

Министерство образования Республики Беларусь  
Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей  
Кафедра информатики  
Дисциплина «Методы трансляции»

**ОТЧЕТ**  
к лабораторной работе № 1  
на тему «Определение модели языка. Выбор инструментальной  
языковой среды»

Выполнил

Я. Ю. Прескурел

Проверил

Н. Ю. Гриценко

Минск 2024

## СОДЕРЖАНИЕ

1	Постановка задачи.....	3
2	Определение модели языка программирования .....	4
2.1	Переменные и константы .....	4
2.2	Типы данных.....	4
2.3	Структуры данных .....	5
2.4	Типы операторов .....	6
2.5	Функции .....	8
2.6	Классы .....	8
2.7	Подключение библиотек .....	8
3	Определение инструментальной языковой среды.....	9
	Выводы.....	10
	Список использованных источников .....	11
	Приложение А (обязательное) Пример реализации программ на языке программирования Python.....	12

## **1 ПОСТАНОВКА ЗАДАЧИ**

Целью выполнения данной лабораторной работы является определить полное подмножество выбранного языка программирования, предоставить тексты двух или трех программ, включающих все элементы этого подмножества, а также определить инструментальную языковую среду, которая включает в себя язык программирования с указанием версии, на котором ведется разработка, операционная система, в которой выполняется разработка, и компьютер.

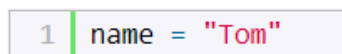
## 2 ОПРЕДЕЛЕНИЕ МОДЕЛИ ЯЗЫКА ПРОГРАММИРОВАНИЯ

### 2.1 Переменные и константы

Для хранения данных в программе на языке Python используются переменные, которые представляют собой именованные участки памяти.[1]

Для хранения данных в программе на языке Python используются переменные, которые представляют собой именованные участки памяти. Имя переменной в Python может содержать буквы, цифры и знак подчеркивания, начинаться должно с буквы или знака подчеркивания, и не может быть ключевым словом языка.

В общем случае определение переменной в языке программирования Python представлено на рисунке 2.1.

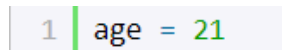


```
1 name = "Tom"
```

Рисунок 2.1 – Общий случай определения переменной

Язык программирования Python – это регистронезависимый язык, то есть регистр символов не имеет значения.

Нотация присваивания организовывается при помощи оператора присваивания переменной некоторого значения. Пример нотации присваивания на языке программирования Python представлен на рисунке 2.2.



```
1 age = 21
```

Рисунок 2.2 – Нотация присваивания на языке программирования Python

В языке программирования Python можно сразу инициализировать несколько переменных в одной строке через запятую.

Кроме переменных в языке программирования Python можно использовать константы. Хотя в самом языке нет отдельного ключевого слова для констант, общепринятой практикой является написание имен констант заглавными буквами, чтобы подчеркнуть их неизменность. Значение константы устанавливается один раз и не изменяется в последующем использовании.

### 2.2 Типы данных

В языке программирования Python существует разнообразие типов данных, каждый из которых определяет характеристики переменной, такие как диапазон значений, поддерживаемые операции и объем памяти, который она занимает. Ниже представлен обзор основных типов данных в Python:

- логический тип;
- целочисленные типы;
- тип чисел с плавающей точкой;
- строковые типы;
- тип комплексных чисел;
- списки;
- кортежи;
- множества;
- словари;
- байтовые строки;
- строки Unicode.

Каждый из этих типов данных предназначен для решения определенных задач и предоставляет различные возможности для работы с данными.[2]

Логический тип `bool` принимать одно из двух значений:

- `true`;
- `false`.

Логический тип в основном применяется в условных выражениях.

Целочисленные типы в языке программирования Python явно не разделены на знаковые и беззнаковые

Тип чисел с плавающей точкой нужен для представления вещественных чисел.

Строковые типы могут содержать символы, цифры и другие символы Unicode, представлены типом `str`.

Тип комплексных чисел позволяет работать с комплексными числами.

Списки представляют упорядоченные изменяемые коллекции.

Кортежи имеют схожесть со списками, но являются неизменяемыми.

Множества предоставляют неупорядоченные уникальные элементы.

Словари представляют коллекции пар ключ-значение.

Байтовые строки представляют последовательность байтов.

Строки Unicode позволяют работать с символами Unicode.

Типы данных в Python являются динамическими, что позволяет им изменяться в процессе выполнения программы. Python автоматически определяет тип переменной в момент присваивания значения.

## 2.3 Структуры данных

В Python существует множество структур данных, предоставляемых встроенными типами и библиотеками. К ним относятся:

- списки;
- кортежи;
- множества;
- словари;
- стек;

- очередь;
- массивы.

В Python массивы могут быть представлены списками, но для работы с числовыми данными также используется библиотека `array`.

Эти структуры данных предоставляют различные возможности для хранения, управления и обработки данных в Python. Они могут быть использованы в зависимости от конкретных задач и требований при разработке программ.

## 2.4 Типы операторов

В языке программирования Python существует множество операторов, позволяющих выполнять различные операции над данными. Операторы могут выполнять арифметические, логические, сравнительные операции.[3]

Арифметические операции производятся над числами. Значения, которые участвуют в операции, называются операндами. В языке программирования Python арифметические операции могут быть бинарными и унарными. К бинарным операциям относятся `+`, `-`, `*`, `/`, `%`.

Также есть две унарные арифметические операции, к которым относятся операция инкремента (`+=`) и операция декремента (`-=`).

К логическим операторам в языке программирования Python относятся логическое И (`and`), логическое ИЛИ (`or`), логическое НЕ (`not`).

Оператор `and` используется для проверки истинности обоих операндов, возвращая истину только в том случае, если оба операнда истинны. Если хотя бы один из операндов ложен, результат операции будет ложным.

Оператор `or` используется для проверки истинности хотя бы одного из операндов, возвращая истину, если хотя бы один из них истинен. Результат операции будет ложным только в том случае, если оба операнда ложны.

Оператор `not` используется для инвертирования логического значения операнда, то есть если операнд истинен, то оператор вернет ложь, а если операнд ложен, то вернет истину.

Операторы сравнения используются для сравнения значений двух операндов и возвращают логическое значение в зависимости от результата сравнения. К операторам сравнения относятся:

- оператор равенства (`==`);
- оператор неравенства (`!=`);
- оператор больше (`>`);
- оператор меньше (`<`);
- оператор больше или равно (`>=`);
- оператор меньше или равно (`<=`).

Данный тип операторов используется в условных выражениях.

К побитовым операторам относятся:

- побитовый И (`&`);
- побитовый ИЛИ (`|`);

- побитовое исключающее ИЛИ (^);
- побитовое отрицание (~);
- побитовый сдвиг влево (<<);
- побитовый сдвиг вправо (>>).

Операции присваивания позволяют присвоить некоторое значение переменной. Эти операции выполняются над двумя операндами. К операторам присваивания относятся:

- базовая операция присваивания (=);
- присваивание после сложения (+=);
- присваивание после вычитания (-=);
- присваивание после умножения (\*=);
- присваивание после деления (/=);
- присваивание результата целочисленного деления (//=);
- присваивание степени числа (\*\*=);
- присваивание остатка от деления (%=).

Операторы цикла в языке программирования Python позволяют выполнять повторяющиеся действия в течении определенного количества итерация

К операторам цикла языка программирования Python относятся:

- цикл for;
- цикл while.

Цикл for используется для выполнения блока кода заданное количество раз.

Цикл while используется для выполнения блока кода до тех пор, пока условие истинно, условие проверяется до выполнения каждой итерации.

Условные операторы в языке программирования Python используются для выполнения различных действий в зависимости от истинности или ложного заданного условия. К основным условным операторам относятся:

- оператор if;
- операторы elif;
- оператор else;
- тернарный оператор.

Оператор if используется для выполнения блока кода, если условие истинно.

Оператор if-else используется для выполнения одного блока кода, если условие истинно, а другого блока кода, если условие ложно.

Оператор if-elif используется для проверки нескольких условий последовательно, если одно из условий истинно, соответствующий блок кода выполняется, и выполнение оператора завершается.

Тернарный оператор предоставляет удобный способ выбора одного из двух возможных вариантов действий на основе значения логического выражения. Синтаксис тернарного условного оператора представлен на рисунке 2.5.

```
result = x if x > y else y
```

Рисунок 2.3 – Синтаксис тернарного условного оператора

Вышеперечисленные условные операторы позволяют в зависимости от выполнения условий выбирать различные пути выполнения программы и принимать различные решения.

## 2.5 Функции

Функция определяет действия, которые выполняет программа. Функции позволяют выделить набор инструкций и назначить ему имя. А затем многократно по присвоенному имени вызывать в различных частях программы.[4]

По сути функция – это именованный блок кода. Первая строка представляет заголовок функции. Для возвращения результата функция применяет оператор `return`.

## 2.6 Классы

В языке программирования Python присутствует поддержка объектно-ориентированного программирования, и для определения собственных типов данных, которые могут содержать как данные, так и методы, используется ключевое слово `class`. [5]

В языке Python переменная класса называется экземпляром класса. Помимо хранения данных, классы могут содержать и функции. Функции, определенные внутри класса, называются методами.

## 2.7 Подключение библиотек

В языке программирования Python для подключения библиотек используется оператор `import`. Этот оператор позволяет включать содержимое модулей или библиотек в программу перед выполнением.

Есть также возможность подключить конкретные функции или объекты из библиотеки с помощью ключевого слова `from`.



### **3 ОПРЕДЕЛЕНИЕ ИНСТРУМЕНТАЛЬНОЙ ЯЗЫКОВОЙ СРЕДЫ**

Для создания транслятора был выбран язык программирования C# 12.0. C# – это современный и мощный язык программирования, предназначенный для разработки приложений на платформе .NET. Он отличается высокой производительностью, широкими возможностями и четким синтаксисом, что упрощает процесс разработки.

Одним из ключевых преимуществ C# является управление памятью через среду выполнения .NET, что освобождает разработчика от необходимости явно управлять памятью. Это способствует упрощению кода, повышению производительности и уменьшению вероятности ошибок.

C# также предлагает статическую строгую типизацию, что означает, что типы данных определяются на этапе компиляции, что помогает выявлять ошибки на ранних этапах разработки. Язык поддерживает объектно-ориентированное программирование, что способствует созданию чистого и модульного кода.

Платформа .NET обеспечивает кроссплатформенность, позволяя запускать приложения, написанные на C#, на различных операционных системах. Это делает C# удобным выбором для разработки приложений, которые должны работать на различных платформах.

В качестве интегрированной среды разработки был выбран Visual Studio от Microsoft.

Операционной системой выступает Windows. Работа проводится на PC.

## **ВЫВОДЫ**

В ходе выполнения данной лабораторной работы была проведена детальная классификация и определение ключевых аспектов модели языка программирования Python. В результате проведенного анализа были описаны переменные и константы языка программирования Python, типы данных, структуры данных, типы операторов и подключение библиотек. В результате определения модели языка было определено, что язык программирования Python предоставляет широкие возможности для работы с различными типами и структурами данных, а также операторами.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] Введение в Python [Электронный ресурс]. – Режим доступа: <https://metanit.com/python/tutorial/1.1.php>. – Дата доступа: 15.02.2024.
- [2] Типы данных [Электронный ресурс]. – Режим доступа: <https://metanit.com/python/tutorial/2.2.php>. – Дата доступа: 15.02.2024.
- [3] Операторы в Python [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/ru-ru/cpp/c-operators>. – Дата доступа: 16.02.2024
- [4] Функции Python [Электронный ресурс]. – Режим доступа: <https://metanit.com/python/tutorial/2.8.php>. – Дата доступа: 16.02.2024.
- [5] Классы Python [Электронный ресурс]. – Режим доступа: <https://metanit.com/python/tutorial/7.1.php>. – Дата доступа: 16.02.2024.

## ПРИЛОЖЕНИЕ А

### (обязательное)

#### Пример реализации программ на языке программирования Python

##### Листинг 1 – Программная реализация сортировки пузырьком

```
def bubble_sort(arr):
    n = len(arr)
    for i in range(n - 1):
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

arr = [64, 25, 12, 22, 11]
bubble_sort(arr)

print("Sorted array:", arr)
```

##### Листинг 2 – Программная реализация поиска подстроки в строке

```
def search_pattern(text, pattern):
    n = len(text)
    m = len(pattern)
    pos = -1

    for i in range(n - m + 1):
        j = 0
        while j < m:
            if text[i + j] != pattern[j]:
                break
            j += 1

        if j == m:
            pos = i
            break

    return pos

text = "Hello, world!"
pattern = "world"

result = search_pattern(text, pattern)

if result != -1:
    print("Pattern found at position:", result)
else:
    print("Pattern not found.")
```