Числа

Используются только целые числа. С ними можно выполнять обычные арифметические операции: сложение, вычитание, умножение, целочисленное деление.

Строки

Строковая константа в Mython — это последовательность произвольных символов, размещающаяся на одной строке и ограниченная двойными кавычками " или одинарными '. Поддерживается экранирование спецсимволов '\n', '\t', '\" и '\'"

Логические константы и None

Язык поддерживает логические значения `True` и `False`. Есть также специальное значение `None`, аналог nullptr в C++

Комментарии

Доступны однострочные комментарии, начинающиеся с символа #. Весь следующий текст до конца текущей строки игнорируется. # внутри строк считается обычным символом.

Идентификаторы

Идентификаторы для обозначения имён переменных, классов и методов формируются так же, как в большинстве других языков программирования: начинаются со строчной или заглавной латинской буквы либо с символа подчёркивания. Потом следует произвольная последовательность, состоящая из цифр, букв и символа подчёркивания.

Типизация

Используется динамическая типизация. Тип каждой переменной определяется во время исполнения программы и может меняться в ходе её работы.

Классы

В Mython можно определить свой тип, создав класс. Как и в C++, класс имеет поля и методы, но, в отличие от C++, поля не надо объявлять заранее. Объявление класса начинается с ключевого слова `class`, за которым следует идентификатор имени и объявление методов класса.

Важные отличия классов в Mython от классов в C++:

- Специальный метод `__init__` играет роль конструктора — он автоматически вызывается при создании нового объекта класса. Метод `__init__` может отсутствовать.
- Неявный параметр всех методов — специальный параметр `self`, аналог указателя this в C++. Параметр `self` ссылается на текущий объект класса.
- Поля не объявляются заранее, а добавляются в объект класса при первом присваивании. Поэтому обращения к полям класса всегда надо начинать с `self.`, чтобы отличать их от локальных переменных.

- Все поля объекта — публичные.

Операции

В Mython определены:

- Арифметические операции для целых чисел, деление выполняется нацело. Деление на ноль вызывает ошибку времени выполнения.
- Операция конкатенации строк, например: `s = 'hello, ' + 'world'`.
- Операции сравнения строк и целых чисел ==, !=, <=, >=, <, >; сравнение строк выполняется лексикографически.
- Логические операции `and`, `or`, `not`.
- Унарный минус.

Приоритет операций (в порядке убывания приоритета):

- Унарный минус.
- Умножение и деление.
- Сложение и вычитание.
- Операции сравнения.
- Логические операции.

Порядок вычисления выражений может быть изменён скобками.

В Mython операция сложения кроме чисел и строк применима к объектам классов со специальным методом `__add__`

```class Fire: def **init**(obj): self.obj = obj

def **str**(): return "Burnt " + str(self.obj)

class Tree: def **str**(): return "tree"

class Matches: # Спички # операция сложения спичек с другими объектами превращает их в огонь def **add**(smth): return Fire(smth)

result = Matches() + Tree() print result # Выведет Burnt tree print Matches() + result # Выведет Burnt Burnt tree```

# Функция str

Функция `str` преобразует переданный ей аргумент в строку. Если аргумент — объект класса, она вызывает у него специальный метод `__str__` и возвращает результат. Если метода `__str__` в классе нет, функция возвращает строковое представление адреса объекта в памяти.

# Команда print

Специальная команда `print` принимает набор аргументов, разделённых запятой, печатает их в стандартный вывод и дополнительно выводит перевод строки.

Команда `print` вставляет пробел между выводимыми значениями. Если ей не передать аргументы, она просто выведет перевод строки. Чтобы преобразовать каждый свой аргумент в строку, команда `print` вызывает для него функцию `str`
```

Условный оператор

В Mython есть условный оператор. Его синтаксис:

```
if <условие>: <действие 1> <действие 2> ... <действие N> else: <де
йствие 1> <действие 2> ... <действие M>
```

<условие> — это произвольное выражение, за которым следует двоеточие. Если условие истинно, выполняются действия под веткой `if`, если ложно — действия под веткой `else`. Наличие ветки `else` необязательно.

<условие> может содержать сравнения, а также логические операции `and`, `or` и `not`. Условие будет истинным или ложным в зависимости от того, какой тип имеет вычисленное выражение.

Если результат вычисления условия — значение логического типа, для проверки истинности условия используется именно оно.

Если результат вычисления условия — число, условие истинно тогда и только тогда, когда это число не равно нулю, как в C/C++.

Если результат вычисления условия — строка, условие истинно тогда и только тогда, когда эта строка имеет ненулевую длину.

Если результат вычисления условия — объект класса, условие истинно.

Если результат вычисления условия — `None`, условие ложно.

Действия в ветках `if` и `else` набраны с отступом в два пробела. В отличие от C++, в

котором блоки кода обрамляются фигурными скобками, в языке Mython команды объединяются в блоки отступами. Один отступ равен двум пробелам. Отступ в нечётное количество пробелов считается некорректным.

Наследование

В языке Mython у класса может быть один родительский класс. Если он есть, он указывается в скобках после имени класса и до символа двоеточия.

Наследование в Mython работает так же, как в C++, — все методы родительского класса становятся доступны классу-потомку. При этом все методы публичные и виртуальные.

Методы

Методы в Mython имеют синтаксис:

```
def <имя метода>(<список параметров>): <действие 1> <действие 2> .
.. <действие N>
```

Ключевое слово `def` располагается с отступом в два пробела относительно класса. Инструкции, составляющие тело метода, имеют отступ в два пробела относительно ключевого слова `def`.

Как и в случае полей класса, обращения к полям и методам текущего класса надо

начинать с `self`.

Команда `return` завершает выполнение метода и возвращает из него результат вычисления своего аргумента. Если исполнение метода не достигает команды `return`, метод возвращает `None`.

Семантика присваивания

Mython — это язык с динамической типизацией, поэтому операция присваивания имеет семантику не копирования значения в область памяти, а связывания имени переменной со значением. Как следствие, переменные только ссылаются на значения, а не содержат их копии. Говоря терминологией C++, переменные в Mython — указатели. Аналог nullptr — значение `None`.

Прочие ограничения

Результат вызова метода или конструктора в Mython — терминальная операция. Её результат можно присвоить переменной или использовать в виде параметра функции или команды, но обратиться к полям и методам возвращённого объекта напрямую нельзя.