Создание функции

```
# В круглых скобках называем аргументы
def new_function(arg1, arg2):
    # Прописываем инструкции
    ...
    # Возвращаем результаты
    return result1, result2
```

Проверка аргументов

```
def get_time(dist, speed):
    # Проверяем аргумент с помощью условия
    if speed <= 0:
        # С помощью raise возвращаем ValueError
        raise ValueError("Speed can't be less than 0")
    # Инструкции для корректных аргументов
    ...
```

Аргументы по умолчанию

```
# С помощью '=' присваиваем
# значение по умолчанию
def root(value, n=2):
    return value ** (1/n)

# Изменяемые типы данных должны
# создаваться в самом теле функции!
def new_function(in_list=None):
    if in_list is None:
        in_list = list()
    ...
```

Получение переменного числа аргументов

```
def new_function(*args, **kwargs):
    # В переменной args — кортеж из
    # порядковых аргументов
    # В kwargs — словарь из именованных
```

```
...
```

Передача аргументов

```
# Сначала порядковые, затем – именованные аргументы
new_function("Arg1", 24, city="Moscow")
# Распаковка списков и словарей в функцию:
new_list = [1,3,4,5]
how = {'sep': ', ', 'end': '; '}
# С помощью операторов * и **
print(*new_list, **how)
# 1, 3, 4, 5;
```

Lambda-функции

```
# От одного аргумента:
get_length = lambda x: len(x)
# От двух
mult = lambda x,y : x*y
# От неограниченного числа:
all_args = lambda *args, **kwargs:\
           (args, kwargs)
l = ['dd', 'bbb', 'aaa']
# Сортировка с lambda по длине слова,
# потом – по алфавиту
l.sort(key=lambda x: (len(x), x))
```