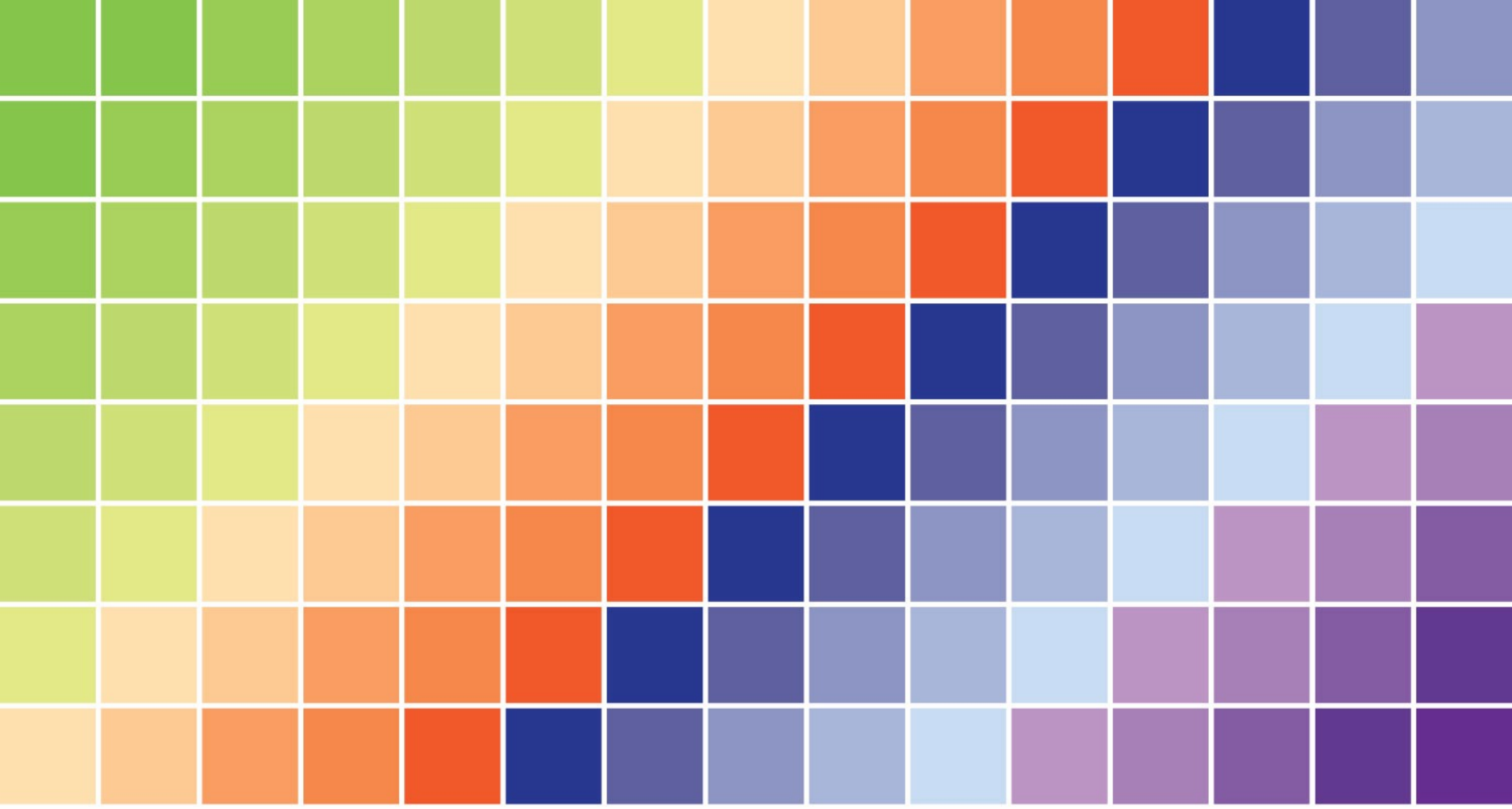


School of Computer Science

АНОНИМНЫЕ ФУНКЦИИ. ТЕСТЫ И ДОКУМЕНТАЦИЯ ДЛЯ ФУНКЦИЙ

ПРОГРАММИРОВАНИЕ НА PYTHON

Лекции для IT-школы

ВОПРОСЫ ПО ПРОШЛЫМ ЗАНЯТИЯМ. СПИСКИ

- Есть такой список:

  ```
  >>> temperatures = [-3, 0, 2.5, 4]
  ```

- Какое выражение даст значение 4:
  1. `>>> temperatures[-1]`
  2. `>>> temperatures[4]`
  3. `>>> temperatures[3:]`
  4. `>>> temperatures[3:4]`

ВОПРОСЫ ПО ПРОШЛЫМ ЗАНЯТИЯМ. СПИСКИ

– Выберите выражения, которые дадут 4:

1. >>> len([1, 2, 3, 4])
2. >>> len(["math"])
3. >>> min([10, 8, 4])
4. >>> sum([4])

ВОПРОСЫ ПО ПРОШЛЫМ ЗАНЯТИЯМ. СПИСКИ

- Есть такой код:

```
>>> grades = [80, 70, 60, 90]
>>> grades.sort()
>>> grades.insert(1, 95)
```

- На что, в итоге, ссылается grades:
  1. [60, 70, 80, 90, 95]
  2. [95, 60, 70, 80, 90]
  3. [60, 95, 80, 90]
  4. [60, 95, 70, 80, 90]

ВОПРОСЫ ПО ПРОШЛЫМ ЗАНЯТИЯМ. КОРТЕЖ

– Какой способ создать кортеж является недопустимым?

1. >>> tuple_var = ()
2. >>> tuple_var = (1)
3. >>> tuple_var = (1, 2, 3)
4. >>> tuple_var = (1, "two", 3.0)
5. >>> tuple_var = (1,)
6. >>> tuple_var = tuple()

ВОПРОСЫ ПО ПРОШЛЫМ ЗАНЯТИЯМ.
КОРТЕЖ

- Есть 2 кортежа: >>> tup_1, tup_2 = (1,2), (4,3)

- Найдите недопустимую операцию:
  1. >>> tup_new = tup_1 + tup_2
  2. >>> tup_triple = tup_1 * 3
  3. >>> flag = 2 in tup_1
  4. >>> sorted(tup_2)
  5. >>> tup_1.sort()
  6. >>> list(reversed(tup_2))
  7. >>> tup_2 = tup_2[::-1]
  8. >>> val = tup_1[-1]
  9. >>> tup_1.index(val)

ВОПРОСЫ ПО ПРОШЛЫМ ЗАНЯТИЯМ.
ПУСТЫЕ ФУНКЦИИ

- Как создать шаблон функции, но отложить реализацию кода этой функции на будущее?

1. Определить прототип функции, но оставить тело функции пустым
2. Добавить в теле функции строку со служебным словом pass
3. Добавить несколько пустых строк после определения функции

СВОИ ФУНКЦИИ. НЕОБХОДИМОСТЬ

- Структурирование кода:
  - Для часто вызываемого кода
  - В виде логически обособленных блоков кода (для создания библиотек функций)
- Повторное использование кода
- Коллективная разработка
- Инкапсуляция и параметризация кода
- **Новые области видимости переменных**

ПРОСТРАНСТВА ИМЁН

Пространство имён – отображение между идентификаторами и объектами

У блока кода, выполняемого в Python, есть три пространства имен:

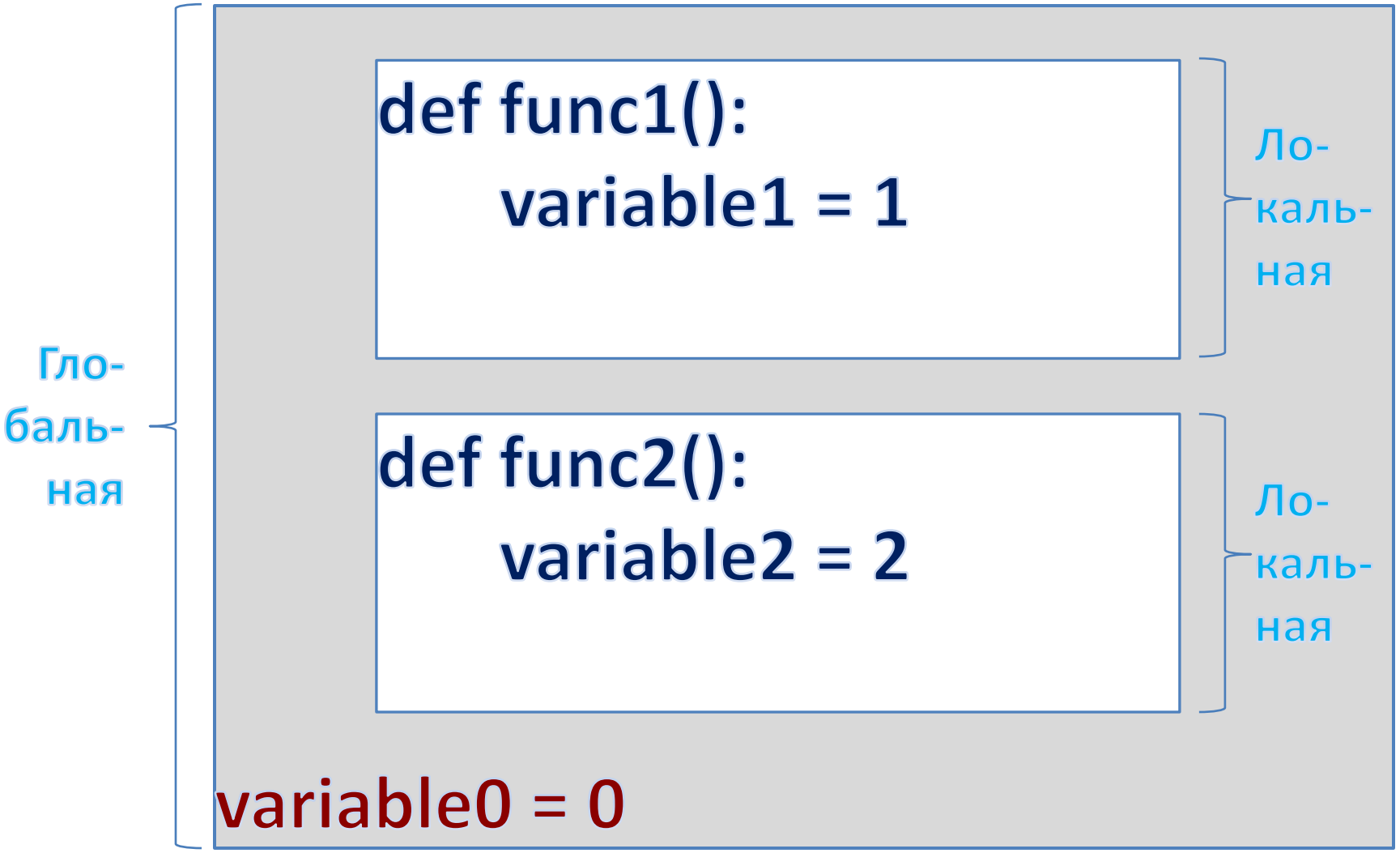
- Локальное
- Глобальное
- Встроенное

ПРОСТРАНСТВА ИМЁН

Порядок поиска идентификатора:

- Локальное
- Глобальное
- Встроенное
- Исключение NameError

ОБЛАСТИ ВИДИМОСТИ



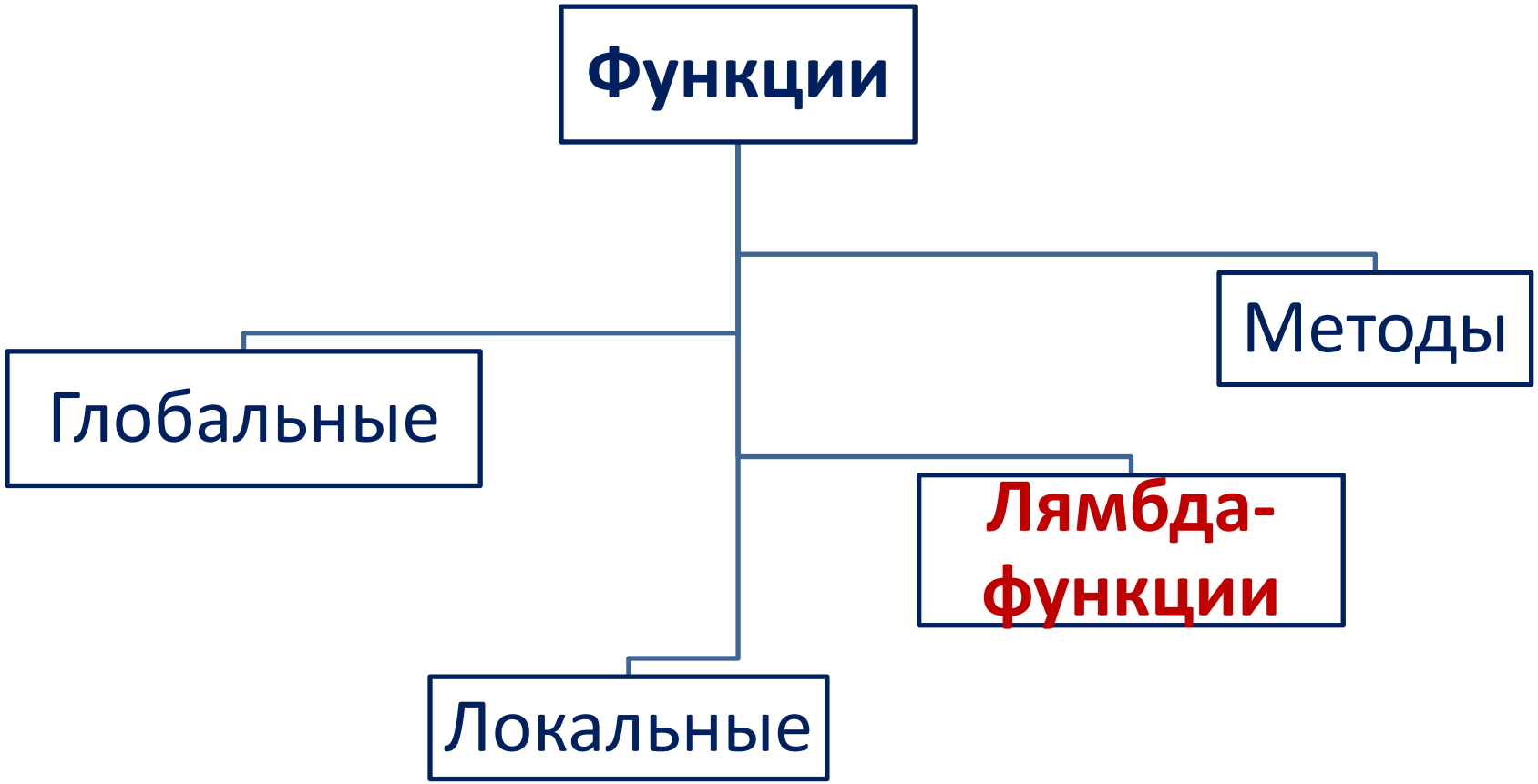
ПЕРЕКРЫТИЕ ПЕРЕМЕННЫХ. СТЕК ИСПОЛНЕНИЯ

- Смотрите примеры в scopes.py

- При исполнении функция занимает память для локальных переменных
- После исполнения память освобождается
- Память выделяется в ***стеке***
- Проследим пошаговое исполнение скрипта convert_min_sec.py

- Используем для этого сайт http://www.pythontutor.com

ВИДЫ ФУНКЦИЙ В PYTHON



АНОНИМНЫЕ ФУНКЦИИ

Создаются с помощью инструкции **lambda**:

lambda параметр1, параметр2, . . .:
выражение

```
>>> func = lambda x, y: x + y
>>> func(1, 2)
3
```

Анонимные функции могут содержать лишь **одно** выражение, но и выполняются они быстрее.
Их хорошо применять со встроенными функциями – такими как *sort()*

АНОНИМНЫЕ ФУНКЦИИ

Фильтрация последовательности *a_list* с помощью *filter()*

```
>>> a_list = [2, 18, 9, 17, 8, 12, 27]
```

С использованием **def**:

```
>>> def filter_func(x):
        if x % 3 == 0:
            return True
        return False

>>> print(list(filter(filter_func, a_list)))

# [18, 9, 12, 27]
```

С использованием **lambda**:

```
>>> print(list(filter(lambda x: x % 3 == 0,
                      a_list)))

# [18, 9, 12, 27]
```

ТЕСТЫ ФУНКЦИИ ЧЕРЕЗ DOC STRING

- Загрузите в Shell скрипт func_test.py
- Импортируйте модуль doctest:

  import doctest
- Протестируйте функцию:

  doctest.testmod()
- Определите ошибки в тестовых кейсах
- Исправьте тестовые кейсы и снова запустите тестирование

ПОДГОТОВКА HTML-ДОКУМЕНТАЦИИ TRIANGLE.HTML С ПОМОЩЬЮ PYDOC

Стандартный модуль pydoc, функция writedoc() используется для сохранения документации из docstring в HTML-документе

```
>>> import pydoc
>>> import os
>>>
>>> os.getcwd()
'C:\\Python37'
>>> os.chdir(r"C:\IT-School\Python\Year 11 2019-2020\Lesson 7\Scripts\DocstringSamples")
>>> os.getcwd()
'C:\\IT-School\\Python\\Year 11 2019-2020\\Lesson 7\\Scripts\\DocstringSamples'
>>>
>>> import triangle
>>> pydoc.writedoc(triangle)
wrote triangle.html
```

ПРАКТИЧЕСКОЕ ЗАДАНИЕ. СИСТЕМЫ СЧИСЛЕНИЯ

- Смотрите шаблон в convert_bin_dec_template.py

- Вспомните правила преобразования из десятичной системы счисления в двоичную и обратно
- Этот сайт поможет вспомнить: cdn.cs50.net/2016/x/psets/0/pset0/bulbs.html

- Напишите функции to_binary() и to_decimal() по заданным требованиям
- Найдите стандартные функции Python, которые делают такие же преобразования

СПАСИБО ЗА ВНИМАНИЕ !

ВОПРОСЫ ?

School of Computer Science