Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Методы трансляции

Отчет по лабораторной работе №1

Определение модели языка. Выбор инструментальной языковой среды.

Выполнил:

студент гр. 953504 Басенко К. А.

Проверил:

Шиманский В.В.

Минск 2022

Содержание

1. Цель работы

Необходимо определить подмножество языка программирования (типы констант, переменных, операторов и функций). В подмножество как минимум должны быть включены:

- числовые и текстовые константы;

- 3-4 типа переменных;

- операторы цикла (**do**...**while**, **for**) ;

- условные операторы (**if**...**else, case**).

Определение инструментальной языковой среды, т.е. языка программирования и операционной системы для разработки включает:

- язык программирования с указанием версии, на котором ведётся разработка (напр. Python 3.6);

- операционная система (Windows, Linux и т.д.), в которой выполняется разработка;

- компьютер;

В отчете по лабораторной работе дается полное определение подмножества языка программирования, тексты 2-3-х программ, включающих все элементы этого подмножества. Приводится подробное описание инструментальной языковой среды.

2. Подмножество языка программирования

В качестве подмножества языка программирования выбран язык C#.

C# — объектно-ориентированный язык программирования. C# относится к семье языков с C-подобным синтаксисом, из них его синтаксис наиболее близок к C++ и Java. Язык имеет статическую типизацию, поддерживает полиморфизм, рефлексию, перегрузку операторов (в том числе операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, LINQ, исключения, комментарии в формате XML.

2.1. Числовые и строковые константы

- 0_0, -2, 1, 0, 0x3, 0b0101, 1_000, 2u, 2l, 3ul, (целочисленные).
- 3.5, 1_000f, -2.7, 2e10,-2E3,1E-10, 0.1f, 10.2m, 5.d, 10., .5, (вещественные литералы)
- ‘*a*’, ‘*f*’, ‘ ‘, ‘*/*’, ‘*ю*’, ‘*t*’, '*u0420*', ’*n*’ (символьные литералы)
- "", "*hello*", @”*t*”, “*”quotes\”*” (строковые литералы)
- true, false (логические литералы)

2.2. Типы переменных

C# поддерживает статическую типизацию, т.е. тип переменной определяется на этапе компиляции. В C# имеются встроенные типы: *bool*, *byte*, *sbyte*, *short*, *ushort*, *int*, *uint*, *long*, *ulong*, *float*, *double*, *decimal*, *char*, *string*, *object*.

Коллекции в C#: массив, список, кортежи, словарь, очередь, стек.

Все значения являются объектами, в том числе функции, методы, модули, классы.

2.3. Операторы цикла:

- while -- выполняет действие до тех пор, пока условие цикла истинно.

```
int num = 20;
while (num != 0)
{
    Console.Write(num % 10);
    num /= 10;
}
```

- do...while -- сначала выполняется код цикла, а потом происходит проверка условия в инструкции while. И пока это условие истинно, цикл повторяется.

Пример:

```
do
{
    if (counted[i])
    {
        res++;
    }
}
while (++i < counted.Length);
```

- *for* -- Объявление цикла for состоит из трех частей. Первая часть объявления цикла - некоторые действия, которые выполняются один раз до выполнения цикла. Обычно здесь определяются переменные, которые будут использоваться в цикле. Вторая часть - условие, при котором будет выполняться цикл. Пока условие истинно, будет выполняться цикл. И третья часть - некоторые действия, которые выполняются после завершения блока цикла. Эти действия выполняются каждый раз при завершении блока цикла.

Пример:

```
for (int i = 0; i < arr.Length; i++)
{
    for (int j = 0; j < arr[i].Length; j++)
    {
        Console.Write(arr[i][j] + ", ");
    }

    Console.WriteLine();
}
```

- *foreach* -- предназначен для перебора набора или коллекции элементов.

Пример:

```
foreach (char symbol in stringToCode)
{
    encodedString += symbol + 1;
}
```

- *break* -- прерывает исполнение цикла

Пример:

```
string input;
while (true)
{
    input = Console.ReadLine();
    if (IsValid(input))
    {
        break;
    }
}
```

- *continue* -- переходит на следующую итерацию цикла

Пример:

```
Stack<string> paramsAndArgs = new ();
int index;
foreach (var consoleArg in consoleArgs.Reverse())
{
    if ((index = consoleArg.IndexOf('=', StringComparison.Ordinal)) != -1)
    {
        paramsAndArgs.Push(consoleArg[(index + 1) ..]);
        paramsAndArgs.Push(consoleArg[..index]);
        continue;
    }

    paramsAndArgs.Push(consoleArg);
}
```

2.4. Условные операторы

- If..else

Пример:

```
if (age > 19)
{
    return "Adult";
}
else if (age > 13)
{
    return "Teenager";
}
else
{
    return "Kid";
}
```

- Switch

Пример:

```
switch (direction)
{
    case "left":
        {
            GoLeft();
            break;
        }
    case "right":
        {
            GoRigth();
            break;
        }
}
```

3. Инструментальная языковая среда

В качестве языковой среды выбран язык программирования Python. Разработка основана на работе с операционной системой Windows на PC.

Python — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода. Синтаксис ядра Python минималистичен. В то же время стандартная библиотека включает большой объём полезных функций.

Python поддерживает структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений, высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

Эталонной реализацией Python является интерпретатор CPython, поддерживающий большинство активно используемых платформ. Он распространяется под свободной лицензией Python Software Foundation License, позволяющей использовать его без ограничений в любых приложениях, включая проприетарные. Есть реализация интерпретатора для JVM с возможностью компиляции, CLR, LLVM, другие независимые реализации. Проект PyPy использует JIT-компиляцию, которая значительно увеличивает скорость выполнения Python-программ.

Примечание. Код программ

1. Нахождение простых чисел в диапазоне:

```
n: int = int(input("Enter number: "))
primes: List[int] =
    [2] +
    [x for x in range(3, n + 1, 2)
        if list(filter(lambda i: x/i == x//i, range(3, x + 1, 2)))[0] == x]

print(primes)
```

2. Сортировка слиянием:

```
import random

def merge(arr, l, m, r):
    n1 = m - l + 1
    n2 = r - m
    L = [0] * (n1)
    R = [0] * (n2)
    for i in range(0 , n1):
        L[i] = arr[l + i]
    for j in range(0 , n2):
        R[j] = arr[m + 1 + j]
    i = 0
    j = 0
    k = l
    while i < n1 and j < n2 :
        if L[i] <= R[j]:
            arr[k] = L[i]
            i += 1
        else:
            arr[k] = R[j]
            j += 1
        k += 1
    while i < n1:
        arr[k] = L[i]
        i += 1
        k += 1
    while j < n2:
        arr[k] = R[j]
        j += 1
        k += 1

def mergeSort(arr,l,r):
    if l < r:
        m = (l + r - 1) // 2
        mergeSort(arr, l, m)
        mergeSort(arr, m + 1, r)
        merge(arr, l, m, r)
```