Алексей Васильев

Язык программирования Python

Киев 2019

Лекция 2. Управляющие инструкции

- Условный оператор if
- Оператор цикла while
- Оператор цикла for
- Тернарный оператор
- Основы обработки ошибок

Условный оператор if

Синтаксис:

```
if условие:
    # команды
else:
    # команды
```

Блок **else** не является обязательным

Условие - выражение логического (тип **bool**). Возможные значения **True** (истина) и **False** (ложь)

```
if условие:
    # команды
elif условие:
    # команды
elif условие:
    # команды
elif условие:
    # команды
...
elif условие:
    # команды
else:
    # команды
```

Условный оператор

Программа (UsingIf.py):

```
# Считываем число:
num=int(input("Введите число: "))
# Условный оператор:
if num%2==0:
    print("Это четное число")
else:
    print("Это нечетное число")
# После условного оператора:
print("Спасибо за сотрудничество!")
```

Результат выполнения:

```
Введите число: 123
Это нечетное число
Спасибо за сотрудничество!
```

```
Введите число: 124
Это четное число
Спасибо за сотрудничество!
```

Условный оператор

При формировании условия можно:

- Использовать операторы сравнения
  == (равно), != (не равно), < (меньше),
  <= (меньше или равно), > (больше),
  >= (больше или равно)

- Использовать логические операторы
  and (логическое и), or (логическое или), not (отрицание)

- Использовать операторы принадлежности
  in (принадлежит), not in (не принадлежит)

Правила приведения к логическому типу:

- Ненулевые значения - True, нулевые значение - False
- Пустые последовательности - False, непустые - True
- Объекты - отвечает метод __bool__()

Условный оператор - 2

Программа (UsingIf_2.py):

```
# Считываем текст:
txt=input("Введите текст: ")
# Символ:
symb='ч'
# Условный оператор:
if symb in txt:
    print("В тексте есть буква",symb)
else:
    print("В тексте нет буквы",symb)
```

Результат выполнения:

```
Введите текст: Изучаем Python
В тексте есть буква ч
```

```
Введите текст: Всем привет
В тексте нет буквы ч
```

Условный оператор - 3

Программа (UsingIf_3.py):

```
# Вводятся параметры уравнения:
a,b=eval(input("Введите (через запятую) два числа: "))
# Если параметры некорректные:
if (type(a)!=int and type(a)!=float) or (type(b)!=int and type(b)!=float):
    print("Введены некорректные значения!")
    # Завершение выполнения программы:
    raise SystemExit(0)
# Если первый параметр ненулевой:
elif a!=0:
    txt="Решение x="+str(b/a)
# Если второй параметр ненулевой (при нулевом первом):
elif b!=0:
    txt="Решений нет!"
# Если оба параметра нулевые:
else:
    txt="Решение - любое число!"
# Вид уравнения:
print("Уравнение "+str(a)+"x="+str(b))
# Результат поиска корня:
print(txt)
print("Поиск решения завершен.")
```

Решение уравнения ax=b

Условный оператор - 3

Результат выполнения:

```
Введите (через запятую) два числа: 2.5,3.6
Уравнение 2.5x=3.6
Решение x=1.44
Поиск решения завершен.
```

```
Введите (через запятую) два числа: 2,3
Уравнение 2x=3
Решение x=1.5
Поиск решения завершен.
```

```
Введите (через запятую) два числа: 0,3.5
Уравнение 0x=3.5
Решений нет!
Поиск решения завершен.
```

```
Введите (через запятую) два числа: 0,0
Уравнение 0x=0
Решение - любое число!
Поиск решения завершен.
```

```
Введите (через запятую) два числа: "2",3
Введены некорректные значения!
```

Задание - 1

Напишите программу, которая определяет, делится ли введенное пользователем число на 7

Задание - 2

Напишите программу, которая определяет, делится ли введенное пользователем число хотя бы на одно из чисел 5 или 11

Задание - 3

Напишите программу, которая определяет, содержится ли цифра 3 во введенном пользователем числе. Для преобразование чисел в текст использовать функцию str()

Оператор цикла while

Синтаксис:

```
while условие:
    # команды
else:
    # команды
```

Блок **else** не является обязательным

```
while условие:
    # команды
```

Инструкция **break** завершает выполнение оператора цикла (в этом случае код в блоке **else** не выполняется)

Инструкция **continue** завершает выполнение текущей итерации

Сумма чисел

Программа (UsingWhile.py):

```
# Считывание верхней границы суммы:
n=int(input("Укажите верхнюю границу суммы: "))
# Начальное значение для суммы:
s=0
# Начальное значение индексной переменной:
k=0
# Оператор цикла для вычисления суммы:
while k<n:
    # Увеличение значения индексной переменной на единицу:
    k=k+1
    # Прибавление слагаемого к сумме:
    s=s+k
# Отображение результата:
print("Сумма чисел от 1 до",n,"равна",s)
```

Результат выполнения:

```
Укажите верхнюю границу суммы: 100
Сумма чисел от 1 до 100 равна 5050
```

Сумма чисел - 2

Программа (UsingWhile_2.py):

```
# Считывание количества слагаемых:
n=input("Укажите количество слагаемых: ")
# Переменная с текстовым значением:
txt="1"
# Индексная переменная:
k=1
# Оператор цикла для формирования текста с
# выражением для суммы натуральных чисел:
while str(k)!=n:
    # Увеличение значения индексной переменной:
    k=k+1
    # Добавление фрагмента к тексту с выражением
    # для суммы натуральных чисел:
    txt=txt+"+"+str(k)
# Отображение результата:
print(txt,"=",eval(txt))
```

Результат выполнения:

```
Укажите количество слагаемых: 10
1+2+3+4+5+6+7+8+9+10 = 55
```

Состав числа

Программа (UsingWhile_3.py):

```
# Вводится число:
number=int(input("Введите число: "))
# Пока число больше нуля:
while number>0:
    # Последняя цифра в числе:
    digit=number%10
    # Отображение цифры:
    print("|"+str(digit),end="")
    # Отбрасывается последняя цифра в числе:
    number=number//10
# Отображается последний разделитель:
print("|")
```

Результат выполнения:

```
Введите число: 31049265
|5|6|2|9|4|0|1|3|
```

Простые числа

Программа (UsingWhile_4.py):

```
# Вводится число:
number=int(input("Введите число: "))
# Верхняя граница для делителя:
num=number//2
# Начальное значение делителя:
k=2
# Поиск делителей числа:
while k<=num:
    # Если число делится на k:
    if number%k==0:
        print("Число не является простым")
        # Завершение оператора цикла:
        break
    # Увеличивается значение делителя:
    k=k+1
# Блок выполняется, если не выполнена инструкция break:
else:
    print("Это простое число")
# Сообщение отображается всегда:
print("Проверка завершена")
```

```
Введите число: 97
Это простое число
Проверка завершена
```

```
Введите число: 95
Число не является простым
Проверка завершена
```

Задание - 4

Напишите программу, которая вычисляет сумму квадратов нечетных натуральных чисел

Задание - 5

Напишите программу, которая вычисляет сумму цифр в числе, введенном пользователем

Задание - 6

Напишите программу, которая вычисляет все делители введенного пользователем числа

Оператор цикла for

Синтаксис:

```
for переменная in коллекция:
    # команды
else:
    # команды
```

Блок else не является обязательным

```
for переменная in коллекция:
    # команды
```

Инструкция break завершает выполнение оператора цикла (в этом случае код в блоке else не выполняется)

Инструкция continue завершает выполнение текущей итерации

Коллекция: список (например, [1,3,5]), текст (например, "Python"), кортеж (например, (1,3,5)), множество (например, {1,3,5}) и пр.

Сумма чисел

Программа (UsingFor.py):

```
print("Сумма натуральных чисел")
n=100 # Количество слагаемых
# Формируем текст для отображения результата:
text="1+2+...+"+str(n)+" ="
# Переменная для записи суммы:
s=0
# Оператор цикла для вычисления суммы:
for i in range(1,n+1):
    # Добавляем слагаемое к сумме:
    s=s+i
# Отображаем результат:
print(text,s)
```

Результат выполнения:

```
Сумма натуральных чисел
1+2+...+100 = 5050
```

Числа Фибоначчи

Программа (UsingFor_2.py):

```
# Количество чисел в последовательности:
n=15
# Первые два числа:
a,b=1,1
# Отображение первых двух чисел:
print(a,b,end=" ")
# За каждый цикл вычисляется одно новое число:
for k in range(n-2):
    # Вычисление нового числа в последовательности:
    a,b=b,a+b
    # Отображение нового числа:
    print(b,end=" ")
```

Результат выполнения:

```
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610
```

Поиск букв в тексте

Программа (UsingFor_3.py):

```
# Текст для поиска букв:
mytext=input("Введите текст для проверки: ")
# Буквы для поиска:
symbs=['а','у','я']
print("Ищем такие буквы:",symbs)
# Поиск букв:
for s in symbs:
    # Если буква найдена:
    if s in mytext:
        print("В тексте есть буква '"+s+"'")
        # Завершение оператора цикла:
        break
    # Если буквы нет:
    else:
        print("В тексте нет буквы '"+s+"'")
# Блок else оператора цикла:
else:
    print("Таких букв в тексте нет")
# Последнее сообщение программы:
print("Поиск завершен")
```

```
Введите текст для проверки: любой текст
Ищем такие буквы: ['а', 'у', 'я']
В тексте нет буквы 'а'
В тексте нет буквы 'у'
В тексте нет буквы 'я'
Таких букв в тексте нет
Поиск завершен
```

```
Введите текст для проверки: язык Python
Ищем такие буквы: ['а', 'у', 'я']
В тексте нет буквы 'а'
В тексте нет буквы 'у'
В тексте есть буква 'я'
Поиск завершен
```

Задание - 7

Напишите программу, которая для введенного числа определяет, сколько в этом числе цифр 0, 1, 2 и так далее, до 9

Задание - 8

Напишите программу, в которой пользователь вводит целое число, а программа каждую цифру в этом числе меняет на "дополнение до 9": цифра 0 меняется на 9, цифра 1 меняется на 8, цифра 2 меняется на 7, и так далее — цифра 8 меняется на 1, а цифра 9 меняется на 0

Тернарный оператор

Синтаксис:

```
значение if условие else значение
```

Программа (OddEven.py):

```
# Вводится число:
num=int(input("Введите целое число: "))
# Использование тернарного оператора:
res="четное" if num%2==0 else "нечетное"
# Сообщение:
print("Это "+res+" число")
```

Результат выполнения:

```
Введите целое число: 123
Это нечетное число
```

```
Введите целое число: 124
Это четное число
```

Обработка ошибок

Синтаксис:

```
try:
    # команды
except:
    # команды
```

- Если в try-блоке ошибок нет, except-блок игнорируется
- Если в try-блоке возникла ошибка, начинает выполняться except-блок

Программа (TryExcept.py):

```
print("Обработка исключений")
# Контролируемый код:
try:
    num=int(input("Введите целое число: "))
    print("Вы ввели число "+str(num))
# Обработка исключения:
except:
    print("Нужно было ввести целое число!")
print("Спасибо за сотрудничество!")
```

```
Обработка исключений
Введите целое число: 123
Вы ввели число 123
Спасибо за сотрудничество!
```

```
Обработка исключений
Введите целое число: 123.0
Нужно было ввести целое число!
Спасибо за сотрудничество!
```

Обработка ошибок - 2

Синтаксис:

```
try:
    # команды
except Тип_ошибки:
    # команды
except Тип_ошибки:
    # команды
...
except Тип_ошибки:
    # команды
```

Для ошибок разных типов (классов) можно использовать разные except-блоки

ValueError - ошибка при преобразовании значений (например, текста в число)

ZeroDivisionError - ошибка деления на ноль

TypeError - ошибка типа

IndexError - ошибка индекса

SyntaxError - синтаксическая ошибка

NameError - ошибка доступа к переменной (нет переменной с таким именем)

Обработка ошибок - 2

Программа (TryExcept_2.py):

```
print("Операции со списком чисел...")
# Контролируемый код:
try:
    nums=eval(input("Введите числовой список: "))
    print("Получено значение: "+str(nums))
    a=int(nums[0])
    b=int(nums[3])
    print(str(a)+"/"+str(b)+"="+str(a/b))
# Обработка исключений:
except ValueError:
    print("ValueError: ошибка при преобразовании!")
except ZeroDivisionError:
    print("ZeroDivisionError: попытка деления на ноль!")
except TypeError:
    print("TypeError: недопустимая операция!")
except IndexError:
    print("IndexError: неверный индекс элемента!")
except SyntaxError:
    print("SyntaxError: невозможно вычислить выражение!")
except NameError:
    print("NameError: неверный идентификатор!")
print("Завершение программы.")
```

```
Операции со списком чисел...
Введите числовой список: [2,4,6,5,1]
Получено значение: [2, 4, 6, 5, 1]
2/5=0.4
Завершение программы.
```

Обработка ошибок - 2

Результат выполнения:

```
Операции со списком чисел...
Введите числовой список: [2,4,6,5,1]
Получено значение: [2, 4, 6, 5, 1]
2/5=0.4
Завершение программы.
```

```
Операции со списком чисел...
Введите числовой список: "Python"
Получено значение: Python
ValueError: ошибка при преобразовании!
Завершение программы.
```

```
Операции со списком чисел...
Введите числовой список: 12345
Получено значение: 12345
TypeError: недопустимая операция!
Завершение программы.
```

```
Операции со списком чисел...
Введите числовой список: [1,2,3]
Получено значение: [1, 2, 3]
IndexError: неверный индекс элемента!
Завершение программы.
```

```
Операции со списком чисел...
Введите числовой список: Python
NameError: неверный идентификатор!
Завершение программы.
```

```
Операции со списком чисел...
Введите числовой список: Hello World!
SyntaxError: невозможно вычислить выражение!
Завершение программы.
```

```
Операции со списком чисел...
Введите числовой список: [1,2,3,0,5]
Получено значение: [1, 2, 3, 0, 5]
ZeroDivisionError: попытка деления на ноль!
Завершение программы.
```

Задание - 9

Напишите программу, которая определяет, какое число (положительное или отрицательное) ввел пользователь. Использовать тернарный оператор

Задание - 10

Напишите программу, в которой пользователь вводит два действительных числа, а программа проверяет, какое из чисел больше. Используйте тернарный оператор и обработку сключительных ситуаций

Домашнее задание

[1] Напишите программу, в которой пользователь вводит список целочисленных значений, а также верхнюю границу для вычисления суммы. Программа вычисляет сумму натуральных чисел, но за исключением тех, которые входят в список. Так, если пользователь ввел список [2,5,6] и 10 в качестве верхней границы суммы, то вычисляется сумма чисел от 1 до 10, но без учета чисел 2, 5 и 6

[2] Напишите программу, в которой пользователь вводит три числа, а программа определяет, может ли существовать треугольник со сторонами, длина которых равняется введенным значениям. Условие существования треугольника такое: сумма двух любых (из трех введенных) чисел должна быть больше третьего числа