:

1. f1(lambda x: f2(x), 5) ☑
2. f1(f2(5), 5) ❌
3. f1(f2(), 5) ❌
4. f1(lambda x: x+1, 5) ☑
5. f1(f2, 5) ☑

Ответ: 145

Вопрос 62

После выполнения операторов значение res равно ...

from functools import reduce

v = [1, 2, 3]

u = [1, 3, 4]

res = reduce(lambda x, y: x + y, map(lambda x, y: x\*y, v, u))

Ответ: 19

Вопрос 63

После выполнения операторов

class Class1:

i = 0

def \_\_init\_\_(self):

self.j = 1

el1 = Class1()

el2 = Class1()

el1.i = 2

Class1.i = 4

print(el1.i, el2.i, Class1.i)

Ответ: 2 4 4

Вопрос 64

После выполнения операторов

class Class1:

def \_\_init\_\_(self, n):

self.\_x = n

@property

def x(self):

return self.\_x + 2

c = Class1(2)

c.x = 5

print(c.x)

Ответ: сообщение об ошибке

Вопрос 65

После выполения операторов

class Base:

def \_\_init\_\_(self):

self.x = 1

self.y = 10

def pr(self):

print(self.x, self.y)

class Child(Base):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.x = 2

el1 = Child()

el1.pr()

Ответ: 2 10

Вопрос 66

После выполнения операторов

class Base:

def \_\_init\_\_(self):

self.\_\_x = 5

self.y = 7

def pr(self):

print(self.\_\_x, self.y)

class Child(Base):

def \_\_init\_\_(self):

super().\_\_init\_\_()

self.\_\_x = 10

el1 = Child()

el1.pr()

Ответ: 5 7

Вопрос 67

Базовые операции при работе со стеком

1. доступ по индексу к нужному значению в стеке
2. добавление элемента на вершину стека
3. удаление элемента с указанным индексом
4. извлечение элемента, находящегося на вершине стека
5. изменение элемента, находящегося на вершине стека

Вопрос 68

Инкапсуляция позволяет ...

1. создавать новые классы на базе существующих
2. создавать классы с одинаковым интерфейсом
3. скрывать внутреннюю реализацию объекта от пользователя
4. работать с различными реализациями классов через один интерфейс
5. использовать одни классы для создания других

Вопрос 71

После выполнения операторов

class Class1:

def \_\_init\_\_(self):

self.\_\_x = 1

def pr(self):

print(self.\_\_x)

a = Class1()

a.\_\_x = 2

a.pr()

на экран будет выведено …

Ответ: 1

Вопрос 73

После выполнения операторов

li1 = [-3, 0, -4, -1]

li2 = [3, 0, 0, 1]

b = any(sum(a) for a in zip(li1, li2))

Значение b равно …

1. 0
2. False
3. Случайное число из 0 и -4
4. True
5. -4

Ответ: 4

Вопрос 75

После выполнения операторов

from functools import reduce

s = "1###55#7"

v = reduce(lambda x, y: x+len(y), filter(None, s.split("#")), 1)

значение v равно …

Ответ: 5

Вопрос 76

В модуле os находятся функции:

1. remove()
2. mkdir()
3. choice()
4. copy()
5. rename()

Ответ: 125

Вопрос 77

После выполнения операторов

txt = 'abc\ndef'

with open("my2.txt", "wt") as f:

print(txt, end='', file=f)

with open("my2.txt", "wt") as f:

print(txt, file=f)

with open("my2.txt", "r") as f:

print(len(f.readlines()))

на экран будет выведено …

Ответ: 2

Вопрос 79

После выполнения операторов

def prog(a, \*\*b):

print(a,b)

d = {'a':3, 'b':2}

prog(1, \*\*d)

на экран будет выведено …

1. 1 {'a':3, 'b':2}
2. 3 {'a':3, 'b':2}
3. сообщение об ошибке
4. 3 {'b':2}
5. 1 {'b':2}

Ответ: 3

Вопрос 80

После выполнения операторов

def f(x):

x[0] = 0

return x

ls1 = [1, 2]

ls2 = f(ls1)

print(ls1 == ls2, ls1 is ls2)

будет напечатано …

1. False False
2. True False
3. сообщение об ошибке
4. False True

Ответ: 3

Вопрос 81

Имеется функция

def my\_f(a, b=1, c='2'):

print(a,b,c)

Корректными вызовами этой функции являются:

1. my\_f(‘1’)
2. my\_f(, 3, 5)
3. my\_f(2, b=4, ‘11’)
4. my\_f(a=3)
5. my\_f(c=2, a=3)

Ответ: 145

Вопрос 82

После выполнения операторов

d = dict([[5, 10], [6, 10]])

s = 0

for v in d.items():

s += v

print(s)

На экран будет выведено ...

TypeError: unsupported operand type(s) for +=: 'int' and 'tuple'

Ответ: ОШИБКА

Вопрос 83

После выполнения операторов

d = dict([[5, 10], [6, 10]])

s = 0

for v in d.items():

s += v

print(s)

На экран будет выведено ...

TypeError: unsupported operand type(s) for +=: 'int' and 'tuple'

Ответ: ОШИБКА

Вопрос 84

После выполнения операторов

d = {1:10, 3:30}

s = 0

for v in range(3):

s+= d.pop(v,1)

print(s)

Ответ: 12

Вопрос 85

После выполнения операторов

d1 = {'a': 2, 'b': 3}

d2 = dict(d1)

d1['a'] = 10

d2['b'] = 20

print(d1['b'] + d2['a'])

Ответ: 5

Вопрос 85

После выполнения операторов

d1 = {'a': 2, 'b': 3}

d2 = dict(d1)

d1['a'] = 10

d2['b'] = 20

print(d1['b'] + d2['a'])

Ответ: 5

Вопрос 86

Правильными операторами, в которых переменной X в качестве значения присваивается кортеж, являются:

X = (1)

X = (, 1)

X = (1,"\*",2.5)

X = ([1], )

X = tuple()

Вопрос 87

Преобразование строки s = '1,2,3' в список L, имеющий значение [1,2,3], могло быть выполнено с помощью оператора ...

L = [i for i in s.split(',')]

L = [int(i) for i in s if i != ',']

L = list(map(int, s.split('')))

L = [i for i in int(s.split(','))]

L = int(s.split(','))

Вопрос 88

После выполнения операторов

A = [1,2,3,4]

A.insert(-1,2)

Ответ: [1, 2, 3, 2, 4]

Вопрос 89

После выполения фрагмента

A = [[]] \* 3

A[0].append(3)

A[1]=1

Ответ: [[3], 1, [3]]

Вопрос 90

После выполнения операторов

s = '1***\*\*\*\*\*2'***

print(s.replace('**\*\*','\*'))**

Ответ: 1***\*\*\*2***

Вопрос 91

В следующем фрагменте кода

a = [1,2,3,4]

for b in a:

print(b)

a.pop()

Цикл выполнится ... раза

Ответ: 2

Вопрос 92

Следующий код

a = [[i + j for j in range(3)] for i in range(2)]

print (a)

Выведет

Ответ: [[0, 1, 2], [1, 2, 3]]

Вопрос 93

После выполнения фрагмента программы

a = 7; b = 3

if a > 5:

if b < 0:

print(a)

else:

print(b)

На экран будет выведено ...

Ответ: сообщение о ошибке

Вопрос 94

Любой объект в логическом контексте может интерпретироваться как True или False.

Значения True имеют следующие выражения:

Варианты:

bool({0:0})

bool(0.0)

bool('0')

bool([])

bool(5)

Ответ:

Вопрос 95

Строка " x = 1.41" может быть выведена на экран с помощью инструкции ...

Ответ: print(f'x = {2**\*\*0.5:5.2f}')**

Вопрос 96

После выполнения фрагмента программы

a = 2

b = (a + a % 5) \* a **\*\* a // 5**

Значение переменной b равно ...

Ответ: 3

Вопрос 97

Список, в котором включение и исключение элементов возможно только с обоих концов, называется ...

Ответ:

Список, в котором доступен только последний элемент называется

Стеком

Вопрос 98

Для массива справедливы следующие утверждения:

Варианты:

\* все элементы массива имеют одинаковый тип

\* время доступа к элементу с любым индексом одинаково

\* это структура данных с последовательным доступом

\* объем памяти при хранении значений в массиве является минимальынм по сравнению с другими структурами данных

\* вставка и удаление элементов выполняются за фиксированное время

Ответ:

Вопрос 99

Плохое фото

Вопрос 100

Имеется генератор

gn = (x\*x for x in range(5))

Без ошибок выполнятся следующие операторы:

Ответ:

\* print(next(gn))

\* for t in gn: print(t)

\* print(max(gn))

Вопрос 101

После выполнения операторов

funcs = [lambda x: x\*2, lambda x: x+2]

t = 5

v = sum(map(lambda x: x(t), funcs))

Ответ: 17

Вопрос 102

После выполнения операторов

li1 = [-3, 0, -4, -1]

li2 = [3, 25, -5, 1]

b = max(map(lambda x, y: x\*y, li1, li2))

Ответ: 20

Вопрос 103

После выполнения операторов

txt= 's1\ns2\ns3\ns4'

with open("my.txt", "w") as f:

print(txt, file=f)

with open("my.txt", "r") as f:

f1 = sum(len(line) for line in f)

переменная f1 будет равна ...

Ответ: 12

Скрин 128

Для структуры данных «дерево» справедливы следующие утверждения:

1. каждая вершина имеет не более двух потомков
2. является связным графом, не содержащим циклы
3. это структура данных с произвольным доступом
4. является набором связанных узлов
5. уровень всех листьев (расстояние до корня) отличается не более чем на 1

Ответ: 24

Скрин 130

Оператор

a = sin(pi)

выполнится корректно, если предварительно выполнить одну из инструкций:

1. from math import \*
2. import sin, pi from math
3. import math as \*
4. from math import sin, pi
5. import math

Ответ: 14

Скрин 131

Язык программирования Python поддерживает:

1. императивное программирование
2. объектно-ориентированное программирование
3. логическое программирование
4. функциональное программирование
5. процедурное программирование

Ответ:

Вопрос 124

Функциональное программирование поддерживается в языках программирования:

1. C++
2. Python
3. Lisp
4. Java
5. Haskell

Вопрос 136

Итератор-это…

1. объект, который позволяет поочередно перебирать элементы
2. синоним понятия “итерируемый объект”
3. класс, который используется для создания итерируемого объекта
4. выражение, которое позволяет поочередно получать элементы
5. функция, содержащая выражение yield

Вопрос 137

Полиморфизм позволяет…

1. разным объектам одного класса иметь разное поведение
2. ограничивать доступ к некоторым переменным и методам
3. работать с различными реализациями через один интерфейс
4. изменять набор методов в дочернем классе
5. скрывать реализацию от пользователя в классе

Ответ: 3

Вопрос 138

Имеется список [15, 17, 31, 10, 7]. После выполнения одного прохода алгоритма пузырьковой сортировки список будет иметь вид:

1. [7, 17, 10, 31, 15]
2. [15, 17, 10, 7, 31]
3. [15, 17, 7, 10, 31]
4. [10, 15, 17, 31, 7]
5. [10, 7, 15, 31, 17]

Ответ: 2

Имеется список [15, 17, 31, 10, 7]. После выполнения одного прохода алгоритма быстрой с разделяющим элементов 15 будет …

1. [15, 17, 10, 7, 31]

2. [15, 17, 7, 10, 31]

3. [10, 15, 17, 31, 7]

4. [7, 17, 10, 31, 15]

5. [10, 7, 15, 31, 17]

Вопрос 139

Для метода \_\_str\_\_ справедливы следующие утверждения. Метод:

1. автоматически вызывается при выводе объекта с помощью функции print
2. преобразует объект в строку
3. вызывается при использовании функции str для объекта
4. может иметь параметры кроме self
5. должен возвращать строку

Для алгоритма сортировки Шелла справедливо:

Для алгоритма быстрой сортировки справедливо:

В среднем имеет временную сложность

Хуже всего работает когда данные упорядочены по возрастанию

Требует доп память

Меняет взаимное расположение элементов с одинаковыми значениями

Для алгоритма сортировки выбором справедливо:

При работе не требуется доп память

Имеет временную сложность

При обработке уже упорядоченных или частично упорядочнеынных данных алгоритм работает быстрее

Меняет взаимное расположение элементов с одинаковыми значаниями

Имеет временную сложность

В наихудшем случае имеет временную сложность

Компилятор – это программа

Скрин 161

t = (1, 2, 3, 4, 5, 6)

a = 1

c = 7

Правильными являются следующие операторы:

1. a, \*b, c = t
2. t2 = [a], {c}, a
3. a, b = b, a
4. a, c, \_ = t
5. t1 = a, c

Для односвязного списка справедливы следующие утверждения:

Это структура данных с последовательным доступом

Вставка и удаление элементов выполняются за фиксированное время

Для алгоритма сортировки вставками:

Имеет временную сложность

При работе не требуется доп память

Имеет временную сложность

Двоичное дерево поиска удовлетворяет условиям:

У всех вершин правого поддерева вершины Х значени больше либо равны Х

У всех вершин левого поддерева вершины Х значения меньше Х

Каждая вершина имеет не более двух потомков

Двоичная куча удовлетворяет условиям:

У всех вершин левого поддерева вершины Х значения меньше значения в Х

Уровень всез листьев

Последний уровень заполняется слева направо

