СОБЫТИЯ

ТЕХНИКА БЕЗОПАСНОСТИ

Запрещается:

· трогать питающие провода и разъемы соединительных кабелей;

· прикасаться к экрану и тыльной стороне монитора;

· размещать на рабочем месте посторонние предметы;

· самостоятельно устранять неисправности в работе аппаратуры;

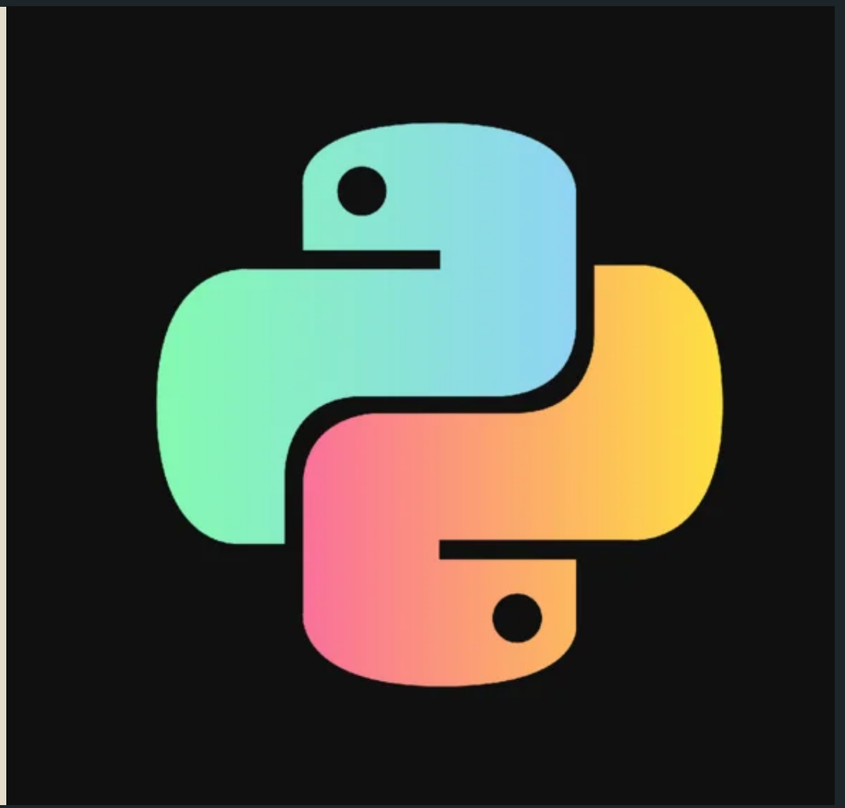
· при неполадках и сбоях в работе компьютера немедленно прекратите работу и сообщите об этом педагогу;

· работайте на клавиатуре чистыми, сухими руками;

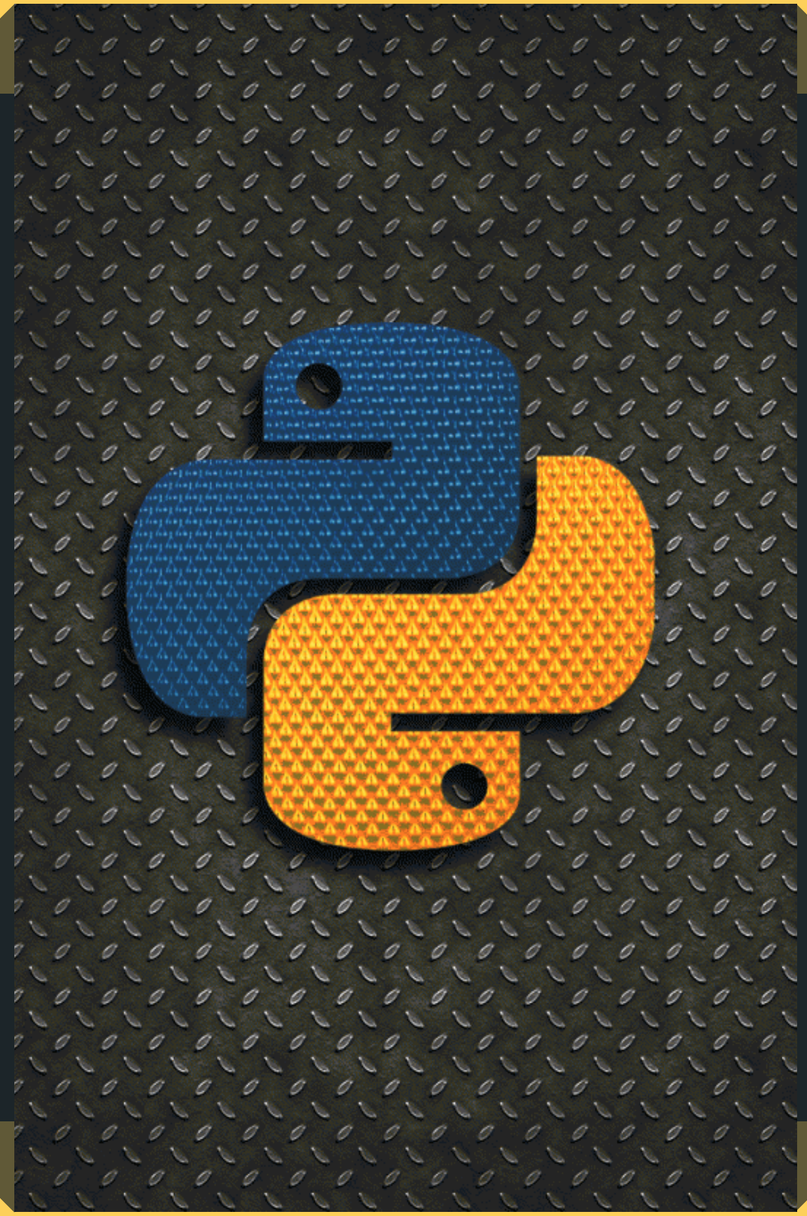
· легко нажимайте на клавиши, не допуская резких ударов и не задерживая клавиши в зажатом состоянии.

ЗАДАНИЕ НА ПОВТОРЕНИЕ

Напишите фрагмент кода для отображения главного окна, на котором должен располагаться вопрос, под ним варианты ответов (checkbutton или radiobutton) и кнопка.



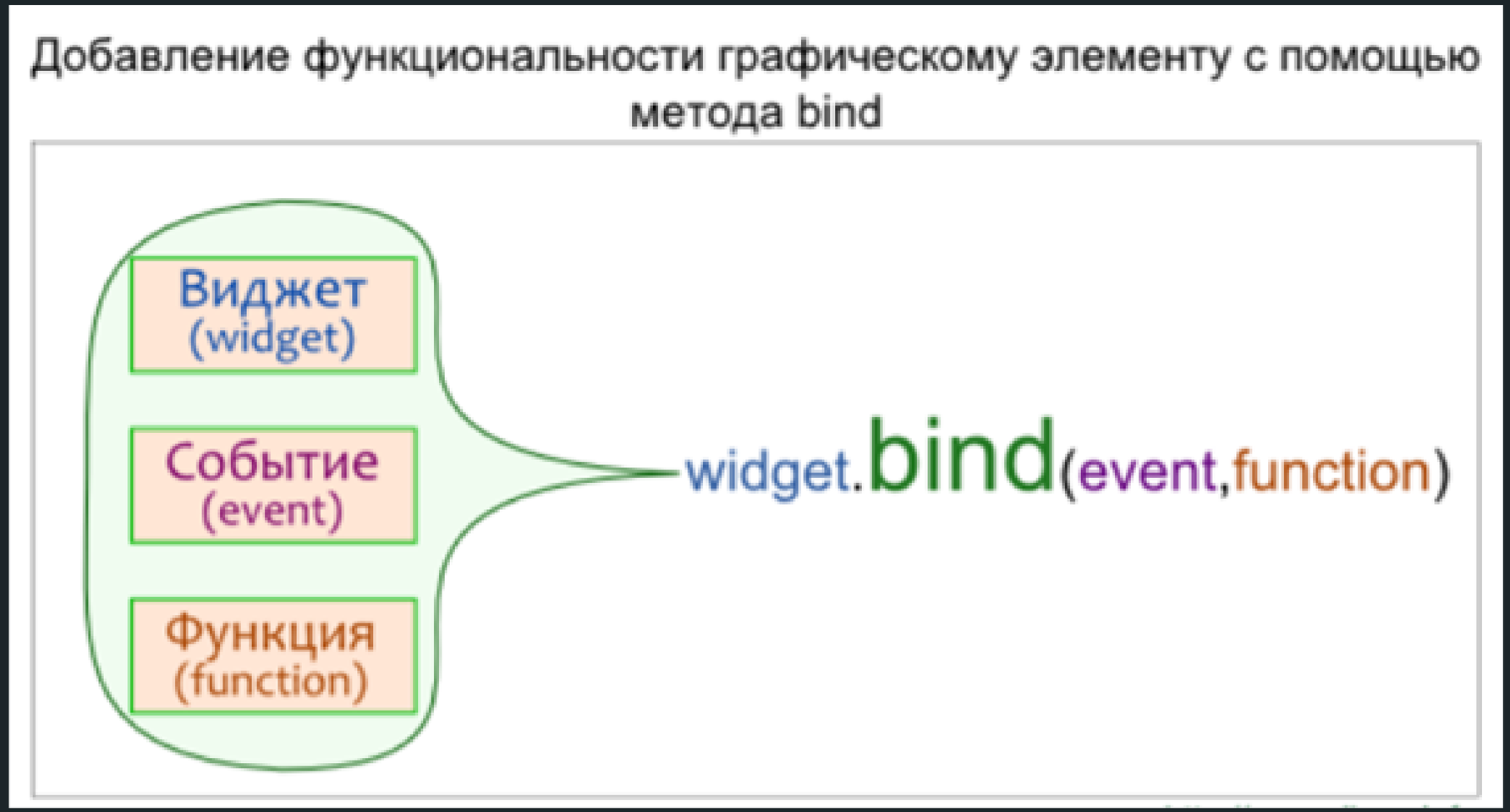
СОБЫТИЯ



“

Внешнее воздействие на графический компонент называется событием.

ОДНИМ ИЗ СПОСОБОВ СВЯЗЫВАНИЯ ВИДЖЕТА, СОБЫТИЯ И ФУНКЦИИ (ТОГО, ЧТО ДОЛЖНО ПРОИСХОДИТЬ ПОСЛЕ СОБЫТИЯ) ЯВЛЯЕТСЯ ИСПОЛЬЗОВАНИЕ МЕТОДА BIND.



ТИПЫ СОБЫТИЙ

Можно выделить три основных типа событий:

1. Производимые мышью.
2. Нажатиями клавиш на клавиатуре.
3. События, возникающие в результате изменения других графических объектов.

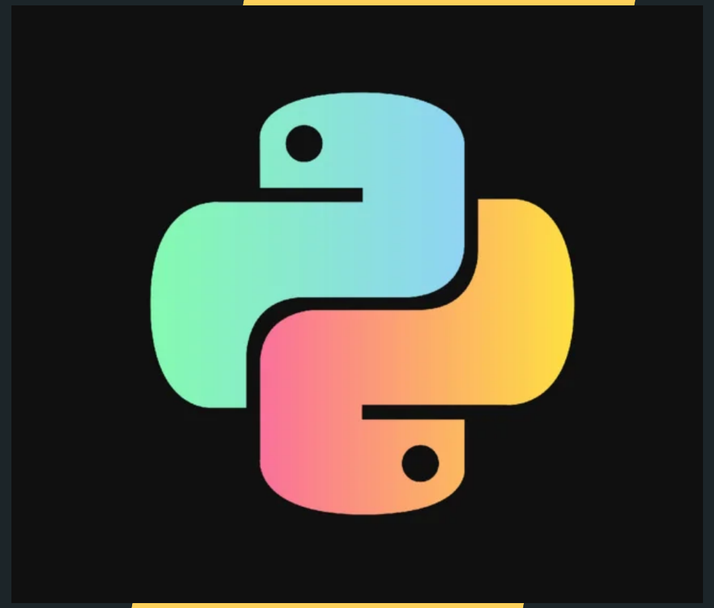
СПОСОБ ЗАПИСИ

ПРИ ВЫЗОВЕ МЕТОДА BIND СОБЫТИЕ ПЕРЕДАЕТСЯ В КАЧЕСТВЕ ПЕРВОГО АРГУМЕНТА.

НАЗВАНИЕ СОБЫТИЯ ЗАКЛЮЧАЕТСЯ В КАВЫЧКИ, А ТАКЖЕ В ЗНАКИ < И >. СОБЫТИЕ ОПИСЫВАЕТСЯ С ПОМОЩЬЮ ЗАРЕЗЕРВИРОВАННЫХ ПОСЛЕДОВАТЕЛЬНОСТЕЙ КЛЮЧЕВЫХ СЛОВ.

СОБЫТИЯ, ПРОИЗВОДИМЫЕ МЫШЬЮ

<Button-1> - щелчок левой кнопкой мыши,
<Button-2> - щелчок средней кнопкой мыши,
<Button-3> - щелчок правой кнопкой мыши,
<Double-Button-1> - двойной клик левой кнопкой мыши,
<Motion> - движение мыши и т. д.



Пример

```
from tkinter import *
def b1(event):
    root.title("Левая кнопка мыши")
def b3(event):
    root.title("Правая кнопка мыши")
def move(event):
    root.title("Движение мышью")
root = Tk()
root.geometry('400x400')
root.bind('<Button-1>',b1)
root.bind('<Button-3>',b3)
root.bind('<Motion>',move)
root.mainloop()
```

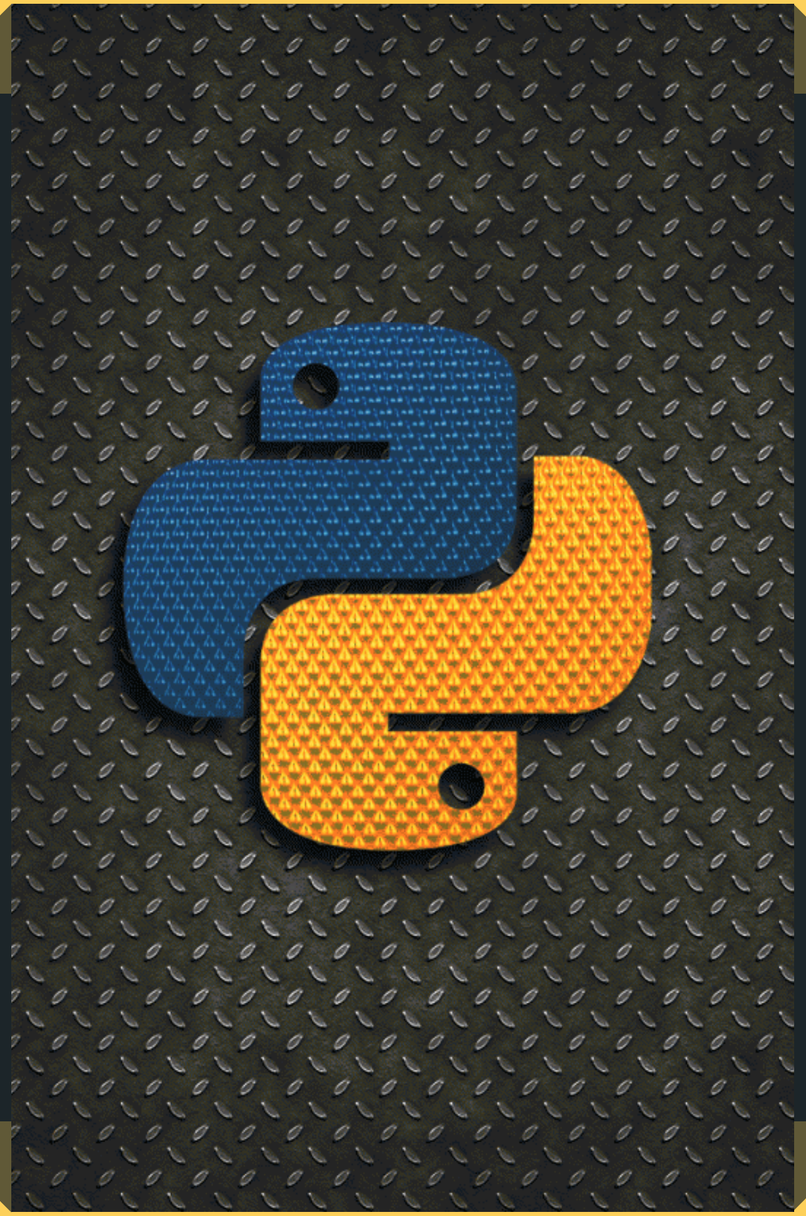
СОБЫТИЯ, ПРОИЗВОДИМЫЕ С ПОМОЩЬЮ КЛАВИАТУРЫ

-БУКВЕННЫЕ КЛАВИШИ МОЖНО ЗАПИСЫВАТЬ БЕЗ УГЛОВЫХ СКОБОК (НАПРИМЕР, 'L').
-ДЛЯ НЕАЛФАВИТНЫХ КЛАВИШ СУЩЕСТВУЮТ СПЕЦИАЛЬНЫЕ ЗАРЕЗЕРВИРОВАННЫЕ СЛОВА (<RETURN> - НАЖАТИЕ КЛАВИШИ ENTER; <SPACE> - ПРОБЕЛ; И Т. Д.)
-СОЧЕТАНИЯ КЛАВИШ ПИШУТСЯ ЧЕРЕЗ ТИРЕ. НАПРИМЕР: <CONTROL-SHIFT> (ОДНОВРЕМЕННОЕ НАЖАТИЕ КЛАВИШ CTRL И SHIFT).

Пример

```
from tkinter import *
def exit_(event):
    root.destroy()
def caption(event):
    t = ent.get()
    lbl.configure(text = t)
root = Tk()
ent = Entry(root, width = 40)
lbl = Label(root, width = 80)
ent.pack()
lbl.pack()
ent.bind('<Return>',caption)
root.bind('<space>',exit_)
root.mainloop()
```

РЕШЕНИЕ ЗАДАЧ



ЗАДАЧА 1

Сделайте так, чтобы при нажатии на кнопку правой кнопкой мыши изменялся цвет кнопки и появлялась надпись на ней.

ЗАДАЧА 1

```
from tkinter import *

def b3(event):
    bt['bg']='red'
    bt['text']='Нет'
root = Tk()
root.geometry('400x400')
bt = Button (root,width = 10,bg = 'yellow')
bt.pack()
bt.bind('<Button-3>',b3)
root.mainloop()
```

ЗАДАЧА 2

Напишите фрагмент кода, в котором при нажатии клавиши A изменялся цвет и размер Label.

ЗАДАЧА 2

```
from tkinter import *

def A(event):
    lb.configure(fg = 'blue',font=(50))
root = Tk()
root.geometry('400x400')
lb = Label (root,text = 'Привет')
lb.pack()
root.bind('a',A)
root.mainloop()
```

ЗАДАЧА 3

Напишите часть кода, в которой при наведении на текстовую область Entry будет появляться текст.

ЗАДАЧА 3

```
from tkinter import *
def text(event):
    en.insert(0,'Привет')
root = Tk()
root.geometry('400x400')
en = Entry(root, width = 100)
en.pack()
en.bind('<Enter>',text)
root.mainloop()
```

СОЗДАНИЕ ФУНКЦИЙ

ОПРОС

ЦИКЛ С ПРЕДУСЛОВИЕМ (WHILE) - ЭТО ЦИКЛ,КОТОРЫЙ:

А) ПОВТОРЯЕТСЯ ОПРЕДЕЛЕННОЕ ЧИСЛО РАЗ

Б) ПОВТОРЯЕТСЯ ДО ТЕХ ПОР, ПОКА УСЛОВИЕ ВЕРНОЕ

В) ПОВТОРЯЕТСЯ ДО ТЕХ ПОР, ПОКА УСЛОВИЕ ЛОЖНОЕ

ЦИКЛ С ПРЕДУСЛОВИЕМ (WHILE) - ЭТО ЦИКЛ,КОТОРЫЙ:

А) ПОВТОРЯЕТСЯ ОПРЕДЕЛЕННОЕ ЧИСЛО РАЗ

Б) ПОВТОРЯЕТСЯ ДО ТЕХ ПОР, ПОКА УСЛОВИЕ ВЕРНОЕ

В) ПОВТОРЯЕТСЯ ДО ТЕХ ПОР, ПОКА УСЛОВИЕ ЛОЖНОЕ

ЦИКЛ ИСПОЛЬЗУЕТСЯ В ТЕХ СЛУЧАЯХ, ...

А) когда вам нужно выполнить что-нибудь при определенных условиях.

Б) когда вам нужно повторить что-нибудь n-ное количество раз.

В) когда вам нужно проверить верность определенного условия.

ЦИКЛ ИСПОЛЬЗУЕТСЯ В ТЕХ СЛУЧАЯХ, ...

А) когда вам нужно выполнить что-нибудь при определенных условиях.

Б) когда вам нужно повторить что-нибудь n-ное количество раз.

В) когда вам нужно проверить верность определенного условия.

ЧТО ВЫВЕДЕТ ДАННЫЙ ФРАГМЕНТ КОДА?

```
a = 1
b =3
while(a<5):
        a+=3
        b+=a
print(b)
```

ЧТО ВЫВЕДЕТ ДАННЫЙ ФРАГМЕНТ КОДА?

```
a = 1
b =3
while(a<5):
        a+=3
        b+=a
print(b)
```

Ответ: 14.

КАКИМ БУДЕТ РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ СЛЕДУЮЩЕЙ ЧАСТИ СКРИПТА?

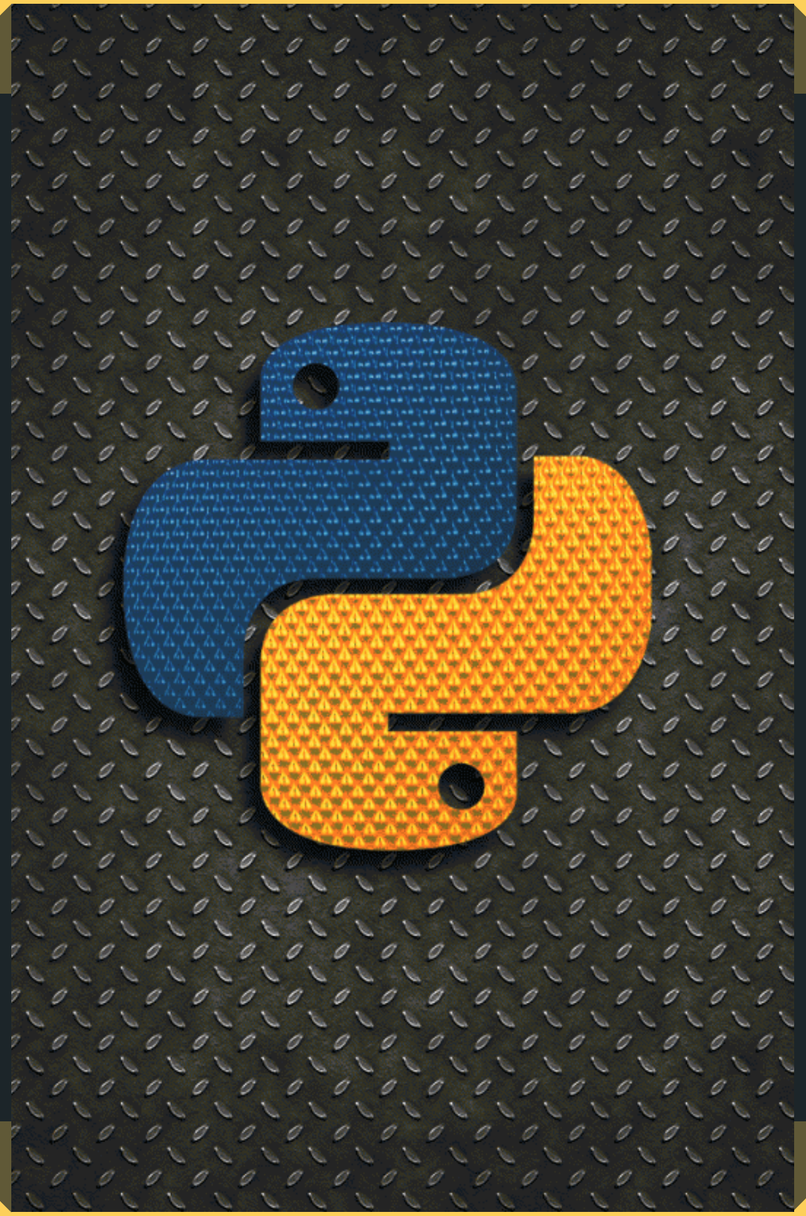
```
s = 1
for k in range(1,30):
        s = (k – 5) * s
print(s)
```

КАКИМ БУДЕТ РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ СЛЕДУЮЩЕЙ ЧАСТИ СКРИПТА?

```
s = 1
for k in range(1,30):
        s = (k – 5) * s
print(s)
```

Ответ: 0.

ФУНКЦИИ



“

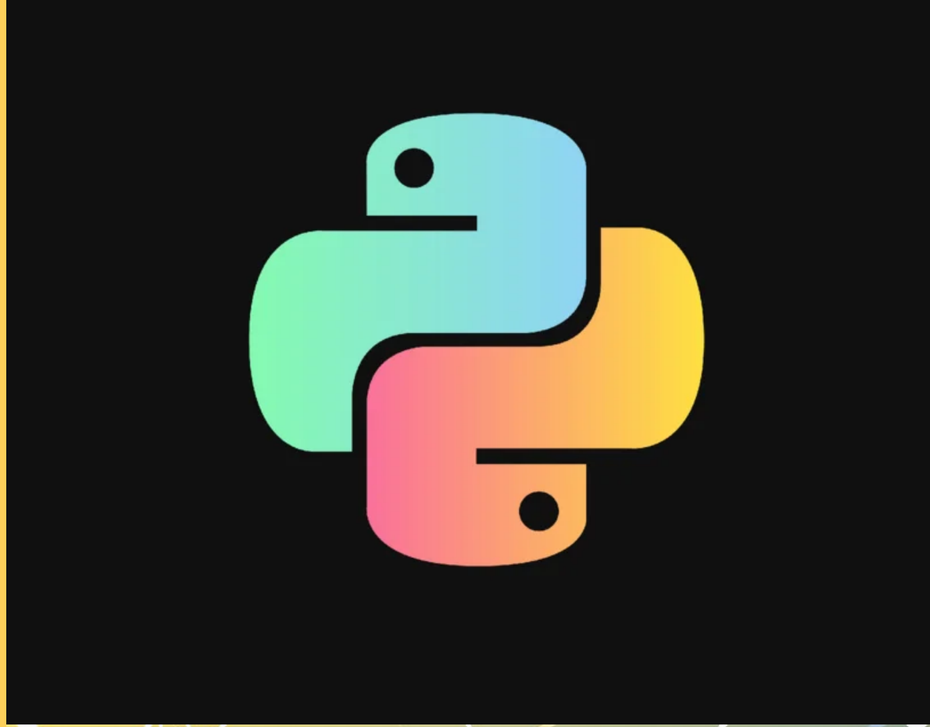
Функция это блок организованного, многократно используемоего кода, который используется для выполнения конкретного задания.

СУЩЕСТВУЮТ НЕКОТОРЫЕ ПРАВИЛА ДЛЯ СОЗДАНИЯ ФУНКЦИЙ:

- Блок функции начинается с ключевого слова def, после которого следуют название функции и круглые скобки ().
- Любые аргументы, которые принимает функция должны находиться внутри этих скобок.
- После скобок идет двоеточие (:) и с новой строки с отступом начинается тело функции.

ПРИМЕР ФУНКЦИИ:

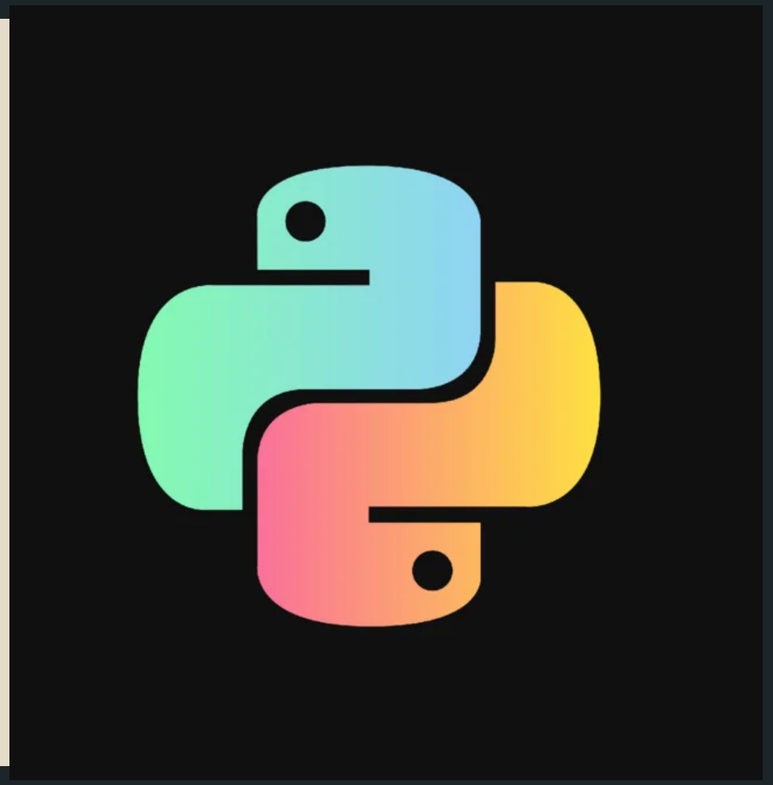
```
def my_function(argument):
    print argument
```



ВЫЗОВ ФУНКЦИИ

После создания функции, ее можно исполнять вызывая из другой функции или напрямую из оболочки Python. Для вызова функции следует ввести ее имя и добавить скобки.

```
my_function("abracadabra")
```



АРГУМЕНТЫ ФУНКЦИИ

Вызывая функцию, мы можем передавать ей следующие типы аргументов:

- Обязательные аргументы (Required arguments)
- Аргументы-ключевые слова (Keyword argument)
- Аргументы по умолчанию (Default argument)
- Аргументы произвольной длины (Variable-length argumens)

ОБЯЗАТЕЛЬНЫЕ АРГУМЕНТЫ ФУНКЦИИ

Если при создании функции указано количество передаваемых ей аргументов и их порядок, то и вызывать ее нужно с тем же количеством аргументов, заданных в нужном порядке.

```
def my_function(a,b):
    print(a,b)

# В описании функции указано, что она принимает 2 аргумента
# Корректное использование функции
my_function(5,6)

# Некорректное использование функции
my_function()
my_function(3)
my_function(12,7,3)
```

АРГУМЕНТЫ - КЛЮЧЕВЫЕ СЛОВА

Аргументы - ключевые слова используются при вызове функции. Благодаря ключевым аргументам, вы можете задавать произвольный (то есть не такой каким он описа при создании функции) порядок аргументов.

```
def person(name, age):
    print name, "is", age, "years old"

# Хотя в описании функции первым аргументом
# идет имя, мы можем вызвать функцию вот так

person(age=23, name="John")
```

АРГУМЕНТЫ, ЗАДАННЫЕ ПО-УМОЛЧАНИЮ

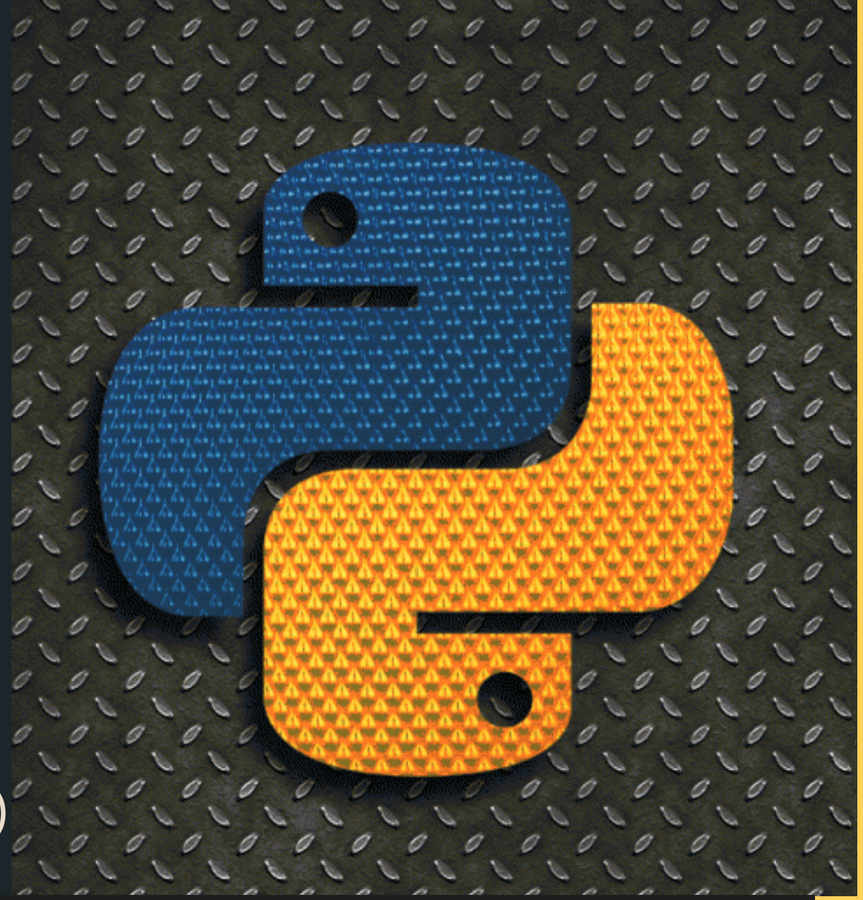
Аргумент по умолчанию, это аргумент, значение для которого задано изначально, при создании функции.

```
def space(planet_name, center="Star"):
    print planet_name, "is orbiting a", center

# Можно вызвать функцию space так:
space("Mars")
# В результате получим: Mars is orbiting a Star

# Можно вызвать функцию space иначе:
space("Mars", "Black Hole")
# В результате получим: Mars is orbiting a Black Hole
```

АРГУМЕНТЫ ПРОИЗВОЛЬНОЙ ДЛИНЫ



Иногда возникает ситуация, когда вы заранее не знаете, какое количество аргументов будет необходимо принять функции. В этом случае следует использовать аргументы произвольной длины. Они задаются произвольным именем переменной, перед которой ставится звездочка (*)

```
def unknown(*args):
    for argument in args:
        print argument

unknown("hello","world") # напечатает оба слова, каждое с новой строки
unknown(1,2,3,4,5) # напечатает все числа, каждое с новой строки
unknown() # ничего не выведет
```

КЛЮЧЕВОЕ СЛОВО RETURN

Выражение return прекращает выполнение функции и возвращает указанное после выражения значение.

```
def bigger(a,b):
    if a > b:
        return a # Если a больше чем b, то возвращаем a и прекращаем выполнение функции
    return b # Незачем использовать else.
# Если мы дошли до этой строки, то b, точно не меньше чем a

# присваиваем результат функции bigger переменной num
num = bigger(23,42)
```

ФИО_____

Итоговое тестирование

по кружку «Python»

1. Где правильно создана переменная?
    a. int num = 2
    b. Нет подходящего варианта
    c. var num = 2
    d. $num = 2
    e. num = float(2)
2. Какая функция выводит что-либо в консоль?
    a. write();
    b. log();
    c. print();
    d. out();
3. Сколько библиотек можно импортировать в один проект?
    a. Не более 10
    b. Неограниченное количество
    c. Не более 3
    d. Не более 5
    e. Не более 23
4. Как получить данные от пользователя?
    a. Использовать метод get()
    b. Использовать метод input()
    c. Использовать метод cin()
    d. Использовать метод read()
    e. Использовать метод readLine()
5. Цикл с предусловием (while) - это цикл,который :
    f. Повторяется определенное число раз
    g. Повторяется до тех пор, пока условие верное
    h. Повторяется до тех пор, пока условие ложное
    i. Ни один из выше перечисленных
6. С какого ключевого слова начинается блок функции?
    a. oct
    b. len
    c. set
    d. range
    e. dir
    f. def
    g. dict
    h. for
    i. Ни один из выше перечисленных
7. Какие существуют типы переменных (выбрать несколько выриантов):
    a. float

b. str
c. num
d. int
e. bool
f. real

8. Что будет в результате выполнения следующего алгоритма:
Входные данные: 57

```
x = int(input())
if x > 0:
   print(x)
else:
   print(-x)
```

9. Выберите правильный синтаксис цикла for.
   a. for <переменная> in <последовательность>:
      <действие>
      else:
      <действие>
   b. for <переменная> in <действие>:
      <последовательность>
      else:
      <последовательность>
   c. for <действие> in <последовательность>:
      <переменная>
      else:
      <переменная>
   d. Ни один из выше перечисленных

10. Какие имена являются правильными в Python?
   a. G
   b. ABC
   c. 41N
   d. Game2
   e. a+b
   f. _ab

11. Определите, что будет напечатано в результате выполнения следующего скрипта:

```
s = 1
for k in range(30):
      s = (–1) * s
print(s)
```

   a. 25
   b. 0
   c. 9
   d. 1
   e. -5
   f. 320

g. Ни один из выше перечисленных

12. Имена переменных не могут включать:

a. Русские буквы
b. Латинские буквы
c. Пробелы
d. Скобки, знаки + = ! ? b др.
e. Цифры

13. Какой класс Tkinter соответствует виджету для поля ввода?

a. Label
b. Text
c. Frame
d. Entry

14. Функция - это...

a. блок организованного, однократно используемого кода, который используется для выполнения различных задач.
b. блок организованного, однократно используемого кода, который используется для выполнения конкретных задач.
c. блок организованного, многократно используемоего кода, который используется для выполнения конкретного задания.
d. Ни один из выше перечисленных

15. Выберите правильный синтаксис функции.

a. def my_function argument
   print(1)
b. def my_function argument:
       print(1)
c. def my_function argument:
   print(1)
d. def my_function (argument):
   print(1)
e. def my_function (argument)
       print(1)
f. Ни один из выше перечисленных

16. Какой шаблон можно использовать для события, при котором клавиша F5 нажимается при нажатой клавише Shift?

a. <ButtonPress-Shift-F5>
b. <Shift-F5>

c. <Shift_L-F5> и <Shift_R-F5>

d. <F15>

17. Выберите правильные варианты списка:

a. l = { 'C', 'C#', 'Java'}
b. l = ['C', 'C#', 'Java']
c. l = ["C", "C#", "Java"]
d. l = [1, 2, 3, 4, 5, 6, 7]
e. l = {"C", "C#", "Java"}
f. l = [C, C#, Java]
g. l = ("C", "C#", "Java")
h. l = ["C', "C#", "Java"]
i. Ни один из выше перечисленных

18. Какой класс Tkinter соответствует виджету для вывода графических примитивов?

a. Label
b. Text
c. Frame
d. Canvas

19. Как называется встроенный в языке Python тип данных неупорядоченной коллекции из нуля или более пар ключ-значение?

a. dict
b. set
c. list
d. frozenset

20. Что делает команда import?

a. создает переменную
b. импортирует файл модуля
c. удаляет файл
d. создает функцию

21. Выберите вариант правильного удаления переменной a.

a. delete(a)
b. delete=a
c. del(a)

22. Закончите предложение по смыслу. Расширение файла Python – as. …

23. Завершите предложение. Символ # в Python обозначает …

24. Добавьте окончание предложения по смыслу. Функция len(строка) – возвращает …

25. Напишите обозначение. Условный оператор в Python - …

26. Что хранит в себе переменная?

a. Имя
b. Значение
c. Тип
d. Длину своего значения

27. Что лучше использовать для множественного ветвления?

a. if – elif –else
b. Много if
c. if – else – elif
d. while

28. Для чего нужен оператор break?
a. Для завершения программы
b. Для выхода из цикла
c. Для поломки компьютера
d. Для удаления программы

29. Как добавить модуль в программу?
a. import math
b. import math()
c. import (math)
d. import.math

30. От чего язык программирования называется «Питон»?
a. В честь змеи
b. В честь ТВ-шоу
c. В честь игры
d. В честь блюда

31. Выберите циклический алгоритм
a.
```
k = 0
while k < 10:
    print(“Привет”)
    k += 1
```
b.
```
a = int(input())
b = int(input())
c = int(input())
s = a+b+c
print(c)
```
c.
```
a = int(input())
if a > 0:
    print(a)
else:
    print(a)
```

32. Какая конструкция останавливает текущую итерацию цикла и переходит к следующей?
a. break
b. continue
c. return
d. range
e. random
f. Ни один из выше перечисленных

33. Выберите правильный вариант вызова функции.
a) def argument
b) (argument)

c) my_function
d) my_function def (argument)
e) def my_function (argument)
f) def my_function
g) my_function (argument) def
h) Ни один из выше перечисленных

34. Какой из перечисленных цветов соответствует (255, 0, 0)?
a) BLACK
b) WHITE
c) BLUE
d) GREEN
e) RED

35. Написать код для обновления экрана в Pygame.

Конспект занятия по Python

Тема: **«Оператор цикла с параметром for. Операторы управления циклом. Пример задачи с использованием цикла for. Решение различных задач с циклами.»**

Тип занятия: учебное занятие изучения и первичного закрепления новых знаний.

Форма работы: фронтальная, индивидуальная.

Цель занятия: изучить оператор цикла с параметром for и рассмотреть примера решения задач с использованием цикла for.

Оборудование: презентация.

Задачи занятия:

Образовательные:

- изучить оператор цикла с параметром for;
- рассмотреть примеры решения задач с циклом for;
- продолжать формировать основные навыки алгоритмизации и программирования;
- продолжать формировать навыки в решении задач по программированию.

Развивающие:

- развивать мыслительную деятельность обучающихся;
- развивать умение самостоятельно работать;
- развивать умение анализировать.

Воспитательные:

- воспитывать умение вести учебный диалог;
- совершенствовать коммуникативные навыки.

Ход занятия

1. **Организационный момент:** Проверить готовность учащихся к занятию. Приветствие. Сообщение темы занятия.

2. **Актуализация занятия (опрос по пройденному материалу):**

Вопросы:

1. Цикл while – это цикл, который:
   a. Повторяется определенное число раз
   b. Повторяется до тех пор, пока условие верное
   c. Повторяется до тех пор, пока условие ложное

Ответ: b.

2. Выберите правильную запись цикла while:
   a. while:
      условие
      инструкции
   b. while условие
      инструкции
   c. while инструкции
      условие
   d. while условие:
      инструкции

Ответ: d.

3. Сколько раз будет выполнен этот цикл?

```
i = 3
while i < 7:
   print ( "Привет!" )
   i += 1
```

Ответ: 4.

4. Чему будет равно значение переменной «a» после выполнения этого цикла?

```
i = 4
a = 12
```

```
while i < 5:
  a += i
  i += 1
```

Ответ: 16.

5. Какие клавиши необходимо нажать, чтобы прекратить выполнения бесконечного цикла?

a. Ctrl + C
b. Ctrl + Z
c. Ctrl + W
d. Ctrl + V

Ответ: a.

3. **Усвоение новых знаний и способов действий. Изучение нового материала.**

Цикл for используется в тех случаях, когда вам нужно повторить что-нибудь n-ное количество раз.

В Python цикл начинается с ключевого слова for, за которым следует произвольное имя переменной, которое будет хранить значения следующего объекта последовательности.

Общий синтаксис for...in в python выглядит следующим образом:

for <переменная> in <последовательность>:
  <действие>
else:
  <действие>

Элементы “последовательности” перебираются один за другим “переменной” цикла; переменная указывает на элементы. Для каждого элемента выполняется “действие”.

Функция range

range() позволяет генерировать ряд чисел в рамках заданного диапазона. В зависимости от того, как много аргументов вы передаете функции, вы можете решить, где этот ряд чисел начнется и закончится, а также насколько велика разница будет между двумя числами.

Есть три способа вызова range():

- range(стоп)

Вызывая range() с одним аргументом, вы получите ряд чисел, начинающихся с 0 и включающих каждое число до, но не включая число, которое вы обозначили как конечное (стоп).

```
for i in range(3):
  print(i)
```

- range(старт, стоп)

Вызывая range() с двумя аргументами, вам нужно решить не только, где ряд чисел должен остановиться, но и где он должен начаться, так что вам не придется начинать с нуля каждый раз.

```
for i in range(1, 8):
  print(i)
```

- range(старт, стоп, шаг)

Вызывая range() с тремя аргументами, вы можете выбрать не только то, где ряд чисел начнется и остановится, но также то, на сколько велика будет разница между одним числом и следующим. Если вы не задаете этот «шаг», то range() автоматически будет вести себя так, как если бы шаг был бы равен 1.

Обратите внимание: шаг может быть положительным, или отрицательным числом, но он не может равняться нулю.

Пример с положительным шагом:

```
for i in range(3, 100, 25):
    print(i)
```

Пример с отрицательным шагом:

```
for i in range(10, -6, -2):
    print(i)
```

Списки

Списки (list) используются для хранения элементов различных типов.

```
l = [“C”, “C++”, “Perl”, “Python”]
```

Элемент списка может быть вызван с помощью индекса, указанного в квадратных скобках. Первый элемент списка имеет индекс 0, а не 1.

```
l = [“C”, “C++”, “Perl”, “Python”]
print(l[0])
print(l[1])
print(l[2])
```

Пример простого цикла for в Python со списком:

```
l = ["C", "C++", "Perl", "Python"]
for x in l:
    print(x)
```

Блок else работает точно так же, как и в цикле while. Он будет выполнен только в том случае, если цикл не был «остановлен» оператором break. Таким образом, он будет выполнен только после того, как все элементы последовательности будут пройдены.

```
for i in range(3):
    print(i)
else:
    print(‘Цикл завершён’)
```

Оператор прерывания в python — break

Если в программе цикл for должен быть прерван оператором break, цикл будет завершен, и поток программы будет продолжен без выполнения действий из else.

Обычно фразы break в pyton связаны с условными операторами.

```
for i in range(5):
    if i == 2:
        break
    print(i)
```

Оператор пропуска python — continue

Предположим, нам нужно просто пропустить какое-то число и продолжить вывод чисел. Тогда нужно использовать оператор continue, для перехода к следующему элементу.

```
for i in range(5):
    if i == 2:
        continue
    print(i)
```

4. **Физкультминутка. Перерыв.**

На прошлом занятии мы познакомились с оператором цикла с параметром for. Сейчас рассмотрим примеры решения задач с циклом for.

5. **Первичная проверка понимания изученного. Решение задач.**

Ребят, перед тем как начать работать за компьютерами, вспомним технику безопасности при работе с ПК. Запрещается:

- трогать питающие провода и разъемы соединительных кабелей;
- прикасаться к экрану и тыльной стороне монитора;
- размещать на рабочем месте посторонние предметы;
- самостоятельно устранять неисправности в работе аппаратуры;
- при неполадках и сбоях в работе компьютера немедленно прекратите работу и сообщите об этом педагогу;
- работайте на клавиатуре чистыми, сухими руками;

- легко нажимайте на клавиши, не допуская резких ударов и не задерживая клавиши в зажатом состоянии.

Пример 1.

Вывести все нечетные числа от 0 до 10 с помощью цикла for.

```
for i in range(10):
    if i%2 == 0:
        continue
    print(i)
```

Пример 2.

Посчитать сумму чисел от 1 до 500.

```
summa = 0
for i in range(1,501):
    summa = summa + i
print(summa)
```

Пример 3.

Посчитать сумму всех нечётных чисел в списке.

```
a = [1,2,3,4,5,6]
summa = 0
for i in a:
    if i%2 != 0:
        summa = summa + i
print(summa)
```

Пример 4.

Даны два целых числа A и B. Выведите все числа от A до B включительно, в порядке возрастания, если A < B, или в порядке убывания в противном случае.

```
a = int(input())
b = int(input())
if a < b:
    for i in range(a, b + 1):
        print(i)
else:
    for i in range(a, b - 1, -1):
        print(i)
```

Задание на закрепление.

Найти сумму четных чисел от 1 до n (n вводит пользователь).

6. **Итоги занятия, рефлексия.**

Вывод:

- Скажите, какая тема занятия у нас была сегодня? (Оператор цикла с параметром for. Операторы управления циклом. Пример задачи с использованием цикла for. Решение различных задач с циклами.)

- Какова была цель занятия? (Изучить оператор цикла с параметром for и рассмотреть примера решения задач с использованием цикла for.)

Достигли цели?

- Что теперь вы знаете?

- Что теперь вы умеете?

Сегодня мы изучили оператор цикла с параметром for и рассмотрели примеры решения задач с использованием циклом for.

-Спасибо за занятие! На сегодня занятие окончено!

Конспект занятия по Python

Тема: **«Работа с различными событиями»**

Тип занятия: учебное занятие комплексного применения знаний и способов деятельности.

Форма работы: фронтальная, индивидуальная.

Цель занятия: рассмотреть примеры с использованием событий.

Задачи занятия:

Образовательные:

- рассмотреть примеры с использованием событий;
- продолжать формировать основные навыки алгоритмизации и программирования;
- продолжать формировать навыки в решении задач по программированию.

Развивающие:

- развивать мыслительную деятельность обучающихся;
- развивать умение самостоятельно работать;
- развивать умение анализировать.

Воспитательные:

- воспитывать умение вести учебный диалог;
- совершенствовать коммуникативные навыки.

Ход занятия

1. Организационный момент: Проверить готовность учащихся к занятию. Приветствие. Сообщение темы занятия.

2. Актуализация занятия.

Вспомним, что такое событие и какие они бывают, его способ записи и использования.

Ребят, перед тем как начать работать за компьютерами, вспомним технику безопасности при работе с ПК. Запрещается:

- трогать питающие провода и разъемы соединительных кабелей;
- прикасаться к экрану и тыльной стороне монитора; - размещать на рабочем месте посторонние предметы;
- самостоятельно устранять неисправности в работе аппаратуры;
- при неполадках и сбоях в работе компьютера немедленно прекратите работу и сообщите об этом педагогу;
- работайте на клавиатуре чистыми, сухими руками;
- легко нажимайте на клавиши, не допуская резких ударов и не задерживая клавиши в зажатом состоянии.

Напишите фрагмент кода, в котором при нажатии правой кнопки мыши менялась надпись заголовка окна, а при нажатии клавиши Enter окно закрывалось.

3. Обобщение и систематизация знаний. Решение задач.

Рассмотрим подробнее, какие бывают события.

- **Return** - Enter
- **Escape** - Esc
- **Control** - Ctrl
- **Alt**
- **Shift**
- **Lock**
- **Extended**
- **Prior** - PgUp
- **Next** - PgDown

- **Button1, B1** - нажата первая (левая) кнопка мыши
- **Button2, B2** - вторая (средняя) кнопка мыши
- **Button3, B3** - третья (правая)
- **Button4, B4** - четвёртая
- **Button5, B5** – пятая
- **Double** - двойной щелчок мыши (например, <Double-Button-1>)
- **Triple** - тройной
- **MouseWheel** - прокрутка колесом мыши
- **KeyPress, KeyRelease** - нажатие и отпускание клавиши на клавиатуре
- **ButtonPress, ButtonRelease, Motion** - нажатие, отпускание клавиши мыши, движение мышью
- **Configure** - изменение положения или размера окна
- **Map, Unmap** - показывание или сокрытие окна (например, в случае сворачивания/разворачивания окна пользователем)
- **Expose** - событие генерируется, когда необходимо всё окно или его часть перерисовать
- **Destroy** - закрытие окна
- **FocusIn, FocusOut** - получение или лишение фокуса
- **Enter, Leave** - Enter генерируется, когда курсор мыши "входит" в окно, Leave - когда "уходит" из окна

Решение задач

1. Напишите фрагмент кода, в котором при нажатии кнопки А изменялся цвет и размер Label.

2. Напишите часть кода, в которой при наведении на текстовую область Entry будет появляться текст.

3. Напишите программу, в которой при закрытии окна появлялось диалоговое окно.

4. Напишите программу, где клик левой кнопкой мыши по метке устанавливает для нее один шрифт, а клик правой кнопкой мыши – другой.

4. Итоги занятия, рефлексия.

Вывод:

- Скажите, какая тема занятия у нас была сегодня? (События.)
- Какова была цель занятия? (рассмотреть примеры с использованием событий.)

Достигли цели?

- Что теперь вы знаете?
- Что теперь вы умеете?

Сегодня мы изучили программирование событий в модуле Tkinter и рассмотрели примеры с их использованием.

-Спасибо за занятие! На сегодня занятие окончено!

Конспект занятия по Python

Тема: **«События»**

Тип занятия: учебное занятие изучения и первичного закрепления новых знаний.

Форма работы: фронтальная, индивидуальная.

Цель занятия: изучить программирование событий в модуле Tkinter и рассмотреть примеры с их использованием.

Оборудование: презентация.

Задачи занятия:

Образовательные:

- изучить программирование событий в модуле Tkinter;
- рассмотреть примеры с использованием событий;
- продолжать формировать основные навыки алгоритмизации и программирования;
- продолжать формировать навыки в решении задач по программированию.

Развивающие:

- развивать мыслительную деятельность обучающихся;
- развивать умение самостоятельно работать;
- развивать умение анализировать.

Воспитательные:

- воспитывать умение вести учебный диалог;
- совершенствовать коммуникативные навыки.

Ход занятия

1. Организационный момент: Проверить готовность учащихся к занятию. Приветствие. Сообщение темы занятия.

2. Актуализация занятия.

Вспомним, как отобразить главное окно и добавить некоторые графические объекты.

Ребят, перед тем как начать работать за компьютерами, вспомним технику безопасности при работе с ПК. Запрещается:

- трогать питающие провода и разъемы соединительных кабелей;
- прикасаться к экрану и тыльной стороне монитора; - размещать на рабочем месте посторонние предметы;
- самостоятельно устранять неисправности в работе аппаратуры;
- при неполадках и сбоях в работе компьютера немедленно прекратите работу и сообщите об этом педагогу;
- работайте на клавиатуре чистыми, сухими руками;
- легко нажимайте на клавиши, не допуская резких ударов и не задерживая клавиши в зажатом состоянии.

Напишите фрагмент кода для отображения главного окна, на котором должен располагаться вопрос, под ним варианты ответов (checkbutton или radiobutton) и кнопка.

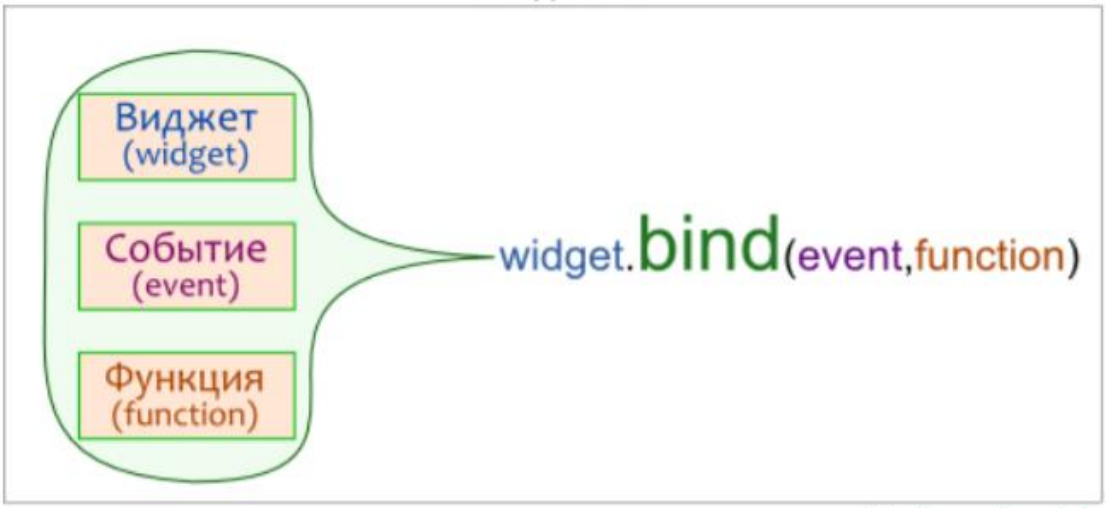
3. Усвоение новых знаний и способов действий. Изучение нового материала.

Приложения с графическим интерфейсом пользователя (GUI) должны не просто красиво отображаться на экране, но и выполнять какие-либо действия, реализуя тем самым потребности пользователя.

В отличие от консольных приложений, которые обычно выполняются при минимальных внешних воздействиях, графическое приложение обычно ждет каких-либо внешних воздействий (щелчков кнопкой мыши, нажатий клавиш на клавиатуре, изменения виджетов) и затем выполняет заложенное программистом действие. Из такого принципа работы можно вывести следующую схему настройки функциональности GUI: на виджет что-то «влияет» из вне → выполняется какая-то функция (действие). Внешнее воздействие на графический компонент называется ***событием***.

Одним из способов связывания виджета, события и функции (того, что должно происходить после события) является использование метода bind. Синтаксис показан на рисунке:

Добавление функциональности графическому элементу с помощью метода bind



Типы событий

Можно выделить три основных типа событий: производимые мышью (щелчки кнопками мыши, ввод и вывод курсора за пределы виджета, движение мыши), нажатиями клавиш на клавиатуре (нажатие клавиши, нажатие комбинаций клавиш, отжатие клавиш), а также события, возникающие в результате изменения других графических объектов (ввод данных, изменение размеров окна, прокрутка и др.).

Способ записи

При вызове метода bind событие передается в качестве первого аргумента.

Название события заключается в кавычки, а также в знаки < и >. Событие описывается с помощью зарезервированных последовательностей ключевых слов.

События, производимые мышью

- <Button-1> - щелчок левой кнопкой мыши
- <Button-2> - щелчок средней кнопкой мыши
- <Button-3> - щелчок правой кнопкой мыши
- <Double-Button-1> - двойной клик левой кнопкой мыши
- <Motion> - движение мыши
- и т. д.

Рассмотрим пример с использованием этих событий.

Пример.

```
from tkinter import *
def b1(event):
      root.title("Левая кнопка мыши")
def b3(event):
      root.title("Правая кнопка мыши")
def move(event):
      root.title("Движение мышью")
root = Tk()
root.geometry('400x400')
root.bind('<Button-1>',b1)
```

```
root.bind('<Button-3>',b3)
root.bind('<Motion>',move)
root.mainloop()
```

В этой программе меняется надпись в заголовке главного окна в зависимости от того двигается мышь, щелкают левой или правой кнопкой мыши.

4. Физкультминутка. Перерыв.

На прошлом занятии мы познакомились с событиями, производимыми мышью. Сейчас рассмотрим события, производимые с помощью клавиатуры.

События, производимые с помощью клавиатуры

- буквенные клавиши можно записывать без угловых скобок (например, 'L').
- Для неалфавитных клавиш существуют специальные зарезервированные слова (<Return> - нажатие клавиши Enter; <space> - пробел; и т. д.)
- Сочетания клавиш пишутся через тире.

Например: <Control-Shift> (одновременное нажатие клавиш Ctrl и Shift).

Рассмотрим пример с их использованием.

Пример.

```
from tkinter import *
def exit_(event):
    root.destroy()
def caption(event):
    t = ent.get()
    lbl.configure(text = t)
root = Tk()
ent = Entry(root, width = 40)
lbl = Label(root, width = 80)
ent.pack()
lbl.pack()
ent.bind('<Return>',caption)
root.bind('<space>',exit_)
root.mainloop()
```

При нажатии клавиши Enter в пределах текстовой строки (ent) вызывается функция caption, которая помещает символы из текстовой строки (ent) в метку (lbl). Нажатие пробела приводит к закрытию главного окна.

5. Первичная проверка понимания изученного. Решение задач.

1. Сделайте так, чтобы при нажатии на кнопку правой кнопкой мыши изменялся цвет кнопки и появлялась надпись на ней.

2. Напишите фрагмент кода, в котором при нажатии кнопки A изменялся цвет и размер Label.

3. Напишите часть кода, в которой при наведении на текстовую область Entry будет появляться текст.

6. Итоги занятия, рефлексия.

Вывод:

- Скажите, какая тема занятия у нас была сегодня? (События.)

- Какова была цель занятия? (изучить программирование событий в модуле Tkinter и рассмотреть примеры с их использованием.)

Достигли цели?

- Что теперь вы знаете?

- Что теперь вы умеете?

Сегодня мы изучили программирование событий в модуле Tkinter и рассмотрели примеры с их использованием.

-Спасибо за занятие! На сегодня занятие окончено!

Промежуточная аттестация

* Обязательно

1. ФИО *

2. Строка - это *

Отметьте только один овал.

- ◯ последовательность символов в кодировке Unicode, заключенных в кавычки.
- ◯ изменяемая последовательность значений любого типа.
- ◯ неизменяемая последовательность значений любого типа.
- ◯ структура, которая хранит данные в формате пар ключ-значение.
- ◯ неупорядоченная коллекция уникальных элементов.

3. Список - это *

Отметьте только один овал.

- ◯ изменяемая последовательность значений любого типа.
- ◯ последовательность символов в кодировке Unicode, заключенных в кавычки.
- ◯ неизменяемая последовательность значений любого типа.
- ◯ структура, которая хранит данные в формате пар ключ-значение.
- ◯ неупорядоченная коллекция уникальных элементов.

4. Кортеж - это *

Отметьте только один овал.

- последовательность символов в кодировке Unicode, заключенных в кавычки.
- изменяемая последовательность значений любого типа.
- неизменяемая последовательность значений любого типа.
- структура, которая хранит данные в формате пар ключ-значение.
- неупорядоченная коллекция уникальных элементов.

5. Словарь - это *

Отметьте только один овал.

- последовательность символов в кодировке Unicode, заключенных в кавычки.
- изменяемая последовательность значений любого типа.
- неизменяемая последовательность значений любого типа.
- структура, которая хранит данные в формате пар ключ-значение.
- неупорядоченная коллекция уникальных элементов.

6. Множество - это *

Отметьте только один овал.

- неупорядоченная коллекция уникальных элементов.
- структура, которая хранит данные в формате пар ключ-значение.
- неизменяемая последовательность значений любого типа.
- изменяемая последовательность значений любого типа.
- последовательность символов в кодировке Unicode, заключенных в кавычки.

7. Что выведет данный фрагмент кода? *

Отметьте только один овал.

- 1+2
- 3
- 12
- Ошибку

8. Что выведет данный код? *

Отметьте только один овал.

- spa
- aps
- m
- spm
- Ошибку

9. Что будет выведено в результате выполнения данного кода? *

```
lst = list("spam")
st = ""

for x in lst:
    st += x

print(st)
```

Отметьте только один овал.

- spa
- spam
- Ошибку
- Ничего

10. Что выведет данный код? *

```
st = "spam "

print(len(st))
```

Отметьте только один овал.

- 4
- 3
- 5
- Ошибку

11. Что выведет данный код? *

```
st = "1" - "2"

print(st)
```

Отметьте только один овал.

- 1-2
- 12
- -1
- Ошибку

12. Что выведет данный код? *

```
st = spam

print(st)
```

Отметьте только один овал.

- spam
- Ошибку
- Ничего

13. Что выведет данный код? *

Отметьте только один овал.

- masp
- spam
- maps
- Ошибку

14. Что выведет данный код? *

Отметьте только один овал.

- 1212
- 2
- 11
- Ошибку
- 1
- 12

15. Что выведет данный код? *

```
st = "spam"

for x in st:
    print(x, end="")
```

Отметьте только один овал.

- spa
- spam
- s p a m
- Ошибку
- maps
- m a p s

16. Ниже представлены утверждения о списках. Какие из них верны? *

Отметьте все подходящие варианты.

- [ ] Один и тот же объект может появляться в списке несколько раз
- [ ] Размеры списка четко не определены
- [ ] Эти два списка одинаковы: ['a', 'b', 'c'] ['c', 'a', 'b']
- [ ] Все элементы в списке должны быть одного типа
- [ ] В списке может содержаться любой тип данных, кроме других списков

17. Объявлен список — a = [1, 2, 3, 4, 5].Ниже представлены строки кода, удаляющие элемент. Какие из них в результате дадут список [1, 2, 4, 5]? *

Отметьте все подходящие варианты.

- [ ] del a[2]
- [ ] a[2:2] = []
- [ ] a[2:3] = []
- [ ] a.remove(3)
- [ ] a[2] = []

18. Объявлен список — a = ['a', 'b', 'c'].Ниже представлены строки кода. Какие из них корректно добавляют элементы ‘d’ и ‘e’ в конец списка? *

Отметьте все подходящие варианты.

- [ ] a += 'de'
- [ ] a.append(['d', 'e'])
- [ ] a[len(a):] = ['d', 'e']
- [ ] a.extend(['d', 'e'])
- [ ] a += ['d', 'e']
- [ ] a[-1:] = ['d', 'e']

19. Объявлен кортеж — t = ('foo', 'bar', 'baz').Ниже представлены варианты изменения ‘bar’ на ‘qux’. Какой вариант верный? *

Отметьте только один овал.

- ◯ Кортежи неизменяемы
- ◯ t[1:1] = 'qux'
- ◯ t(1) = 'qux'
- ◯ t[1] = 'qux'

20. Ниже представлены варианты объявления кортежа с одним элементом — ‘foo’. Какой из них правильный? *

Отметьте только один овал.

- ◯ t = ['foo']
- ◯ t = ('foo',)
- ◯ t = ('foo')
- ◯ t = {'foo'}

21. Есть строка кода — a, b, c = (1, 2, 3, 4, 5, 6, 7, 8, 9)[1::3]. Чему равно b? *

Отметьте только один овал.

- 2
- 6
- 4
- 5

22. С помощью какого метода можно развернуть список? *

Отметьте только один овал.

- back()
- return()
- reversed()
- reverse()

23. К чему приведет обращение к непустому списку по индексу -1? *

Отметьте только один овал.

- вернется последний элемент
- будет ошибка IndexError
- вернется первый элемент
- ошибка KeyError

24. Что будет выведено на экран? *

```
L = list('abc')
print(L.pop(1))
```

Отметьте только один овал.

- a
- b
- ['a', 'b', 'c', 'b']
- ['a', 'c']
- None

25. Какой метод добавляет элемент в конец списка? *

Отметьте только один овал.

- insert()
- extend()
- pop()
- append()

26. Выберете верное утверждение. *

Отметьте только один овал.

- Можно добавить элемент в кортеж
- Все не верно
- Можно удалить элемент из кортежа
- Можно удалить элемент, но нельзя добавить новый

27. Когда точно нужно использовать кортеж? *

Отметьте только один овал.

- ◯ До нас данные записывали в кортеж
- ◯ Мы записываем данные, которые будут обновляться
- ◯ Мы записываем данные, которые нельзя менять
- ◯ Мы записываем данные, которые меняются раз в год

28. Что выведет этот код? *

```
sample = (10, 20, 30)
sample.append(60)
print(sample)
```

Отметьте только один овал.

- ◯ Ошибку
- ◯ (10, 20, 30)
- ◯ (10, 20, 30, 60)
- ◯ [10, 20, 30, 60]

29. Какая функция создаст пустой кортеж? *

Отметьте только один овал.

- ◯ taple()
- ◯ typle()
- ◯ tupl()
- ◯ Такой функции в python нет
- ◯ tuple()

30. Как получить последний элемент этого кортежа: lake = ("Python", 51, False, "22") *

Отметьте только один овал.

- Все варианты подходят
- lake[lake.index("22")]
- lake[-1]
- lake[3]

31. Выберите верные утверждения: *

Отметьте все подходящие варианты.

- [ ] Словари изменяемы
- [ ] Словарь может содержать объект любого типа, кроме другого словаря
- [ ] Доступ к элементам словаря осуществляется с помощью ключа
- [ ] Доступ к элементам словаря осуществляется с помощью позиции в словаре
- [ ] Все ключи в словаре должны быть одного и того же типа

32. Ниже представлены несколько вариантов кода. Какой из них удалит элемент с ключом ‘baz’ из словаря? *

Отметьте только один овал.

- del d['baz']
- d[‘baz’]
- del.d(baz)
- del d(baz)

33. Объявлен словарь — d = {'foo': 100, 'bar': 200, 'baz': 300}.Какой результат будет у d['bar':'baz']? *

Отметьте только один овал.

- ◯ Возникнет ошибка
- ◯ (200, 300)
- ◯ [200, 300]
- ◯ 200 300

34. Ниже представлен список ключей. Какие из них синтаксически правильны? *

Отметьте все подходящие варианты.

- ☐ ('foo', 'bar')
- ☐ (3+2j)
- ☐ 'foo'
- ☐ ['foo', 'bar']
- ☐ dict(foo=1, bar=2)

35. Объявлен словарь — d = {'foo': 100, 'bar': 200, 'baz': 300}. Какой метод удалит элемент со значением 200? *

Отметьте только один овал.

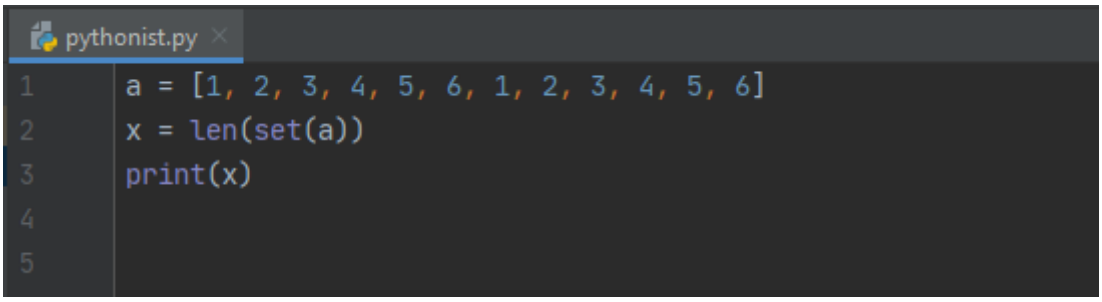
- ◯ del d(200)
- ◯ del d("bar")
- ◯ d.pop(200)
- ◯ d.pop("bar")

36. Каким образом правильно объявляется множество? *

Отметьте только один овал.

- a = {}
- a = []
- a = set()
- a = set

37. Что выведет этот код? *



Отметьте только один овал.

- 6
- 5
- 12
- 0

38. Чем отличаются методы remove() и discard(), применяемые к множеству? *

Отметьте только один овал.

- Ничем
- remove() удаляет элемент если он есть, но бросает ошибку если элемента нет. discard() просто удаляет элемент если он есть
- discard() удаляет элемент если он есть, но бросает ошибку если элемента нет. remove() просто удаляет элемент если он есть
- Метода discard() для множеств не существует

39. Какой метод используется для добавления элемента в множество? *

Отметьте только один овал.

- add()
- append()
- new()
- get()

40. Элемент под каким индексом выбросится из множества при использовании метода pop()? *

Отметьте только один овал.

- -1, то есть последний
- 0, первый элемент
- Никакой, потому что множество – неупорядоченная последовательность
- Неизвестно, выбросится случайный элемент

41. Каким методом можно очистить множество? *

Отметьте только один овал.

- clear()
- delete()
- empty()
- zero()

42. Напишите сравнение двух списков. *

43. Пользователь вводит строку, содержащую несколько слов, между которыми один или несколько пробелов. Требуется найти количество символов в самом длинном слове. *

Тест "Функции"

Всего 20 вопросов

* Обязательно

1. ФИО *

2. С какого ключевого слова начинается блок функции? *

Отметьте только один овал.

- dir
- oct
- len
- set
- range
- def
- dict
- for
- Ни один из выше перечисленных

3. Выберите правильный вариант вызова функции. *

Отметьте только один овал.

- def argument
- (argument)
- my_function
- my_function def (argument)
- def my_function (argument)
- def my_function
- my_function (argument) def
- Ни один из выше перечисленных

4. Какие из данных аргументов функции существуют? *

Отметьте все подходящие варианты.

- [ ] Обязательные аргументы
- [ ] Аргументы определенной длины
- [ ] Позиционированные аргументы
- [ ] Аргументы-ключевые слова
- [ ] Ни один из выше перечисленных

5. Рекурсивная функция - ... *

Отметьте только один овал.

- ◯ это функция, которая вызывает саму себя.
- ◯ это функция, которая вызывает в себе другую функцию.
- ◯ это функция, которая может вызываться определенное количество раз.
- ◯ это функция, которая может быть вызвана множество раз.
- ◯ это функция, которая принимает ни одного аргумента.
- ◯ Ни один из выше перечисленных

6. Сколько параметров может принимать функция? *

Отметьте только один овал.

- ◯ Нисколько, функция не принимает значения, только возвращает
- ◯ 1
- ◯ 2
- ◯ Бесконечно много
- ◯ Ни один из выше перечисленных

7. Какое ключевое слово используется для возврата значения из функции? *

Отметьте только один овал.

- get
- post
- return
- answer
- Ни один из выше перечисленных

8. Что выведет данный код? *

```
def get_sum(a,b):
    pass

print(get_sum(4,2))
```

Отметьте только один овал.

- 4
- 2
- 6
- None
- Ни один из выше перечисленных

9. Что выведет данный код? *

```
def get_sum(a=2,b=3):
    print(a+b)

get_sum(4)
```

Отметьте только один овал.

- 5
- 4
- 7
- 6
- Ни один из выше перечисленных

10. Что выведет данный код? *

```
def get_sum(a,b):
    print(a+b)

get_sum(4)
```

Отметьте только один овал.

- 4
- 1
- Ошибку
- 0
- Ни один из выше перечисленных

11. Выберите правильный синтаксис функции. *

Отметьте только один овал.

- def my_function argument
- def my_function argument:
- def my_function (argument):
- def my_function (argument)
- Ни один из выше перечисленных

12. Сколько аргументов у функции? *

randrange(0,100)

Отметьте только один овал.

- 0
- 100
- 2
- 3
- Бесконечно
- 4
- Ни один из выше перечисленных

13. Какое будет самое большое число, которое эта функция выведет на экран? *

```
def print_numbers():
    print(1)
    print(6)
    return
    print(10)
    print(9)
```

Отметьте только один овал.

- 4
- 0
- 1
- 6
- 2
- 10
- 9
- 7
- Ни один из выше перечисленных

14. Написать функцию нахождения наибольшего из 3-х чисел. *

15. Опишите 3 встроенные функции в Python. *

16. Напишите три функции, которые вычисляют сумму чисел от 1 до N (N вводит пользователь): с циклом for, с циклом while, с рекурсией. *