



*School of
Computer
Science*

ЦИКЛ WHILE

ПРОГРАММИРОВАНИЕ НА PYTHON

Лекции для IT-школы



ВОПРОСЫ ПО ПРОШЛОМУ ЗАНЯТИЮ

1. Если две переменные ссылаются на одно значение, то что вернет выражение:
`id(var_1) == id(var_2)`
2. Можно ли скорректировать значение неизменяемого типа данных после его присваивания переменной?
3. Какой приём используется для оператора `if` с несколькими условиями, чтобы убедиться, в отсутствии непредусмотренных условий?
4. При каких условиях разумно использовать тернарный оператора выбора `...if...else...`?



ВОПРОС ПО ПРОШЛОМУ ЗАНЯТИЮ

Какая из операций изменяет строку `str_var`:

1. `str_var[0] = 'A'`
2. `str_var[:] = ''`
3. `str_var.replace('a', 'A')`
4. никакая из вышеперечисленных



ВОПРОС ПО ПРОШЛОМУ ЗАНЯТИЮ

- Как в Python определяется константа:
 1. При объявлении используется ключевое слово `const`
 2. При объявлении используется ключевое слово `final`
 3. Это обычная переменная, объявленная в начале области видимости, имя которой задано в верхнем регистре
 4. В виде макро-подстановки после ключевого слова `#define`

?



ВОПРОС ПО ПРОШЛОМУ ЗАНЯТИЮ

Схема оператора ветвления:

if <логическое выражение 1>:

 <блок кода 1>

elif <логическое выражение 2>:

 <блок кода 2>

elif <логическое выражение 3>:

 <блок кода 3>

else:

 <блок кода 4>

**Какое количество блоков кода *как минимум*,
и *как максимум*, здесь выполнится?**



ВОПРОС ПО ПРОШЛОМУ ЗАНЯТИЮ

В чем разница между этими примерами кода:

```
if <условие 1>:  
    <блок кода 1>  
elif <условие 2>:  
    <блок кода 2>  
elif <условие 3>:  
    <блок кода 3>  
else:  
    <блок кода 4>
```

```
if <условие 1>:  
    <блок кода 1>  
if <условие 2>:  
    <блок кода 2>  
if <условие 3>:  
    <блок кода 3>  
<блок кода 4>
```



ПРОВЕРОЧНЫЙ ВОПРОС

Какая проблема есть в этом коде?

```
name = 'Egyptian Mummy'
...
age = 3000
...
if name == 'Alice':
    print('Hi, Alice.')
elif age < 12:
    print('You are not Alice, kiddo.')
elif age > 100:
    print('You are not Alice, grannie.')
elif age > 2000:
    print('Unlike you, Alice is not undead, immortal beast.')
```

Как ее можно решить?



БАЗОВАЯ СТРУКТУРА ЦИКЛА WHILE

Отступы
обязательны!

```
while <логическое выражение>:  
    ← КОД, выполняемый при True  
        для логического  
        выражения...  
[else:  
    ← КОД, выполняемый, если блок  
        while нормально завершился  
        или не выполнялся вовсе  
]
```




КАК МЫ БЫ ЖИЛИ БЕЗ ЦИКЛОВ...

```
>>> figure = 7
>>> multiplier = 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 1 = 7
>>> multiplier = multiplier + 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 2 = 14
>>> multiplier = multiplier + 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 3 = 21
>>> multiplier = multiplier + 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 4 = 28
>>> multiplier = multiplier + 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 5 = 35
>>> multiplier = multiplier + 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 6 = 42
>>> multiplier = multiplier + 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 7 = 49
>>> multiplier = multiplier + 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 8 = 56
>>> multiplier = multiplier + 1
>>> print('figure, 'x', multiplier, '=', figure * multiplier)
7 x 9 = 63
```



ЦИКЛ ОБЛЕГЧАЕТ ПОВТОРЕНИЕ ОДНОТИПНЫХ ДЕЙСТВИЙ

«Быстрая» таблица умножения для семерки:

```
>>> figure = 7
>>> multiplier = 1
>>> while multiplier < 10:
    print('7 x', multiplier, '=', figure * multiplier)
    multiplier = multiplier + 1
```

```
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
>>> |
```



ЦИКЛ WHILE. ТИПОВАЯ СТРУКТУРА

while **a** логический_оператор **b**:

действие(я)

изменение **a**

Изменение условия цикла в его теле
обязательно, если не используется **break**



ЦИКЛ WHILE. ОПРЕДЕЛЕНИЯ

- Блок кода, исполняемый в цикле, называется ***итерацией*** или ***телом цикла***
- Обычно перед входом в цикл иницииируют ***управляющую переменную***, а внутри цикла **while** ее значение **изменяют**
- Даже при изменении управляющей переменной в теле цикла нужно следить, чтобы условие цикла хоть когда-нибудь стало ложным



ПРАКТИЧЕСКОЕ ЗАНЯТИЕ НА WHILE

См. `losing_battle_bad.py`



- Программа, моделирует последнюю битву главного героя в Action-игре
- Если здоровья биться с троллями уже не хватает, то цикл битвы должен завершиться
- Так в чем же проблема?



НАМЕРЕННО «БЕСКОНЕЧНЫЙ» ЦИКЛ

- Рассмотрите скрипт `strange_counter.py`
- Команда `break` внутри тела цикла передает управление за его пределы
- Команда `continue` внутри тела цикла передает управление на его очередную итерацию:
 - То есть происходит возврат к началу цикла
 - При этом снова проверяется условие цикла и, если оно `True`, то тело цикла снова будет выполняться



ВЛОЖЕННЫЕ ЦИКЛЫ WHILE. ПРИМЕР С ELSE

Поиск простых чисел до NUM_MAX:

```
>>> NUM_MAX = 20
>>> num = 2
>>> while num < NUM_MAX:
    div = 2
    while div < num:
        if num % div == 0:
            break
        div += 1
    else:
        print(num, "простое число")
    # конец внутреннего цикла while div < num
    num += 1
# конец внешнего цикла while num < NUM_MAX
```

верхняя граница диапазона для поиска простых чисел (константа)
проверяемое число будет лежать в диапазоне от num до NUM_MAX
цикл для перебора num
проверим будет ли num делиться на div, начиная с div = 2
num делится на div - это число имеет целочисленный делитель
т.е. оно составное, перейдем к следующему
это аналог присваивания div = div + 1
не нашли целочисленный делитель в цикле - это простое число
это аналог присваивания num = num + 1

```
2 простое число
3 простое число
5 простое число
7 простое число
11 простое число
13 простое число
17 простое число
19 простое число
>>>
```

Смотрите скрипт `prime_numbers.py`



ПРАКТИЧЕСКОЕ ЗАДАНИЕ №1.

ТАБЛИЦА УМНОЖЕНИЯ ДО 10-ТИ

- Второклассник учит таблицу умножения
- Сегодня он должен сдать её наизусть, но вот беда, забыл некоторые результаты
- Напишите программу, которая распечатает шпаргалку малышу:

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100



ПЛАНИРОВАНИЕ НА ПСЕВДОКОДЕ

- Алгоритм зарабатывания \$ 1 000 000:
 - Если вы способны придумать новый полезный товар – выпустите его в свет, иначе
 - Выпустите существующий товар под своей маркой
- А именно:
 - Создайте рекламный ролик своего товара
 - Покажите этот ролик по TV
 - Назначьте цену \$100 за единицу товара
 - Продайте 10 000 единиц товара



ПРАКТИЧЕСКОЕ ЗАДАНИЕ №2.

ИГРА «ОТГАДАЙ ЧИСЛО»

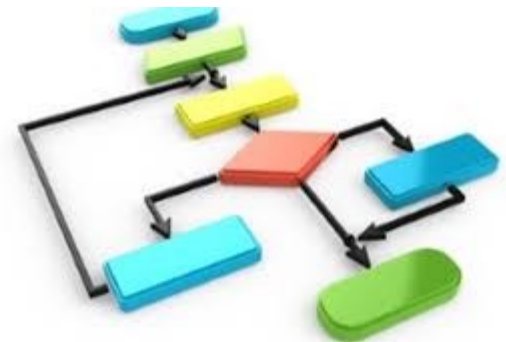
- Программа загадывает случайное число в диапазоне от 1 до 100 и не сообщает его
- Пользователь пытается угадать это число
- Если не угадал, то выдается подсказка:
 - «Вы ввели меньшее число»
 - или
 - «Вы ввели большее число»
- Цикл подбора числа продолжается, пока человек не угадает загаданное число



1-Я ВЕРСИЯ ПСЕВДОКОДА ПРОГРАММЫ «ОТГАДАЙ ЧИСЛО»

- Выбрать случайное число
- До тех пор, пока игрок его не отгадает:
 - Предоставить игроку возможность отгадывать
- Поздравить игрока

guess_num_template.py





2-Я ВЕРСИЯ ПСЕВДОКОДА ПРОГРАММЫ «ОТГАДАЙ ЧИСЛО»

- Поздороваться с игроком и объяснить ему правила игры
- Выбрать случайное число от 1 до 100
- Установить переменные для отгадывания и количества попыток
 - Пока указанное игроком число не совпадает с загаданным:
 - если оно больше загаданного – запросить число поменьше
 - иначе – запросить число побольше
 - Вновь предложить игроку отгадать число
 - Увеличить порядковый номер попытки на 1
- Поздравить игрока с победой
- Сообщить сколько попыток потребовалось



ПРАКТИЧЕСКОЕ ЗАДАНИЕ. ИГРА «ОТГАДАЙ ЧИСЛО» V.2

- Измените программу таким образом, чтобы у игрока было ограниченное количество попыток:
 - 1) Указанное в константе в тексте программы
 - 2) Запрошенное у игрока на старте. Если игрок вводит пустое значение, то использовать ограничение по умолчанию из пункта 1
 - 3)* Указанное в качестве параметра командной строки, а если отсутствует, то см. пункт 2
- Если игрок не укладывается в заданное число попыток и проигрывает, то программа должна выводить очень суровый текст



ДОМАШНЕЕ ЗАДАНИЕ. ИГРА «ЗАДУМАЙ ЧИСЛО»

- Спроектируйте алгоритм игры, в которой число от 1 до 100 загадывает человек, а отгадывает компьютер
- Какой должна быть оптимальная стратегия отгадывания?
- Сколько попыток нужно, чтобы отгадать число в диапазоне от 0 до 100 с помощью этой стратегии?
- Как ограничить пользователя, если он будет врать компьютеру?
- Реализуйте игру на Python по продуманному алгоритму

СПАСИБО ЗА ВНИМАНИЕ !
ВОПРОСЫ ?



*School of
Computer
Science*