

Билеты

Вариант 1

1. Что такое переменная? Какие типы данных вы знаете? Опишите их.

Переменная - это имя, с которым связано то или иное значение, либо же именованная область памяти, в которой хранится значение.

Типы данных:

- `int(integer)` - тип данных, хранящий целые числа как со знаком, так и без. Может иметь любую длину(из-за длинной арифметики). Примеры: 10; 91; -21453
- `float/double` - тип данных, хранящий числа с плавающей точкой, количество знаков после запятой ограничено .
Примеры: -1,342; 3,14; 90234,7546
- `string` или строка - тип данных, хранящий последовательность символов. Пример: "sdgsdgsdgsdgsdsgd"
- Список - тип данных, представляющий из себя коллекцию элементов. Пример: [1,2,3,4,5]. Может хранить как элементы одного типа, так и различных: [1, 2.0, "sdaa", [1,2]].
Динамически расширяемый(нестатический)
- Кортеж - константный тип данных, хранящий различные элементы. Похож на список, однако его элементы нельзя изменить, да и места в памяти он занимает поменьше. Пример: (1,2,3)
- Словарь - тип данных, хранящий элементы в виде <ключ>:<значение>. Пример {1:"Moscow", 2:"Kazan"}

2. Составьте 4 сложных логических выражения с помощью оператора **and**, два из которых должны давать истину, а два других - ложь.

```
a = True
b = False
c = True
print(a and b and c)
print(a and not b and c)
print(not a and c and not b)
print((a or b) and c)
```

False
True
False
True

Думаю, что он подразумевал именно это

3. Напишите цикл, выводящий каждое третье число от -1 до 21

```
1
4
7
10
13
16
19
for i in range(1, 22, 3):
    print(i)
```

4. Дана строка str. Напишите срез, который получит из str строку 'nkhe'

```
str = 'abcdefghijklmno'
print(str[-3:-13:-3])
```

5. Напишите функцию, которая вычисляет сумму трёх чисел и возвращает результат в основную ветку программы. При этом два числа функция получает в виде параметров, а третье хранится в некоторой глобальной переменной.

```
c = 5
5
def f(x, y):
    return x + y + c
5
a = int(input())
b = int(input())
s = f(a, b)
15
print(s)
```

Вариант 2

1. Можно ли преобразовать дробное число в целое? наоборот? В каких случаях строку можно преобразовать в число?

Дробное число можно преобразовать в целое: результатом будет целая часть исходного числа. Обратный ход тоже возможен: получится число с целой частью, равной исходному числу, и дробной частью, равной 0.

Строку можно преобразовать число, если она содержит только цифры в качестве символов, если строка в начале имеет нули, то они просто отбросятся.

```
print(int(88.1))
print(float(81))
print(int("1235"))
print(int("000000123"))
print(int("asd123"))
```

```
Traceback (most recent call last):
  File "C:\Users\Nikita\PycharmProjects\KAI\FirstCourse\FirstSemester\Labs\Lab5\s.py", line 29, in <module>
    print(int("asd123"))
ValueError: invalid literal for int() with base 10: 'asd123'
88
81.0
1235
123
```

2. Составьте 4 сложными логическими выражения с помощью оператора **or**, два из которых должны давать истину, а два других - ложь.

```
a = False
b = True
c = True
print(a or b or c)
print(a or not b or c)
print(a or not b or not c)
print(not(a or b or c))
```

3. Напишите программу, которая предлагала бы пользователю решить пример $4 \cdot 100 - 54$. Если пользователь напишет правильный ответ, то получит поздравление от программы, иначе программа предложит решить пример снова.

```

right_ans = 346
while True:
    print("Предлагаю решить вам пример 4*100 - 54 = ...")
    ans = input("Ваш ответ: ")
    try:
        ans = int(ans)
    except ValueError:
        print("Это даже не число! Попробуйте снова!")
        continue
    if ans == right_ans:
        print("Мои поздравления! Вы величайший математик всех времён!")
        break
    else:
        print("Неверно :( Попробуйте еще раз!")

```

```

Предлагаю решить вам пример 4*100 - 54 = ...
Ваш ответ: я на дурака похож?
Это даже не число! Попробуйте снова!
Предлагаю решить вам пример 4*100 - 54 = ...
Ваш ответ: 90
Неверно :( Попробуйте еще раз!
Предлагаю решить вам пример 4*100 - 54 = ...
Ваш ответ: 346
Мои поздравления! Вы величайший математик всех времён!

```

4. Дана строка str. Напишите срез, который получит из str строку pib

```

str = 'abcdefghijklmnop'
print(str[-1:-16:-7])

```

5. Написать программу которая показывает какую-то геометрическую фигуру (например зелёный овал), и чтобы при нажатии правой кнопкой мыши на эту фигуру она исчезла.

<Button-3> правильно(фикс)

```
from tkinter import *

def del_ov(event):
    c.delete(oval)

root = Tk()
c = Canvas(root, width = 500, height = 500, bg = "white")
c.pack()
oval = c.create_oval(100,100,300,300, fill = "green")
c.tag_bind(oval, "<Button-1>", del_ov)

root.mainloop()
```

Вариант3

1. Приведите примеры арифметических операций. Для чего предназначена операция присвоения?

```
1 print(5 + 4)
2 print(3 ** 2)
3 print(18-9)
4 print(3//2)
5
6
```

```
C:\prog\venv\Scripts\python.exe C:/prog/exam.py
9
9
9
1

Process finished with exit code 0
```

Операция присвоения предназначена для того,

- 1) чтобы соединять элементы из множества имен с элементами из множества значений
 - 2) чтобы присваивать переменным определенные значения
2. Придумайте программу, в которой бы использовалась инструкция if-elif-else. Количество веток должно быть как минимум четыре

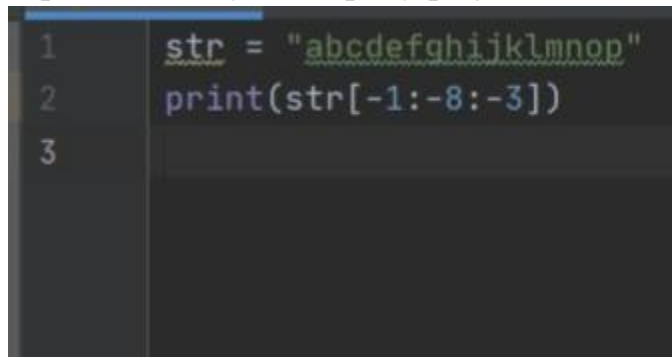
```
points = int(input("Баллы ЕГЭ: "))
if points < 220:
    print("Стипка 2200")
elif points < 240:
    print("Стипка 4400")
elif points < 260:
    print("Стипка 6600")
elif points < 280:
    print("Стипка 8800")
else:
    print("Стипка 15000")
```

Или еще пример:

3. Цикл, выводящий четные числа в диапазоне от 0 до 20.

```
for i in range(21):  
    if i%2 == 0:  
        print(i, end = " ")
```

4. Дана строка `str = "abcdefghijklmnop"` Напишите срез который из строки `str` получит строку `pmj`.



```
1 str = "abcdefghijklmnop"  
2 print(str[-1:-8:-3])  
3
```

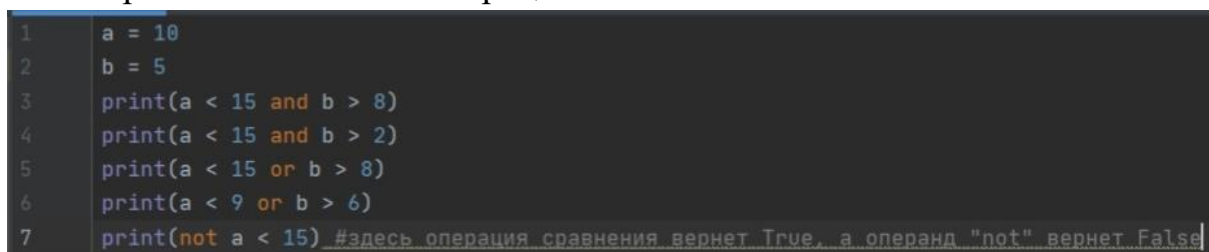
5. Приведите пример программы, которая рисует какую-нибудь фигуру, при нажатии на нее она исчезает.

```
from tkinter import *  
  
def dlet(event):  
    c.delete(oval)  
root = Tk()  
c = Canvas(root, width = 500, height = 500, bg = "white")  
c.pack()  
oval = c.create_oval(100, 100, 200, 200,  
                     fill = "orange",  
                     outline = "black")  
c.tag_bind(oval, "<Button-1>", dlet)  
  
root.mainloop()
```

Вариант4

1. Приведите примеры логических операций. Для чего предназначена операция присвоения?

К логическим операциям относятся "and", "or" и "not". Примеры с применением таких операций:



```
1 a = 10  
2 b = 5  
3 print(a < 15 and b > 8)  
4 print(a < 15 and b > 2)  
5 print(a < 15 or b > 8)  
6 print(a < 9 or b > 6)  
7 print(not a < 15) #здесь операция сравнения вернет True, а операнд "not" вернет False
```

```
C:\prog\venv\Scripts\python.exe C:/prog/exam.py
False
True
True
False
False

Process finished with exit code 0
```

Операция присвоения предназначена для того,

- 1) чтобы соединять элементы из множества имен с элементами из множества значений
 - 2) чтобы присваивать переменным определенные значения
2. Присвойте переменной var_str значение “NoNoYesYesYes”. При формировании значения используйте операции конкатенации и повторения строки и переменные strNo = “No” strYes = “Yes”

```
no = "No"
ya = "Yes"
print(no*2+ya*3)
```

3. Напишите программу, которая предлагала решить пользователю $4 \cdot 100 - 54$. Если пользователь напишет правильный ответ, поздравьте его. Иначе сообщит ему об ошибке. Код универсальный, не ругайте))

```
print("ЗАДАЧА")
a = int(input())
while a != 300:
    a = int(input("Ваш ответ неверен, решите заново\n"))
print("Вы молодец!!!")
```

4. Str = “abcdefghijklmnop” Напишите срез, который получит “oje”

```
str = "abcdefghijklmnop"
print(str[-2:-13:-5])
```

5. Вывод на экран содержимого text.txt который расположен в директории диска E: . В случае неудачной попытки сообщите об ошибке.


```
try:
    file = open("E:\text.txt", "r")
    readed = file.read()
    print(readed)
except IOError:
    print("You have error. Try again plz.")
finally:
    file.close()
```

Дополнительные билеты

Вариант 1

- 1) Каков будет результат выполнения `int("88")`?
88 :)
- 2) Списки, кортежи и словари. Описание, примеры использования.
 - Список - тип данных, представляющий из себя коллекцию элементов. Пример: [1,2,3,4,5]. Может хранить как элементы одного типа, так и различных: [1, 2.0, "sdaa", [1,2]]. Динамически расширяемый(нестатический). Используется для хранения чего-либо, удобен, если количество элементов неизвестно, как и их значение.
 - Кортеж - константный тип данных, хранящий различные элементы. Похож на список, однако его элементы нельзя изменить, да и занимает места в памяти он поменьше. Пример: (1,2,3). Используем для хранения констант, допустим для хранения координат точек на оси по x, если они известны заранее.
 - Словарь - тип данных, хранящий элементы в виде <ключ>:<значение>. Пример {1:"Moscow", 2:"Kazan"}. Также называют хэш-таблицами. Удобно, когда необходимо найти количество вхождений какого-то элемента(строим словарь, где ключ есть сам элемент, а значением будет количество вхождений)

- 3) Написать программу, которая вычисляет произведение элементов списка с четным индексом

```
a = [32, 45, 245, 35, 53, 53]
k = 1
for i in range(len(a)):
    if i%2 == 0:
        k*=a[i]
print(k)
```

Вариант 2

- 1) Каков будет результат выполнения `str(88)`?
"88"
- 2) Потоки. Описание, примеры использования
Потоки - 7 лекция
- 3) Написать программу, в которой дочерний поток выводит на экран последовательность чисел от 1 до 10.

```
import threading

def thr():
    for i in range(1,11):
        print(i, end= " ")

child = threading.Thread(target= thr, args = ())

child.start()
```

Вариант 3

- 1) Каков будет результат выполнения `float(88)`?
88.0
- 2) Приведите описание и примеры использования 3-5 известных вам виджетов Tkinter(кнопка, холст и т.д)
Button - кнопка;
Canvas - рисунок;
Menu - меню;
Scrollbar - полоса прокрутки;
Text - форматированный текст;
Entry - поле ввода
и т.д.
- 3) Напишите программу, которая имеет на своём окне кнопку, при нажатии на которую увеличивается значение целочисленной переменной counter.

```

from tkinter import *

count = 0
def ct(event):
    global count
    count+=1
    print(count)

root = Tk()

But = Button(text = 'КНОПКА')
But.pack()
But.bind("<Button-1>", ct)

root.mainloop()

```

Вариант 4

- 1) Каков будет результат выполнения `int(88.0)`?

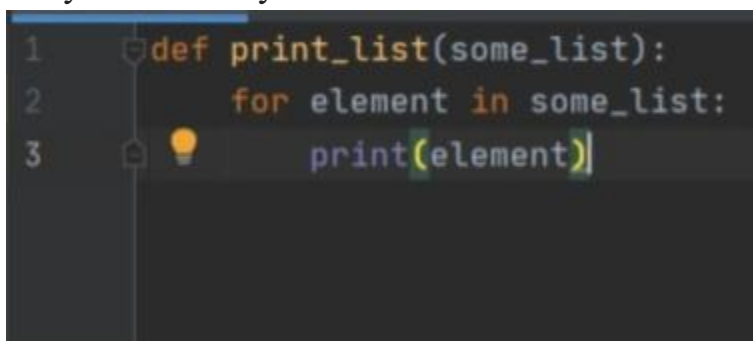
88

- 2) Функции, области видимости переменных.

Функции -5 лекция

Функции по области видимости делятся на 3 типа: локальные, глобальные и нелокальные (добавлены в Python 3)

Грубо говоря, локальные переменные, это такие переменные, работа с которыми происходит только в функции. "element" и "some_list" – локальные переменные, которые видны только внутри функции, и которые не могут использоваться за ее пределами с теми значениями, которые были им присвоены внутри функции при ее работе. То есть, если мы в основном теле программы вызовем `print(element)`, то получим ошибку



```

1 def print_list(some_list):
2     for element in some_list:
3         print(element)

```

В Python есть ключевое слово "global", которое позволяет изменять изнутри функции значение глобальной переменной. Оно

записывается перед именем переменной, которая дальше внутри функции будет считаться глобальной. Как видно из примера, теперь значение переменной “candy” увеличивается, и обратите внимание на то, что мы не передаем ее в качестве аргумента функции “get_candy()”

```
1  candy = 5
2
3  def get_candy():
4      global candy
5      candy += 1
6      print('У меня {} конфет.'.format(candy))
7
8  get_candy()
9  get_candy()
10 print(candy)
```

а вот нелокальная переменная видна в пределах объемлющей инструкции def(хз какой пример привести)

- Изнутри функции видны переменные, которые были определены и внутри нее и снаружи. Переменные, определенные внутри – локальные, снаружи – глобальные.
- Снаружи функций не видны никакие переменные, определенные внутри них.
- Изнутри функции можно изменять значение переменных, которые определены в глобальной области видимости с помощью спецификатора global

3) Напишите программу, которая рекурсивно вычисляет факториал числа, содержащегося в файле input.txt(textik.txt).

Рекурсивное вычисление факториала числа, содержащегося в файле textik.txt.

```
with open("textik.txt", "r") as file:
    readed = file.read()

def fact(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * fact(n-1)
print(fact(int(readed)))
```

и для тех, кто предохраняется

```
def fact(n):  
    if n == 0 or n == 1:  
        return 1  
    else:  
        return n * fact(n - 1)  
  
try:  
    file = open("D:/py.txt", "r")  
    for line in file:  
        num = line  
        print(fact(int(num)))  
except FileNotFoundError:  
    print("Файл не найден")  
except IOError:  
    print("Невозможно открыть файл")  
except ValueError:  
    print("В файле не число")  
else:  
    file.close()
```