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.**

Студент А. В. Скворцов

Преподаватель Н. Ю. Гриценко

Минск 2024

СОДЕРЖАНИЕ

[1 Цель работы 3](#_Toc158333536)

[2 Подмножество языка программирования 4](#_Toc158333537)

[2.1 Общее описание 4](#_Toc158333538)

[2.2 Числовые и строковые константы 4](#_Toc158333539)

[2.3 Операторы цикла 4](#_Toc158333540)

[2.4 Условные операторы 5](#_Toc158333541)

[2.5 Структуры данных 6](#_Toc158333542)

[3 Инструментальная языковая среда 7](#_Toc158333543)

[Заключение 8](#_Toc158333544)

[Список использованных источников 9](#_Toc158333545)

[Приложение А (обязательное) Листинг кода 10](#_Toc158333546)

1. **ЦЕЛЬ РАБОТЫ**

Необходимо определить подмножество языка программирования (типы констант, переменных, операторов и функций). В подмножество должны быть включены числовые и текстовые константы, 3-4 типа переменных, операторы цикла, условные операторы, структуры данных, подключение библиотек.

1. **ПОДМНОЖЕСТВО ЯЗЫКА ПРОГРАММИРОВАНИЯ**

## 2.1 Общее описание

В качестве подмножества языка программирования был выбран язык программирования Python.

Python – это объектно-ориентированный, интерпретируемый и высокоуровневый язык с простым синтаксисом и мощными функциями. Python предоставляет возможность создавать сложные приложения с меньшими усилиями и большей удобочитаемостью. Python широко применяется в разработке искусственного интеллекта и машинного обучения [1].

## Числовые и строковые константы

В Python существует несколько примеров числовых констант, таких как целочисленные константы (42), константы с плавающей точкой (3.14), константы, представляющие собой число, умноженное на 10 в степени (1e9). Также в Python имеются примеры и строковых констант, заключенных в кавычки (“Hello, World!”).

## 2.3 Операторы цикла

В Python есть несколько операторов цикла, которые позволяют выполнять повторяющиеся действия.

1. Цикл *while* выполняется пока выполняется условие цикла (рисунок 1).

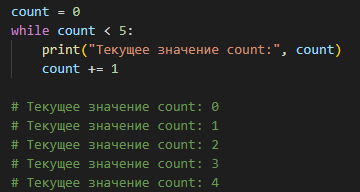


Рисунок 1 – Цикл while

1. Цикл *for* выполняется пока проходит по коллекцию, ключевое слово *continue* пропускает итерацию цикла (рисунок 2).

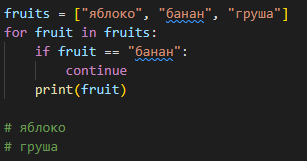


Рисунок 2 – Цикл *for* и ключевое слово *continue*

1. Ключевое слово *break* прерывает выполнение цикла (рисунок 3).

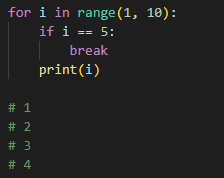


Рисунок 3 –Ключевое слово *break*

## 2.4 Условные операторы

В Python есть несколько условных операторов, которые позволяют выполнять различные действия в зависимости от условий.

1 Условный оператор *if-else* работает по принципе, если не одно, то другое (рисунок 4).

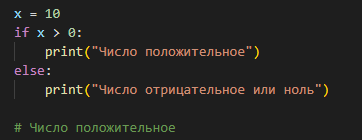


Рисунок 4 – Условный оператор *if-else*

2 Условный оператор *if-elif-else* работает так же, как и предыдущий, только с еще одним условием (рисунок 5).

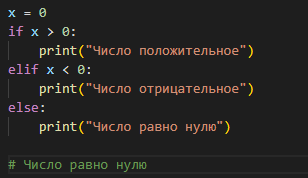


Рисунок 5 – Условный оператор *if-elif-else*

## 2.5 Структуры данных

В Python есть различные встроенные структуры данных, которые позволяют организовывать и хранить данные (рисунок 6).

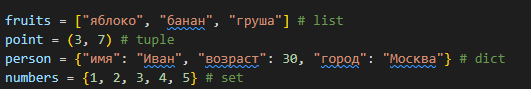


Рисунок 6 – Структуры данных Python

## 2.6 Типы данных

Python поддерживает динамическую типизацию, то есть тип переменной определяется только во время исполнения. Поэтому вместо «присваивания значения переменной» лучше говорить о «связывании значения с некоторым именем». Также Python поддерживает подсказки типов с помощью аннотаций, добавляющие поддержку последовательной типизации и позволяющие использовать статический анализ кода [2].

## 

1. **ИНСТРУМЕНТАЛЬНАЯ ЯЗЫКОВАЯ СРЕДА**

Выбранным языком программирования для разработки выбран Rust, версии 1.76.0, эта версия является последней на момент написания данного текста.

**Rust** – это системный язык программирования, разработанный Mozilla Research. Он был представлен в 2010 году и с тех пор приобрел популярность благодаря своей безопасности, производительности и современным инструментам разработки.

Одной из ключевых особенностей **Rust** является его система типов, которая обеспечивает безопасность памяти и предотвращает множество ошибок, связанных с памятью, таких как указатели на нулевые значения, чтение неинициализированных данных и гонки данных. Для достижения этой цели **Rust** использует инновационный механизм владения, который позволяет контролировать время жизни объектов и обеспечивает автоматическое освобождение памяти без необходимости сборки мусора.

**Rust** также предлагает богатый набор инструментов разработки, включая удобную систему управления пакетами (Cargo) и интегрированную систему сборки (build system), что делает процесс разработки более эффективным. Компилятор **Rust** обеспечивает строгую проверку типов, статический анализ и обширные сообщения об ошибках, что помогает разработчикам быстро находить и исправлять проблемы в коде [3].

Для анализа был выбран язык программирования Python версии 3.10. Операционной системой является Linux, дистрибутив Ubuntu 22.04 и так как этот дистрибутив использует в своем ядре Python 3.10, то и для анализа используется именно эта версия, а не последняя. Компьютером для разработки является обычный стационарный персональный компьютер, собранный на базе Intel.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы было определено подмножество языка Python. Была рассмотрена типизация данного языка программирования и так как она является динамической, типы данных имеют скорее удобный для чтения кода смысл. Также были учтены основные конструкции языка, такие как условные операторы, операторы цикла, представлены некоторые виды встроенных структур данных языка Python.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Python Basics: What makes Python so Powerful? [Электронный ресурс]. – Режим доступа: https://medium.com/edureka/python-basics-f371d7fc0054. – Дата доступа: 08.02.2024.
2. Python [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Python. – Дата доступа: 08.02.2024.
3. Rust [Электронный ресурс]. – Режим доступа: https://doc.rust-lang.org/cargo/index.html. – Дата доступа: 08.02.2024.

# ПРИЛОЖЕНИЕ А (обязательное) Листинг кода

Листинг 1 – lab1.py

count = 0

while count < 5:

print("Текущее значение count:", count)

count += 1

# Текущее значение count: 0

# Текущее значение count: 1

# Текущее значение count: 2

# Текущее значение count: 3

# Текущее значение count: 4

fruits = ["яблоко", "банан", "груша"]

for fruit in fruits:

if fruit == "банан":

continue

print(fruit)

# яблоко

# груша

for i in range(1, 10):

if i == 5:

break

print(i)

# 1

# 2

# 3

# 4

Листинг 2 – lab1\_2.py

x = 10

if x > 0:

print("Число положительное")

else:

print("Число отрицательное или ноль")

# Число положительное

x = 0

if x > 0:

print("Число положительное")

elif x < 0:

print("Число отрицательное")

else:

print("Число равно нулю")

# Число равно нулю

fruits = ["яблоко", "банан", "груша"] # list

point = (3, 7) # tuple

person = {"имя": "Иван", "возраст": 30, "город": "Москва"} # dict

numbers = {1, 2, 3, 4, 5} # set