Моя IT Школа

сертифицированные IT курсы

Цикл for

Проверка домашнего задания

Моя IT Школа

Казино. Компьютер генерирует числа от 1 до 10 и от 1 до 2-х соответственно. Цифры от одного до 10 отвечают за номера, а от 1 до 2 за цвета (1 красный, 2 черный).
Пользователю дается 5 попыток угадать номер и цвет (Прим. введения с клавиатуры: 3 красный). В случае неудачи вывести на экран правильную комбинацию.

```
color_ = ["красный", "черный"]  # Задаю список с цветами
numbers = random.randint(1, 10)  # Задаю переменную в которой генерируются случайные числа от 1 до 10
comp_color = random.choice(color_)  # Этим методом добавляем в переменную случайный элемент из последовательности
n = 0  # Счетчик
while n < 5:  # Пока счетчик будет меньше 5
    num_user = int(input("Ведите число от 1 до 10: "))  # Вводим число от 1 до 10
    col_user = input("Введите цвет: (красный) или (черный):  ")  # Вводим число цвета 1 или 2
    n += 1  # к каждой итерации добавляем 1 чтобы цикл имел концовку
    if 0 < num_user <= 10 and col_user == "красный" or col_user == "черный":  # проверяем правильность введенных данных
        if num_user == numbers and col_user == comp_color:  # Если число с клавиатуры = случайному,
            # и цвет введенный с клавиатуры = случайному цвету
            print("Поздравляем, вы выиграли")  # Выводим на экран результат
            break  # Заканчиваем цикл
        elif n == 5:  # Если счетчик равен 5
            print(f"К сожалению вы проиграли!" "\n" "Правильное число и цвет: ", numbers,
                  comp_color)  # Выводим на экран результат
    else:  # Если условия не соблюдены
        print("Вы ввели некорректные данные, попробуйте еще раз.")  # Выводим на экран результат
```

План занятия

Моя IT Школа
сертифицированные IT курсы

1 Циклы

2 Цикл for

3 Модуль array

Списки в Python

Цикл for

Моя IT Школа

Каждый язык программирования, содержит какую-нибудь конструкцию цикла. В большей части языков есть больше одной такой конструкции. В мире Python есть два типа циклов:

- Цикл **for**
- Цикл **while**

Циклы используются в тех случаях, когда нам нужно сделать что-нибудь много раз. Нередко вам придется выполнить какую-нибудь операцию (или ряд операций) в части данных снова и снова.
Тут то и вступают в силу циклы. Благодаря им становится возможно максимально упростить данный вопрос.

Цикл for

Моя IT Школа

Цикл for, также называемый циклом с параметром, в языке Питон богат возможностями. В цикле for указывается переменная и множество значений, по которому будет пробегать переменная.

Для повторения цикла некоторое заданное число раз і используют цикл for вместе с функцией range:

```
for i in range(4):
    print(i)
```

Цикл for

Моя Школа

Так же в range можно передать два или три параметра:

range (4) - формирует диапазон от 0 до 4, не включая 4 = (0,1,2,3), а переменная i самостоятельно перебирает этот диапазон и с помощью команды print(i), данная переменная выводится на экран

range (4, 8) - формирует диапазон от 4 до 8, не включая 8 = (4,5,6,7)
range (4, 8, 1) - 1 - это шаг, шаг может быть и отрицательным.

Например:
range (4, 8, 1) = (4,5,6,7)
range (1, 9, 3) = (1,4,7)
range (10, 5, -2) = (10, 8, 6)

В чем основное отличие циклов for и while?

Несмотря на то, что и **for и while** необходимы для повторения некоего количества раз одной и той же операции , циклы отличаются друг от друга и имеют свою специфику. Даже с учетом их формальной взаимозаменяемости.

While удобен тогда, когда повторяющаяся операция проводится до тех пор, пока условие верно, т.е. возвращает **True**. Отсюда возможна ситуация, когда цикл не сработает ни разу либо будет повторяться бесконечно. Чтобы была возможность применить данный вид цикла для объекта, тот обязан иметь атрибут **bool()**.

Цикл **for** применяется для последовательного манипулирования с элементами итератора. Другими словами, он проходит по очереди элементы объекта (например, списка) и заканчивается (в общем случае) после их полного перебора.

Таким образом, **for** удобен для перебора, а **while** – проверки истинности условия перед каждой итерацией.

Разберем следующую программу:

```
for i in "Я учу Python":
    print(i)
```

Цикл for

Моя IT Школа

Переменная і пробегает по строке «Я учу Python» и на каждой итерации цикла, переменной і присваивается следующий символ строки. Поэтому на экран и вывелась наша строка, но посимвольно.

Цикл for

Моя IT Школа

Разберем задачу: Необходимо вывести числа от 1 до 15 в порядке убывания.

```
for i in range(15, 0, -1):
    print(i)
```

Цикл for

В range мы передаем три параметра 15, 0, -1. range формирует диапазон от 15 до 0 с шагом -1, не включая 0 = (15, 14.....1) и далее мы просто выводим на экран нашу переменную i, которая каждую новую итерацию берет следующее число из нашей последовательности.

Задание №1

Пользователь вводит строку и один символ. Программа должна вывести на экран строку без этого символа.

Например: «Я учу программирование» символ «о»
Результат «Я учу программирование».

Примечание: напоминаю, что строки можно складывать
«Я учу » + «программирование» = «Я учу программирование»

Решение и результат

Моя Школа

```
a = input('Введите строку: ')
b = input('Введите символ: ')
c = ''
for i in a:
    if i != b:
        c += i
print('Результат: ', c)
```

```
s1 = input('some string: ')
symbol = input('One symbol: ')
new_string = ''
for i in s1:
    if i == symbol:
        continue
    new_string += i
print(new_string)
```

Задание №2

Вывести все трехзначные числа, которые делятся на 100 без остатка.

Решение и результат

Моя IT Школа

```
for i in range(100, 1001, 100):
    print(i)
```

Задание №3

Вывести на экран все числа в диапазоне от 94 до 350 кратные 5.

Решение и результат

Моя Школа

```
for i in range(94, 351):
    if i % 5 == 0:
        print(i)
```

Списки и массивы в Python

Моя Школа

В python есть такие типы данных, как **список и массив**.

Массивы могут быть одномерными и многомерными.
from array import *

```
arr = array('i', [2, 5, 4, 0, 8])
print(type(arr)) #class 'array.array'>
print(arr) #array('i', [2, 5, 4, 0, 8])
```

Любой **массив** может содержать данные только одного типа, то есть нельзя использовать *int и float* в одном массиве. Это является недостатком при работе. Для устранения этого были введены списки.

Массив и список в Python похожи, вместе с тем список-это одномерный массив. Различие в том, что в списках можно хранить объекты различных типов данных. Размер списка не статичен и поддается изменениям.

Списки в Python

Моя IT Школа

Списки в Python представляют собой набор элементов. Значения указываются внутри квадратных скобок, где перечисляются через запятую. Как правило, любой элемент можно вызвать по индексу и присвоить ему новое значение.

Пустой список:
lst = []

Список строк в **Python:**

lst = ['string1', 'string2', 'string3']

Методы списков в Python

Моя IT Школа

list_name.append(x) - добавление элемента в конец списка.

list_name.count(x) - возвращает количество вхождений x в список.

list_name.index(x) - номер первого вхождения x в список.

list_name.pop (i) - удаляет i-ый элемент из списка и возвращает его. По умолчанию удаляется последний элемент.

list_name.remove(x) - удалить первое вхождение x из списка.

list_name.reverse(x) - обратный порядок элементов в списке.

Добавление элемента

Моя Школа

Добавление элемента в конец списка в Python.
Имя списка.append(значение)

```
a = [10, 2, 3]
print(a)
a.append(7)
print(a)
```

```
a = [10, 2, 3]
print(a)
a = a + [7]
print(a)
```

Цикл for

Моя Школа

Чтобы возвратить число элементов внутри массива, используют функцию **len():**

```
arr = ['string1', 'string2', 'string3']
l = len(arr)
print(arr, 'Длинна: ', l)
```

Цикл for

Моя Школа

Когда нужно перечислить элементы списка, применяют цикл **for.**
Записывается он в следующем виде:
[**for** переменная **in** список]

```
arr = [1, 7, 9, 10]

for i in arr:
    print(i)
```

```
1
7
9
10

Process finished with exit code 0
```

Оператор break

Моя IT Школа

В Python выражение **break** дает возможность выйти из цикла, когда его внешнее условие равно **True**. Выражение **break** помещается в блок кода внутри выражения, обычно после условного выражения **if**.

```
arr = [1, 7, 9, 10]

for i in arr:
    print(i)
    if i == 9:
        break
```

```
for number in range(10):
    if number == 5:
        break

    print('Number: ' + str(number))

print('Какой-то вне цикла for-in')
```

Оператор continue

Моя IT Школа

Выражение **continue** дает возможность пропустить часть цикла, где внешнее условие равняется True, те мы активируем данное условие, но при этом выполнить остальную часть цикла.
При этом прерывается текущая итерация цикла, а программа возвращается к началу цикла.

Выражение **continue** размещается в блоке кода под выражением цикла, обычно после условного выражения **if.**

```
for number in range(10):
    if number == 5:
        continue

    print('Number: ' + str(number))

print('Какой-то код вне цикла for-in')
```

```
arr = [1, 7, 9, 10]

for i in arr:
    if i == 9:
        continue
    print(i)
```

Оператор pass

Моя IT Школа

Также нам может потребоваться некоторое выражение, чтобы работать с условием, но не влиять на цикл.

Такое условие в Python - **pass**.

Когда внешнего условия выражение равно **True** и у нас выражение **pass**, чтение кода будет продолжаться до появления выражения **break** или другого выражения.

Как и в случае с другими выражениями, выражение **pass** будет содержаться обычно после условного выражения **if.**

```
for number in range(10):
    if number == 5:
        pass

    print('Number: ' + str(number))

print('Какой-то код вне цикла for-in')
```

Задание

Моя Школа

Что отобразится на экране и почему ?

```
for letter in "I'm python developer":
    if letter == 'P':
        break
else:
    print('Мы успешно прошли циклом for-in по строке')
```

Задание №4

Дан массив из 7 цифр. Если четных цифр в нем больше чем нечетных, то найти сумму всех цифр массива, если нечетных больше, то найти произведение 1, 3, 6 элементов

Решение и результат

Моя Школа

```
arr = [1,2,3,4,5,6,7]
even, odd = 0, 0
for number in arr:
    if number % 2 == 0:
        even += 1
    else:
        odd += 1
if even > odd:
    print(f'Сумму всех цифр массива arr: {math.sum(arr)}')
else:
    print(f'Произведение 1, 3, 6 элементов: {arr[0]*arr[2]*arr[5]}')
```

Задание №5

Дан массив чисел. Найти их сумму и произведение.

Решение и результат

Моя IT Школа

```
arr = [1, 2, 5, 6, 7, 8, 9]
sum = 0
pr = 1
for i in arr:
    sum += i
    pr *= i
print("Сумма", sum)
print("Произведение", pr)
```

Вложенные циклы

Моя IT Школа

Вложенный цикл - цикл который выполняется внутри другого цикла.

Обычно вложенные циклы используются для работы с двумя измерениями: рисование, таблица умножения и т.д.

При каждом обходе(итерации) внешнего цикла внутренний цикл будет выполнен полностью. Внутренний цикл должен завершить все свои итерации, прежде чем внешний цикл сможет перейти к следующей итерации.

```
1  for x in range(10, 510, 100): #10, 110, 210, 310, 410, 510
2      for y in range(10, 51, 10): #10, 20, 30, 40, 50
3          print(x+y)
4      print('Вы крутые, молодцы :)')
5  print('ой бой')
```

строки 1, 2, 3, 4 - внешний цикл

строки 2, 3 - вложенный цикл

строки 2, 3, 4 - тело внешнего цикла

строки 3 - тело вложенного цикла

Задание №6

Написать таблицу умножения от 1 до 9.

Решение и результат

Моя Школа

```
# Практическое 6


for i in range(1,10):
    for j in range(1,10):
        print(i*j, ' ', end=' ')
    print()
```

```
1   2   3   4   5   6   7   8   9
2   4   6   8   10   12   14   16   18
3   6   9   12   15   18   21   24   27
4   8   12   16   20   24   28   32   36
5   10   15   20   25   30   35   40   45
6   12   18   24   30   36   42   48   54
7   14   21   28   35   42   49   56   63
8   16   24   32   40   48   56   64   72
9   18   27   36   45   54   63   72   81

Process finished with exit code 0
```

Творческое задание

Придумайте задачу на пройденную тему и решите её

Домашнее задание

Моя IT Школа

1. Перемножить все нечётные значения в диапазоне от 1 до 30.

2. Записать в массив все числа в диапазоне от 1 до 100 кратные 5.

3. Вывести на экран все чётные значения в диапазоне от 1 до 71.

4. Дан массив чисел. Если число встречается более двух раз, то добавить его в новый массив.