

*School of  
Computer  
Science*

# **ОШИБКИ. ИСКЛЮЧЕНИЯ. ПЕРЕМЕННЫЕ. МОДУЛИ. ОПЕРАЦИИ. ОСНОВНЫЕ ТИПЫ ДАННЫХ. ПРОГРАММИРОВАНИЕ НА PYTHON**

**Лекции для IT-школы**



## WEB-редакторы для Python:

- <http://www.pythontutor.com/live.html#mode=edit>  
– визуальный редактор и отладчик Python
- <https://trinket.io/python3> или  
<https://repl.it/languages/python3> – Python 3 в  
браузере для тех, у кого проблемы с Python  
на учебном компьютере

## Материалы лекций:

- <https://gitlab.com/slukashenko75/python-course-2019-2020>



# НА ПЕРВОМ ЗАНЯТИИ МЫ УЗНАЛИ

- Start / All Programs / Python 3.x / IDLE
- `exit()`, `help()` для выхода и справки
- Математические операторы: `+` `-` `*` `/` `//` `%` `**`
- Строки можно сливать (конкатенировать) с помощью оператора `+`
- Переменные – именованные данные, которыми оперирует программа
- Оператор присваивания `=` нужен для активации переменных
- `input('Приглашение: ')` – ввод строки
- `print(...)` – вывод информации



К каким категориям относятся эти языки программирования:

1. Python, C, C++, Java, C#
2. Java Script, PHP
3. Pascal, Basic
4. Машинный код, Ассемблер



# ВОПРОСЫ

1. Что такое транслятор?
2. Какие типы трансляторов вам известны?
3. Трансляция какого типа используется в Python?
4. Рецензирование (обзор) кода это:
  - a) Мера повышения качества программного обеспечения
  - b) Способ наказания непослушных программистов
  - c) Процедура определения уровня программиста – junior, middle или senior



# ВОПРОСЫ

– Определите результаты выражений:

1)  $2 * 3 + 4$

2)  $2 * (3 + 4)$

3)  $2 + 3 * 4$

4)  $6 - 5 / 2$

5)  $10 // 4 + 10 \% 3$

6)  $(5 + 2) ** 2 - 3 * 2$



# ОШИБКА = BUG

9 сентября 1947

Вычислительная  
машина  
Harvard Mark II

“First actual case  
of bug being  
found”

9/9

0800 Antan started  
1000 " stopped - antan ✓

1300 (032) MP-MC { 1.2700 9.037 847 025  
2.130476415 (2) 4.615925059(-2) }  
(033) PRO 2 2.130476415  
convd 2.130676415

Relays 6-2 in 033 failed special speed test  
in relay 10.000 test.

Relays changed

1100 Started Cosine Tape (Sine check)  
1525 Started Multi-Adder Test.

1545 Relay #70 Panel F  
(moth) in relay.

First actual case of bug being found.

1700 closed down.



# ИСКЛЮЧЕНИЯ ДЛЯ ОБРАБОТКИ ОШИБОК

Отступы  
обязательны!

```
try:
    ←→ ИСПЫТУЕМЫЙ КОД
    ...
except exception1 [as var1]:
    ←→ реакция на исключение 1
    ...
except exceptionN [as varN]:
    ←→ реакция на исключение N
[else:
    ←→ код при отсутствии ошибок]
```





# ПРАКТИЧЕСКАЯ РАБОТА.

## «СЛОЖЕНИЕ ИЛИ КОНКАТЕНАЦИЯ»

- Пишем программу, которая запрашивает ввод двух значений
- Если хотя бы одно из них не является числом, то выполняем конкатенацию этих значений, т.е. соединение строк
- В остальных случаях введенные числа суммируются
- Условные операторы и функции работы со строками не используем, т.к. мы их еще не рассматривали



- Синтаксические ошибки:
  - Обнаруживаются компилятором на раннем этапе, при анализе текста программы
  - Для интерпретатора (Python) – могут проявляться только при исполнении программы
- Ошибки времени исполнения:
  - Всегда обнаруживаются при исполнении программы
  - Могут проявляться при определенных условиях
- Логические ошибки



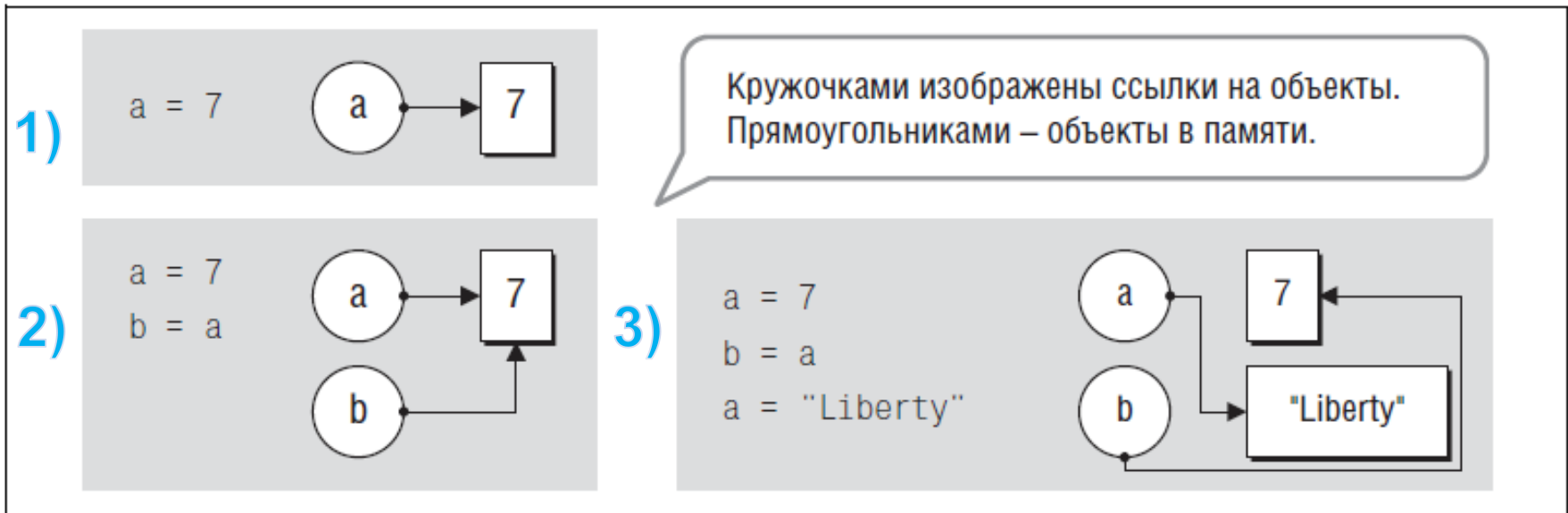
# ПРАКТИЧЕСКОЕ ЗАДАНИЕ 1

## ЛОГИЧЕСКАЯ ОШИБКА

- В скрипте `trust_fund_bad.py` текст программы
- Программа вычисляет общую сумму издержек в месяц вашего друга-миллионера
- Необходимо:
  1. Выполнить программу, ввести запрошенные данные и определить работает ли она корректно
  2. При обнаружении проблемы – предложить исправления
  3. Обработать ввод данных, которые не являются целыми числами



Переменные в Python – это **ссылки** на объекты:



См. ролик [Что такое переменные?](#)



# ИМЕНА ПЕРЕМЕННЫХ. ТРЕБОВАНИЯ

- Используются для идентификации ссылок на объекты в Python
- Состоят из букв, цифр и знака подчеркивания
- Не могут начинаться с цифры
- Регистр в Python ИМЕЕТ значение:
  - Переменные `accountBalance` и `AccountBalance` – РАЗНЫЕ



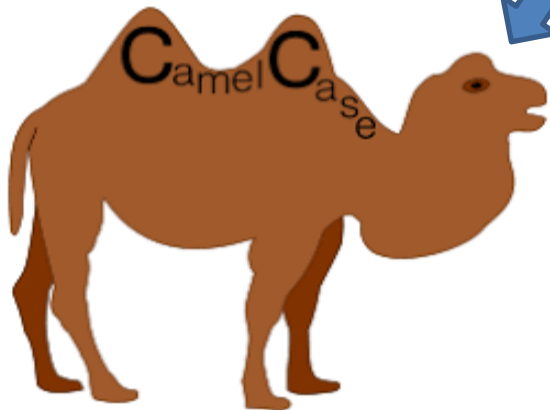
# ИМЕНА ПЕРЕМЕННЫХ. СТИЛЬ

- Имена должны выбираться так, чтобы они описывали суть представляемых данных:
  - `debt_amount`
  - `avail_balance`
  - ...
- Выбираются по принятым соглашениям, стилю и обычаям языка
- Не должны быть слишком длинными (рекомендуется не более 15 символов)
- Следует избегать имен на русском языке

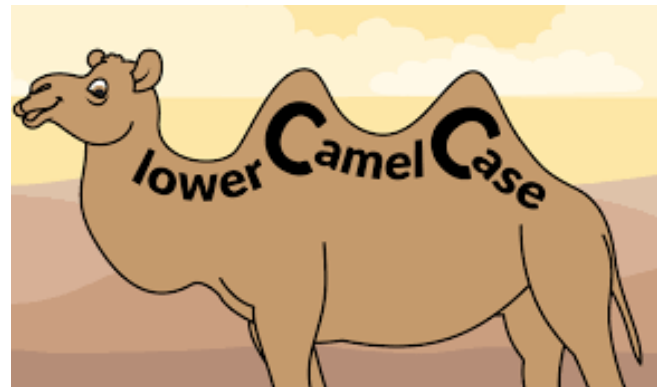


# ИМЕНА ПЕРЕМЕННЫХ. «ВЕРБЛЮЖИЙ» СТИЛЬ

## CamelCase



Ограниченно  
используется  
в Python



Является  
общепринятым  
в Java и C++



## snake\_case\_name

- Является общепринятым для Python
- Используется для обозначения переменных, имен функций и методов
- Имя переменной – это, обычно, минимум одно английское слово
- При необходимости, добавляются еще слова на английском через знак “\_”





# ИМЕНА ПЕРЕМЕННЫХ В PYTHON.

## ПРИМЕРЫ ОШИБОЧНЫХ ИМЕН

Ошибочное имя	Вариант исправления
<code>tax%</code>	<code>tax_percent</code>
<code>1code</code>	<code>code_1</code>
<code>sign!</code>	<code>exclamation_sign</code>
<code>amount-fee</code>	<code>clear_amount</code>



# ИМЕНА ПЕРЕМЕННЫХ В PYTHON.

## ПРИМЕРЫ ПЛОХИХ ИМЕН

Очень плохое имя	Хорошее имя
<b>o</b>	<b>output_data</b>
<b>l</b>	<b>line_number</b>
<b>personal_current_account _balance_by_the _end_of_month</b>	<b>curr_acct_balance</b>
<b>td</b>	<b>total_debt</b>



- `import sys` – информация о системе
- Обращение к содержимому модуля происходит по точечной нотации:
  - `sys.version`
- Ключевые слова языка:
  - `import keyword`
  - `keyword.kwlist`
- `dir(__builtins__)` – встроенные функции языка
  - Можно перекрывать, но это плохой стиль
  - `del <имя_переменной>` – удаление имени в локальном контексте



- Служат для выполнения действий над данными:



- Оператор присваивания задает связь между данными и переменной, которая нужна для доступа к ним:

`user_name = 'Вася'`



# ОСНОВНЫЕ ТИПЫ ДАННЫХ

- Числовые:
  - Целые числа (`int`):
    - 4, 687, -45, 0
  - Числа с плавающей точкой (`float`)
    - 1.45, 3.789654, 2.220446049250313e-16
  - Логические (`bool`)
    - True, False
- Строки (`str`)
  - 'What is your name', '6589', "В других кавычках"



- Целые числа (`int`):
  - 4, 687, -45, 0
- Применимые операторы:
  - =, +, -, \*, \*\*, /, //, %, >, <, >=, <=, !=, ==
- Приведение к типу `int`:
  - `int(3.748)` ➡ 3
  - `int("386")` ➡ 386



# ГЕНЕРАТОР СЛУЧАЙНЫХ ЧИСЕЛ

- `import random` – импортируем модуль `random`
- `dir(random)` – список функций, применимых для генерации случайных чисел
- Практический пример:
  - Рассмотрите скрипт `dice_roller.py`
  - `random.randint(min, max)` – генерация случайного целого числа в указанном диапазоне
  - `random.randrange(max)` – генерация случайного целого числа от нуля до `max-1`
  - `random.choice( [1, 2, 3, 4, 5, 6] )` – выбор случайного числа из указанного списка





# ЧИСЛА С ПЛАВАЮЩЕЙ ТОЧКОЙ

- Числа с плавающей точкой (`float`)
  - 1.45, 3.789654, 2.220446049250313e-16
- Применимые операторы:
  - `=`, `+`, `-`, `*`, `/`, `//`, `%`, `>`, `<`, `>=`, `<=`, `!=`, `==`
- Если в операции с числами хотя бы один операнд `float`, то результат будет `float`
- Приведение к типу: `float("string")`
  - `float(386)`                      386.0
  - `float("38.46")`                38.46





# ЧИСЛА С ПЛАВАЮЩЕЙ ТОЧКОЙ

## ЛОВУШКА ТОЧНОСТИ

– Не следует сравнивать float-значения операторами “**==**” и “**!=**”

- `>>> 0.1 + 0.1 + 0.1`
- `0.30000000000000004`
- `>>> 0.1 + 0.1 + 0.1 == 0.3`
- `False`
  
- `>>> 9**19 - int(float(9**19))`
- `89`



# ЧИСЛА С ПЛАВАЮЩЕЙ ТОЧКОЙ

## РЕШЕНИЕ ПРОБЛЕМЫ ТОЧНОСТИ

### 1. Минимально возможное значение float:

- `import sys`
- `sys.float_info.epsilon`

Значения float считаем равными, если:

- `import math`
- `math.fabs(float1 - float2) <= sys.float_info.epsilon`

### 2. Использование специального типа данных `Decimal` из модуля `decimal`



# ЧТО ЗДЕСЬ ПРОИСХОДИТ?

## СМ. ТАКЖЕ [0.30000000000000004.COM](http://0.30000000000000004.COM)

```
Python 3.6.1 Shell
File Edit Shell Debug Options Window Help
Python 3.6.1 (v3.6.1:69c0db5, Mar 21 2017, 17:54:52) [MSC v.1900 32 bit (Intel)]
on win32
Type "copyright", "credits" or "license()" for more information.
>>> import sys
>>> sys.float_info.epsilon
2.220446049250313e-16
>>> num = 0.3 - 0.1 - 0.1 - 0.1
>>> num == 0
False
>>> num
-2.7755575615628914e-17
>>> abs(num) < sys.float_info.epsilon
True
>>>
>>> from decimal import Decimal
>>> num_d = Decimal('0.3') - Decimal('0.1') - Decimal('0.1') - Decimal('0.1')
>>> num_d == 0
True
>>> num_d
Decimal('0.0')
```



# ЧИСЛА С ПЛАВАЮЩЕЙ ТОЧКОЙ

## МАКСИМАЛЬНО ВОЗМОЖНОЕ ЗНАЧЕНИЕ

– Максимально возможное значение float:

```
1  >>> # inf = infinity, т.е. бесконечность
2  >>> float_max = float("inf")
3  >>> float_max
4  inf
5
6  >>> 9999999999999999.999999 < float_max
7  True
8
```



# РАСШИРЕННЫЙ МАТЕМАТИЧЕСКИЙ КАЛЬКУЛЯТОР

- `import math` – расширение стандартных математических операций Python
- `dir(math)` – список математических функций, применимых для действительных чисел
- Практическое задание №2:
  - Напишите программу, которая вычисляет длину гипотенузы прямоугольного треугольника, запрашивая длины его катетов
  - Следуйте правилам именования переменных
  - **Предусмотрите сообщение пользователю для случая неверного формата введенных данных**



# ЛОГИЧЕСКИЕ ВЫРАЖЕНИЯ

- Логические выражения (`bool`):
  - `True`, `False`, `0`, `28`, `1`, "Строка", ""
- Все операторы сравнения дают логический результат – `True` или `False`
- Применимые операторы:
  - `and`, `or`, `not`
- Приведение к типу `bool`:
  - `bool(-5.16)` ➡ `True`
  - `bool("")` ➡ `False`



# ТАБЛИЦЫ ЗНАЧЕНИЙ ЛОГИЧЕСКИХ ОПЕРАЦИЙ

Операнд 1	Операнд 2	Операнд 1 <b>and</b> Операнд 2
True	True	True
True	False	False
False	True	False
False	False	False

Операнд	<b>not</b> Операнд
True	False
False	True

– Практический пример:

Смотрим скрипт `bool_operations.py`

Операнд 1	Операнд 2	Операнд 1 <b>or</b> Операнд 2
True	True	True
True	False	True
False	True	True
False	False	False



- Строки (`str`)
  - `'What is your name?'`, `'6589'`
- Применимые операторы:
  - `=`, `+`, `*`, `>`, `<`, `>=`, `<=`, `!=`, `==`
- Приведение к типу `str`:
  - `str(3.1415)` ➡ `"3.1415"`
  - `str(386)` ➡ `"386"`







# НЕКОТОРЫЕ СТРОКОВЫЕ ОПЕРАЦИИ

- Длина строки – `len(“строка”)`
- Тиражирование строки – `string * n`
- Обращение к символу строки – `string[n]`
- Получение среза строки – `string[start:end]`
  - Символы в строке нумеруются с позиции 0
  - Позиция `end` в срезе не включается
- Практические примеры:
  - Рассмотрите скрипт `str_operations.py`
  - Рассмотрите скрипт `quotation_manipulation.py`



# ВОПРОСЫ

– Какое выражение даст результат **True**?

1)  $5.73 == 9.23$

2)  $5 \geq 3$

3)  $6.0 != 6$

4)  $4 < 9$



# ВОПРОСЫ

- Переменные `first` и `last` ссылаются на одно и то же значение типа `int`.
- Какие из этих выражений дают результат `True`?

- 1) `first != last`
- 2) `first >= last`
- 3) `first == last`
- 4) `last > first`



- Как в Python извлечь квадратный корень из числа?
- Как определить длину строки?
- Можно ли выполнить сложение строки и числа?
- Можно ли из числа с плавающей точкой сделать целое?
- Какой тип данных получится в результате вычисления выражения:  
 $1 + 2.0 + 3$



## ПРАКТИЧЕСКОЕ ЗАДАНИЕ №3. «СТОИМОСТЬ 2-УХ ТОВАРОВ»

- Покупатель приобрел два товара
- Первый товар стоит “А” рублей “В” копеек
- Второй – “С” рублей “D” копеек
- Определите сколько рублей и копеек стоят эти товары вместе?
- Программа должна запрашивать 4 числа и выводить 2 числа:
  - Сколько рублей стоят 2 товара
  - Сколько копеек стоят 2 товара



## ДОМАШНЕЕ ЗАДАНИЕ. «ЭЛЕКТРОННЫЕ ЧАСЫ»

- Электронные часы показывают время в формате «часы:минуты:секунды»
- То есть сначала записывается количество часов в виде числа от 0 до 23
- Потом, выводится количество минут и секунд как числа от 0 до 59
- С начала суток прошло N секунд
- $0 \leq N \leq 1000000$
- Запросите N у пользователя и выведите, что покажут часы

**СПАСИБО ЗА ВНИМАНИЕ !**  
**ВОПРОСЫ ?**



*School of  
Computer  
Science*