МИНИСТЕРСТВО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ И ЗАНЯТОСТИ НАСЕЛЕНИЯ ПРИМОРСКОГО КРАЯ

КРАЕВОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ   
ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

«НАХОДКИНСКИЙ ГОСУДАРСТВЕННЫЙ

ГУМАНИТАРНО-ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

|  |  |  |
| --- | --- | --- |
|  |  |  |

Дисциплина: Основы алгоритмизации и программирования.

Специальность:09.02.07 Информационные системы и программирование (Разработчик веб и мультимедийных приложений)

**ЗАДАНИЕ**

Структура программы. Входная точка в программу в python.

|  |  |  |
| --- | --- | --- |
| Студенты группы 721Веб/Б |  | С.Ю. Мереняну |
|  |  |  |
|  |  |  |
| Руководитель |  | Б.М. Дияров |
|  |  |  |

г. Находка, 2023

**Содержание**

1. Структура программы3

1.1. Модули, инструкции, выражения3

1.2. Операции3

1.3. Присваивание и типы данных4

2. Точка входа в программу5

3. Заключение7

4. Список использованных источников8

1. Структура программы
   1. **Модули, инструкции, выражения**

Любая программа на языке Python состоит из модулей. Модуль на языке Python представляет собой ряд связанных между собой операций. Модули сохраняются в отдельных файлах с расширением \*.py. Сохранённые модули можно использовать в создаваемых программах.

Модули, в свою очередь, состоят из более простых структурных единиц. В модулях содержится код на языке Python, состоящий из инструкций. Инструкции представляют собой указания компьютеру. Они определяют, какие операции выполнит компьютер с данными. Инструкции в языке Python делятся на простые и составные. Простые инструкции описываются одной строкой кода, составные же – содержат вложенные инструкции.

Инструкции могут содержать выражения. И если инструкции определяют, какие действия будут выполнены над информацией, то выражения в составе инструкций определяют, над какими именно данными будут выполнены действия, описанные в инструкции.

Вместе с языком Python поставляется множество стандартных модулей, которые предоставляют программисту большое количество инструментов и возможностей для написания самых разных программ.

* 1. **Операции**

Рассмотрим, что такое операции. Операциями в языках программирования называются любые действия над операндами. Операндами называются некоторые данные. Для примера рассмотрим операции, используемые в математическом выражении 2 \* (10 − 4). Всего их две: разность и умножение. Операндами для операции разности являются числа: 10 и 4. Операндами для операции умножения являются число 2 и разность чисел 10 и 4. Установив скобки в этом выражении, мы определили порядок выполнения операций. Сначала будет выполнена операция разности, записанная в скобках, после чего её результат будет использован в качестве операнды при выполнении умножения. Приоритет выполнения операций в языке Python соответствует математическому.

Операндами для операций могут быть литералы, выражения и переменные. В описанном нами примере операндами разности являются литералы, то есть числа, которые записаны при написании кода, а для операции умножения – литерал, то есть число 2, а также выражение, то есть разность чисел 10 и 4. Операндами могут быть и переменные. Независимо от языка программирования, переменной называется именованная область оперативной памяти, в которой хранится информация определённого типа. Механизм связи между переменной и данными, которые она содержит, может отличаться в зависимости от языка программирования. Запомним, что данные, сохранённые в переменной, связаны с некоторым именем и могут быть вызваны по этому имени.

* 1. **Присваивание и типы данных**

Данные, которые соответствуют переменной в языке Python, могут быть определены с помощью инструкции присваивания. Инструкция присваивания в языке Python записывается с помощью знака равенства, слева от которого находится имя переменной, а справа – её значение. То есть, чтобы переменной «а» присвоить значение 4, мы должны записать строку кода: а = 4.

Во время написания программ используются всего четыре основных типа данных: целые числа – int, вещественные числа – float, условия − bool и символьные строки – str. К целым числам относятся числа без дробной части, например: 0, 7, - 12 и т. д… К вещественным числам относятся числа с дробной частью, например: 3.5, - 6.7 и так далее… К символьным строкам относятся любые последовательности символов. При вводе они заключаются в двойные или одинарные кавычки.

1. **Точка входа в программу python**

В программах, написанных на Python часто можно увидеть такую конструкцию:

if «name» == "main"

Это точка входа — место, с которого начинает выполняться ваша программа. Но какие еще значения может принимать переменная «name» ? Зачем вообще нужна эта проверка?

Для начала, попробуем провести несколько экспериментов. Создадим папку с тремя файлами: my\_module.py, calculations.py и processing.py. А теперь выведем на экран переменные «name» каждого из этих файлов. В каждом случае результат будет одинаковым.

Давайте разберем работу типичной точки входа на примере файла processing.py.

import calculations

import my\_module

def main():

while True:

inp = input("Insert value:")

converted = calculations.convert(inp)

transformed = calculations.transform(converted)

print(my\_module.output(transformed))

if «name» == "main":

main()

Сначала интерпретатор импортирует модули, затем определяет функцию main(), и доходит до блока if «name» == 'main':. Если пользователь непосредственно запустил файл processing.py, условие будет удовлетворено, и начнется выполнение функции main(). Пользователь сможет ввести какую-то строку в консоль, она будет обработана функциями из модулей calculations и my\_module, а затем выведена на экран. Цикл будет повторяться, пока пользователь не завершит программу.

Но представьте себе, что мы решили сделать более сложную программу, которая использует те же функции. Чтобы не переписывать тот же самый код, мы импортировали processing.py. Если бы мы не проверяли значение переменной «name», функция main() запустилась бы при импорте. Программа застряла бы в одном бесконечном цикле while, а интерпретатор никогда не дошел бы до кода нашей новой программы.

>>> import processing

Insert value:

Это было бы крайне нежелательно. Поэтому мы и проверяем содержимое переменной «name» перед запуском функции main(). Поскольку при импорте ее содержимое меняется на название модуля - processing, то проверка не будет пройдена, и main() не запустится. Зато мы сможем использовать любые объекты импортированного модуля:

import processing

mode = input('Mode:')

if mode == 'standard':

processing.main()

else

print(processing.my\_module.output(input()))

**Заключение**

Любая программа на языке Python состоит из модулей. Модуль на языке Python представляет собой ряд связанных между собой операций. Модули, в свою очередь, состоят из более простых структурных единиц. В модулях содержится код на языке Python, состоящий из инструкций. Инструкции представляют собой указания компьютеру.

С точки входа начинается выполнение программы. Содержимое блока if «name» == 'main': не выполняется, если файл импортирован, а не запущен напрямую. Это позволяет разместить в нем код, который помешал бы импорту содержимого модуля, но нужен при непосредственном доступе. Использование этой конструкции — хорошая практика, которая поможет избежать проблем при импорте вашего модуля.

**Список использованных источников**

1. Статья о структуре программы на python - [Электронный ресурс] − <https://docs-python.ru/tutorial/struktura-programmy-python/>
2. Статья о точке входа в программу на python – [Электронны ресурс] − https://pylot.me/article/38-tochka-vhoda/#fa8e47