

Введение в программирование

**Алексей Гуревич,
СПБАУ РАН**

gurevich@ablab.spbau.ru

План

1. Введение
2. Установка Python
3. Переменные, типы данных
4. Операторы
5. Ввод-вывод
6. Функции
7. Модули
8. Вызов внешних программ
9. Разное

1. Введение

- Программа – последовательность инструкций, предназначенных для исполнения компьютером
- Язык программирования – формальная знаковая система, предназначенная для записи программ. ЯП определяет набор лексических, синтаксических и семантических правил
- Интерпретация – покомандный анализ, обработка и тут же выполнение исходной программы или запроса
- Инструкция (оператор) – наименьшая автономная часть языка программирования; команда

2. Установка Python

- **Linux, MacOS:**

- У вас уже есть Python!
- *python --version*

- **Windows:**

- Скачать с <http://python.org/download/> инсталлятор
- Или скопировать его с флешки
- Установить

2. Установка Python

- Пробуем создать и запустить простейшую программу:

1. Создаем в блокноте файл *program.py*

2. Пишем в нём:

```
print "Hello bioinformatic world!"
```

3. Сохраняем и запускаем:

```
python program.py
```

3. Переменные, типы данных

- Переменная – поименованная область памяти, адрес которой можно использовать для осуществления доступа к данным и изменять значение в ходе выполнения программы
- Простейшие типы данных:
 - **Bool:** *True, False*
 - **Int:** *0, 42, -146, ...*
 - **Float:** *0. , -1.5, 2.1e3, ...*
 - **String:** *'c', "str", "my name is 'Alexey' ", ...*
- Динамическая типизация:

```
t = 1
```

```
print t
```

```
t = 'str'
```

```
print t
```

4. Операторы

- Присваивание:

=

Примеры: a = 1 a = False

- Арифметические:

+, -, *, /, %, += .., -= ..

Примеры: a = 1 + 1 a = b % c a += 4

- Логические:

and, or, not, ==, <>

Примеры: a = (1 > 5) or (2 < 4) a = not a

4. Операторы

- Оператор ветвления:

if condition1:

 do something # condition1 is *True*

elif condition2:

 do something else # cond1 is False but cond2 is True

else:

 do something else # cond1 is False and cond2 is False

3. Переменные, типы данных

- Список – тип данных, представляющий собой упорядоченный набор значений, в котором некоторое значение может встречаться более одного раза
- Операции:
 - **Создание:** $a = [1,2,3]$, $a = [True, False, True]$, $a = []$, ...
 - **Обращение к элементу:** $a[3]$ – четвертый элемент списка
 - **Сложение списков:** $a = [1,2,3] + [4,5,6]$
 - **Добавление элемента:** $a.append(42)$
 - **Удаление элемента:** $del a[2]$ – удаление третьего элемента списка
 - **Длина списка:** $len(a)$

4. Операторы

- Операторы цикла:

for i in [1, 2, 3]:

print "current i is", i

while condition:

print "condition is", condition

condition = not condition

- Специальные команды:
 - **break** – выйти из цикла
 - **continue** – перейти на следующую итерацию
- Для выполнения чего-то n раз:
 - **range(n)**

5. ВВОД-ВЫВОД

- Командная строка

- Вывод:

- ```
print "something", 123, True
```

- ```
print ("something" + str(123) )
```

- ```
print "something",
```

- Ввод:

- ```
var = input([prompt])
```

- ```
var = raw_input([prompt])
```

# 5. ВВОД-ВЫВОД

- **Файлы**

- **Начало работы:**

```
f = open('filename', 'r') # 'w', 'a'
```

- **Чтение:**

```
line = f.readline()
```

```
all = f.read()
```

- **Запись:**

```
f.write('string' + '\n')
```

- **Завершение работы:**

```
f.close()
```

# 6. Функции

- Функция – это проименованная часть программы, которая может вызываться из других частей программы столько раз, сколько необходимо

- Задание функции:

```
def my_function(arg1, arg2, arg3):
```

```
 # do something
```

```
 return r_value1, r_value2
```

- Вызов функции:

```
a, b = my_function(1, False, 'str')
```

# 7. Модули

- Модуль – функционально законченный фрагмент программы, оформленный в виде отдельного файла с исходным кодом
- Создание:  
`my_module.py`
- Использование:  
`import my_module`  
`my_module.foo()`

# 7. Модули

- Полезные модули Python
  - `os` – модуль для работы с ОС
    - `os.mkdir('dirname')`
    - `os.remove('filename')`
    - `os.getcwd()`
    - `os.chdir('dirname')`
    - `os.path.isfile('name')`   `os.path.isdir('name')`
    - `os.path.abspath('filename')`
    - `os.path.join('dirname1', 'dirname2', 'filename')`
    - `os.path.basename('fn')`   `os.path.dirname('fn')`

# 7. Модули

- Полезные модули Python

- `sys` – system-specific parameters and functions

```
sys.exit(0)
```

```
sys.maxint()
```

```
sys.argv # len(sys.argv) sys.argv[0] sys.argv[1]
```

## Пример:

```
if len(sys.argv) != 3:
```

```
 print("Usage: " + sys.argv[0] + " <input genes> <output genes>")
```

```
 sys.exit()
```

```
out = open(sys.argv[2], 'w')
```



# 8. ВЫЗОВ ВНЕШНИХ ПРОГРАММ

- Модуль subprocess

```
subprocess.call(['md', 'my_dir']) # mkdir
```

```
output = open('tmp.txt', 'w')
```

```
subprocess.call(['dir'], stdout=output) # ls
```

```
output.close()
```

# 9. Разное

- <http://rosalind.info/problems/list-view/?location=python-village>
- <http://docs.python.org/>
- <http://www.google.com/>

# Всё

- Спасибо за внимание!