# Python как первый язык

День 3

# Стандартная библиотека Python

# Стандартная библиотека

– набор модулей, функций, утилит и пр. доступный в любой программе

# Стандартная библиотека

– набор модулей, функций, утилит и пр. доступный в любой программе

```
0: print(“Shalom”)
```

Функции

# Стандартная библиотека

– набор модулей, функций, утилит и пр. доступный в любой программе

```
0: print(“Shalom”)
```

## Функции

```
0: [1, 2, 3].append(4)
```

## Методы

# Стандартная библиотека

– набор модулей, функций, утилит и пр. доступный в любой программе

```
0: print(“Shalom”)
```

Функции

```
0: [1, 2, 3].append(4)
```

Методы

```
0: import math
```

Модули

# Стандартная библиотека

– набор модулей, функций, утилит и пр. доступный в любой программе

- Типы данных
- Синтаксические конструкции
- Средства для работы с вводом/выводом
- Поддержка разных форматов данных

# Встроенные функции, операторы и методы

```
0: print(“Shalom”)
1: len(“Shalom”)
2: abs(-20)
3: 23 % 8
```

# Модули Python

, которые нам пригодятся

# Модули Python

, которые нам пригодятся

```
0: import math
1: import sys
2: import os
3: import datetime
3: import re
```

- math - математические функции
- sys - для работы с интерпретатором
- os - для работы с операционной системой
- datetime - для работы с форматами времени
- re - для работы с регулярными выражениями

# math

```
0: from math import sqrt
1:
2: sqrt(16) # 4.0
```

# math

```
0: from math import sqrt
1:
2: sqrt(16) # 4.0
```

```
0: from math import sqrt, pi
1:
2: print(pi) # 3.141592653589793
```

# math

```
0: from math import sqrt
1:
2: sqrt(16) # 4.0
```

```
0: from math import sqrt, pi
1:
2: print(pi) # 3.141592653589793
```

```
0: import math
1:
2: print(math.pi) # 3.141592653589793
3: print(math.sqrt(16)) # 4.0
```

# math

Подключение всех функций из модуля

```
0: from math import *
```

# math

## Подключение всех функций из модуля

```
0: from math import *
```

## Использование псевдонимов

```
0: import math as m
1:
2: print(m.pi) # 3.141592653589793
```

# math

## Подключение всех функций из модуля

```
0: from math import *
```

## Использование псевдонимов

```
0: import math as m
1:
2: print(m.pi) # 3.141592653589793
```

## Использование псевдонимов у функций

```
0: from math import sqrt as sq
1:
2: print(sq(16)) # 4.0
```

# sys

Модуль для работы с интерпретатором Python

# sys

Модуль для работы с интерпретатором Python

```
0: import sys
1:
2: print(sys.argv)
```

# sys

Модуль для работы с интерпретатором Python

```
0: import sys
1:
2: print(sys.argv)
```

```
$ python test.py Hello World!
['hello.py', 'Hello', 'World!']
```

# re

Модуль регулярных выражений

# re

Модуль регулярных выражений

regex101.com

REGULAR EXPRESSION

1 MATCH - 6 STEPS

" Hello " gmixsu

TEST STRING

```
Hello, dude! Wazzup!
Digits: 123, 0xe3412
```

REGULAR EXPRESSION

4 MATCHES - 9 STEPS

" " g

TEST STRING

```
Hello, dude! Wazzup!
Digits: 123, 0xe3412
```

REGULAR EXPRESSION

8 MATCHES - 51 STEPS

" \d " g

TEST STRING

```
Hello, dude! Wazzup!
Digits: 123, 0xe3412
```

https://mail.yandex-team.ru/?
uid=1120000000035142&login=zeffirsky#inbox

31 MATCHES - 74 STEPS

" \w " g

TEST STRING

Hello, dude! Wazzup!
Digits: 123, 0xe3412

REGULAR EXPRESSION

6 MATCHES - 24 STEPS

" \w+ " g

TEST STRING

```
Hello, dude! Wazzup!
Digits: 123, 0xe3412
```

REGULAR EXPRESSION

1 MATCH - 4 STEPS

" 0x[a-f0-9]+ " i

TEST STRING

```
Hello, dude! Wazzup!
Digits: 123, 0xe3412
```

# re

Модуль регулярных выражений

```
0: import re
1:
2: pattern = r“\d+\s?\$”
```

# re

Модуль регулярных выражений

```
0: import re
1:
2: pattern = r“\d+\s?\$”
3: regexp = re.compile(pattern)
```

# re

## Модуль регулярных выражений

```
0: import re
1:
2: pattern = r“\d+\s?\$”
3: regexp = re.compile(pattern)
4: result = regexp.match(“””
5: I can give you only 10$.
6: But you ask for 20 $”””)
```

# re

## Модуль регулярных выражений

```
0: import re
1:
2: pattern = r“\d+\s?\$”
3: regexp = re.compile(pattern)
4: result = regexp.match(“””
5: I can give you only 10$.
6: But you ask for 20 $”””)
```

```
0: import re
1:
2: result = re.match(r“\d+\s?\$”, ”…”)
```

# Создание своего модуля

Файл: utils.py

```
01: def some(predicate, list):
02:     for item in list:
03:         if predicate(item):
04:             return True
05:     return False
06:
07: def every(predicate, list):
08:     for item in list:
09:         if not predicate(item):
10:             return False
11:     return True
```

# Создание своего модуля

```
>>> import utils
```

# Создание своего модуля

```
>>> import utils
>>> utils.some(lambda x: x > 2, [1,2,3])
True
```

# Создание своего модуля

```
>>> import utils
>>> utils.some(lambda x: x > 2, [1,2,3])
True
>>> utils.some(lambda x: x > 2, [1,2,0])
False
```

# Создание своего модуля

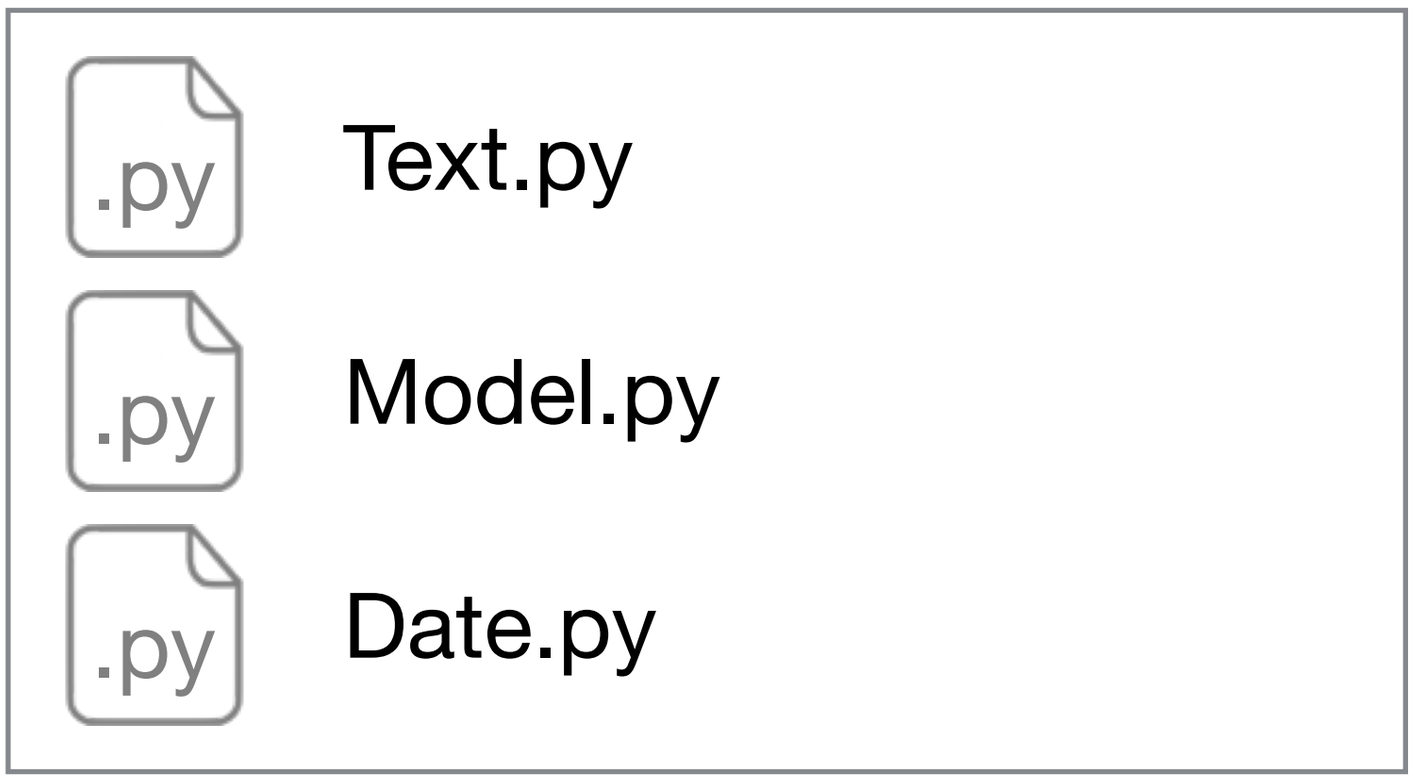
```
>>> import utils
>>> utils.some(lambda x: x > 2, [1,2,3])
True
>>> utils.some(lambda x: x > 2, [1,2,0])
False
>>> from utils import every
```

# Создание своего модуля

```
>>> import utils
>>> utils.some(lambda x: x > 2, [1,2,3])
True
>>> utils.some(lambda x: x > 2, [1,2,0])
False
>>> from utils import every
>>> every(lambda x: x > 2, [3,10,15])
True
```

# Пакеты

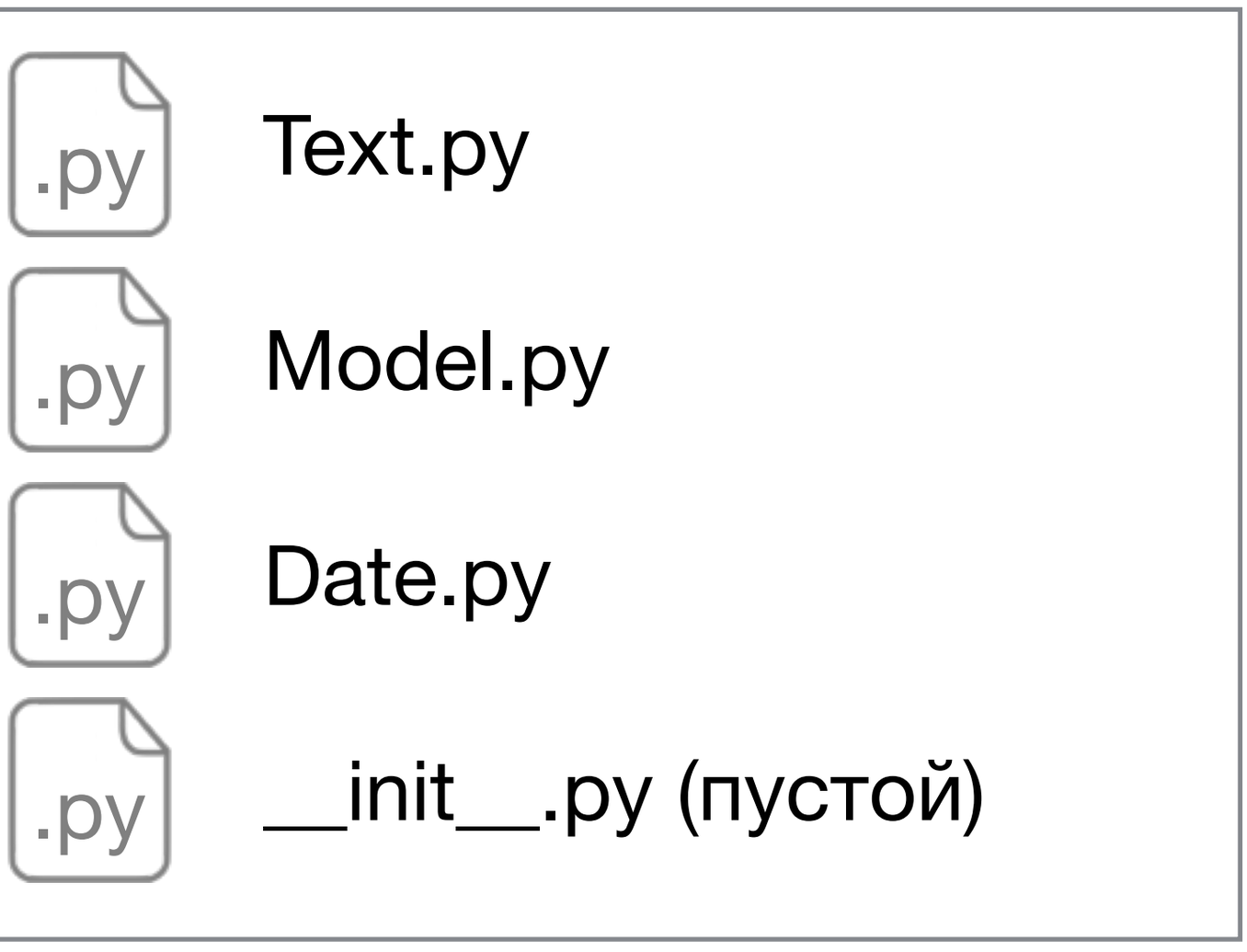
~/project/Utils




Общий пакет утилит

# Пакеты

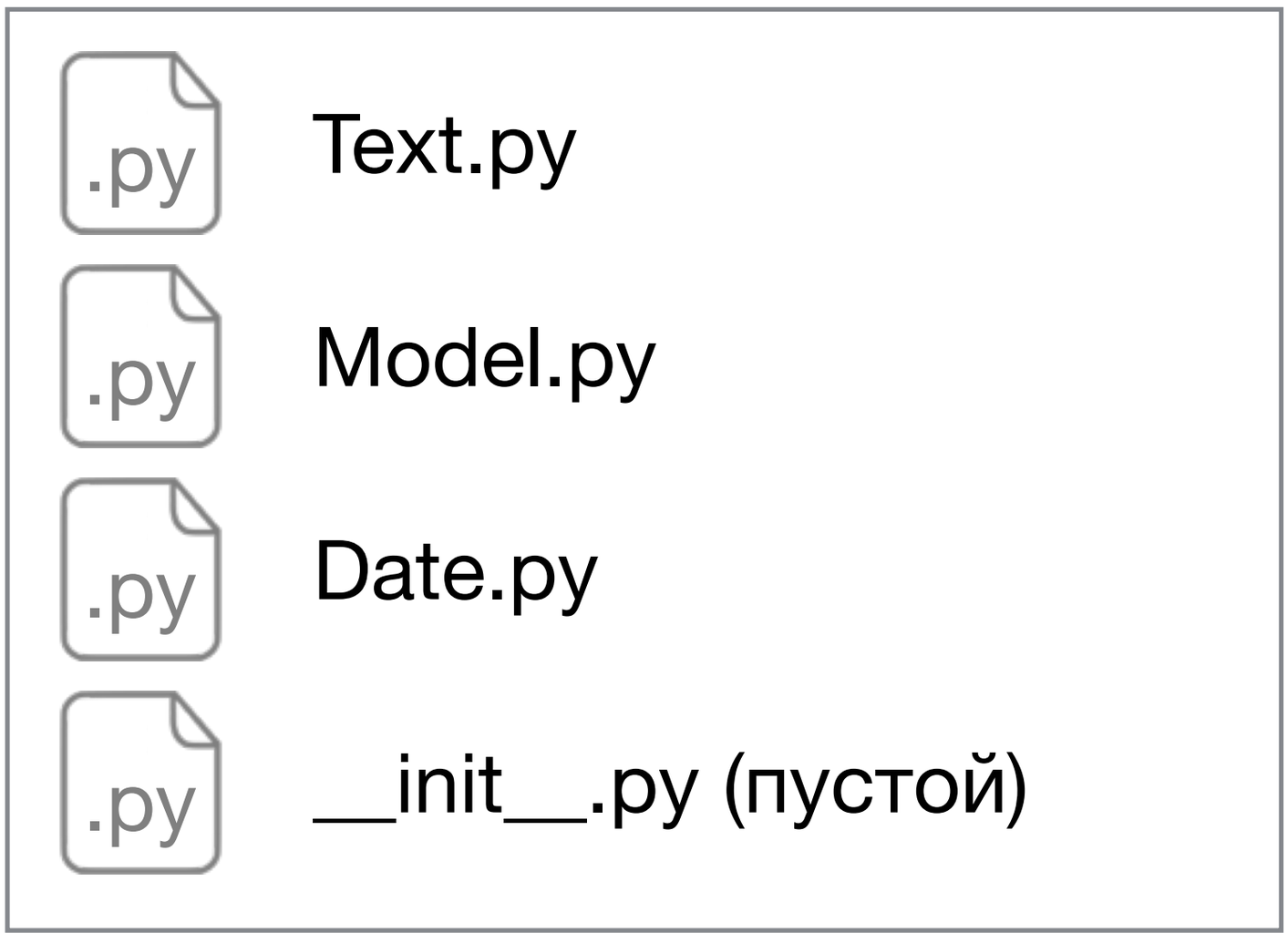
~/project/Utils




Общий пакет утилит

# Пакеты

~/project/Utils




Общий пакет утилит

# Пакеты

```
0: import Utils.Text as Text
1: from Utils import Model as M
```

# Приведение типов

# Приведение типов

| Тип | Название |
|---|---|
| Целые числа (Integer) | int |
| Рациональные числа (float) | float |
| Строки (string) | str |
| Списки (list) | list |
| Кортежи (tuple) | tuple |
| Словари (dictionary) | dict |
| Множества (set) | set |

# Приведение типов

```
0: str(20) == ”20”
1: int(”20”) == 20
2: float(20) == 20.0
3: list(”20”) == [”2”, ”0”]
4: tuple([”2”, ”0”]) == (”2”, ”0”)
5: set([”2”, ”0”]) == {”2”, ”0”}
6: dict({”a”: ”b”}) == {”a”: ”b”}
```

# Приведение типов

```
>>>list(42)
```

# Приведение типов

```
>>>list(42)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'int' object is not iterable
```

# Приведение типов

```
>>>list(42)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'int' object is not iterable
```

# Приведение типов

```
>>>list(42)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'int' object is not iterable
```

```
>>>int([1,2,3])
Traceback (most recent call last):
   File "<stdin>", line 1, in <module>
TypeError: int() argument must be a string, a bytes-like
object or a number, not 'list'
```

# Методы типов данных

## Строки

# Методы типов данных

## Строки

### lower и upper

```
0: ”I am the King of the Lizards”.lower()
```

# Методы типов данных

## Строки

### lower и upper

```
0: ”I am the King of the Lizards”.lower()
```

```
0: ”i am the king of the lizards”
```

# Методы типов данных

## Строки

### lower и upper

```
0: ”I am the King of the Lizards”.lower()
```

```
0: ”i am the king of the lizards”.upper()
```

# Методы типов данных

## Строки

### lower и upper

```
0: "I am the King of the Lizards".lower()
0: "i am the king of the lizards".upper()
0: "I AM THE KING OF THE LIZARDS"
```

# Методы типов данных

## Строки

### lower и upper

```
0: "I am the King of the Lizards".lower()
0: "i am the king of the lizards".upper()
0: "I AM THE KING OF THE LIZARDS"
```

# Методы типов данных

## Строки

### join

```
0: " ".join(["a", "b"]) # "a b"
```

# Методы типов данных

## Строки

### join

```
0: ” ”.join([”a”, ”b”]) # ”a b”
1: ”, ”.join([”a”, ”b”]) # ”a, b”
```

# Методы типов данных

## Строки

### join

```
0: ” ”.join([”a”, ”b”]) # ”a b”
1: ”, ”.join([”a”, ”b”]) # ”a, b”
2: ” and ”.join([”a”, ”b”]) # ”a and b”
```

# Методы типов данных

## Строки

### format

```
0: "Hello, {0}".format("world!") # "Hello, world!"
```

# Методы типов данных

## Списки

### append

```
01: ['a'].append('b') # ['a', 'b']
```

### extend

```
01: lst = ['a']
02: lst.extend(['b'])
03: print(lst) # ['a', 'b']
```

# Методы типов данных

## Списки

### sort

```
01: lst = [4, 1, 3, 2]
02: lst.sort()
03: print(lst) # [1, 2, 3, 4]
```

# Файловая система

# Файловая система

~/project

# Файловая система

```
0: with open(”./text.txt”, ”r”) as text_file:
1:     content = text_file.read()
2:     print(content)
```

# Файловая система

**Путь до файла**

```
0: with open(”./text.txt”, ”r”) as text_file:
1:     content = text_file.read()
2:     print(content)
```

# Файловая система

**Режим открытия**

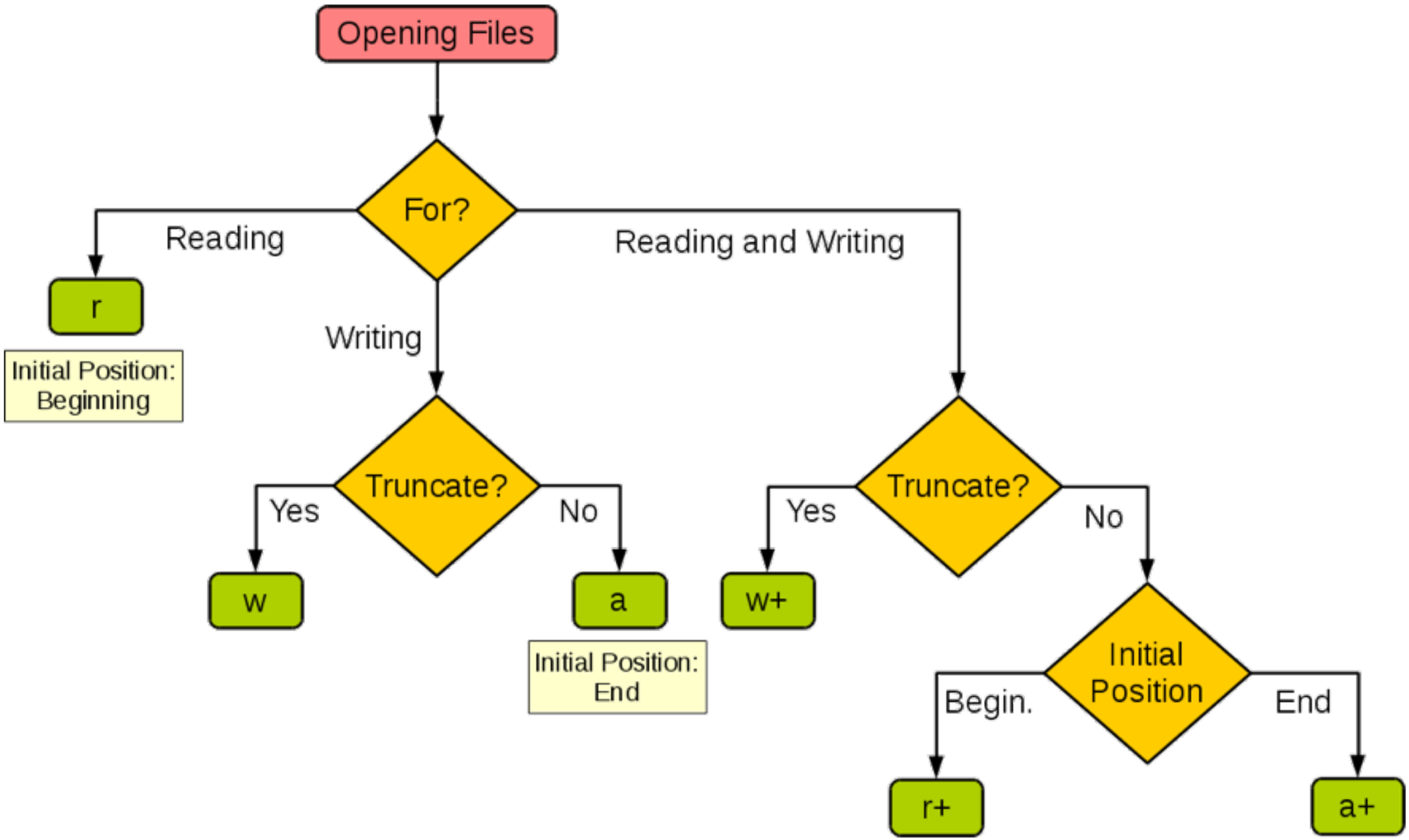
```
0: with open(”./text.txt”, ”r”) as text_file:
1:     content = text_file.read()
2:     print(content)
```

# Файловая система

```
0: with open(”./text.txt”, ”r”) as text_file:
1:     content = text_file.read()
2:     print(content)
```

| Режим | Описание |
|---|---|
| r | Открыть файл на чтение, указатель в начале файла |
| r+ | Открыть на чтение и запись, указатель в конце файла |
| w | Записать пустой файл, если существует, иначе то создать файл |
| w+ | Создать файл на чтение и запись. Указатель в начале файла |
| a | Открыть или создать файл на чтение и запись |
| a+ | Как a, но указать в конце файла |

# Файловая система



# Чтение из файла

```
01: with open(”./text.txt”, ”r”) as text_file:
01:     content = text_file.read()
02:     print(content)
```

# Запись в файл

```
01: with open(”text.txt”, ”a”) as text_file:
01:     text_file.write(”Hello\n”)
```

# Файлы и модуль os

```
01: import os
```

# Файлы и модуль os

## удаление файла

```
01: import os
02:
03: os.remove('text.txt')
```

# Файлы и модуль os

## удаление файла и директории

```
01: import os
02:
03: os.remove('text.txt')
04: os.rmdir('files')
```

# Файлы и модуль os

## проверка существования

```
01: import os
02:
03: os.path.exists(‘text.txt')
```