Министерство образования Республики Беларусь  
Учреждение образования  
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

КАФЕДРА ИНФОРМАТИКИ

Отчёт по лабораторной работе No1

По теме “Определение модели языка. Выбор инструментальной языковой среды.”

Выполнила: студентка гр. 053503 Зырянова М.М.

Проверил: ассистент кафедры информатики Гриценко Н. Ю.

Минск 2023

**СОДЕРЖАНИЕ**

[1 ЦЕЛЬ РАБОТЫ 3](#_Toc125926526)

[2 ОПРЕДЕЛЕНИЕ ПОДМНОЖЕСТВА ЯЗЫКА ПРОГРАММИРОВАНИЯ 4](#_Toc125926527)

[2.1 Типы констант и переменных 4](#_Toc125926528)

[2.2 Типы операторов 5](#_Toc125926529)

[2.2.1 Арифметические операторы 5](#_Toc125926530)

[2.2.2. Побитовые операторы 6](#_Toc125926531)

[2.2.3. Операторы присваивания 6](#_Toc125926532)

[2.2.4. Условные операторы 7](#_Toc125926533)

[2.2.5. Операторы цикла 7](#_Toc125926534)

[2.3 Типы функций 8](#_Toc125926535)

[3 ИНСТРУМЕНТАЛЬНАЯ ЯЗЫКОВАЯ СРЕДА 10](#_Toc125926536)

[ПРИЛОЖЕНИЕ А 11](#_Toc125926537)

# **1 ЦЕЛЬ РАБОТЫ**

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

1. числовые и текстовые константы;
2. 3-4 типа переменных;
3. операторы цикла (do...while, for);
4. условные операторы (if...else, case).

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

1. язык программирования c указанием версии, на котором ведётся разработка (напр. Python 3.7);
2. операционная система (Windows, Linux и т.д.), в которой выполняется разработка;
3. компьютер (PC / Macintosh).

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 ОПРЕДЕЛЕНИЕ ПОДМНОЖЕСТВА ЯЗЫКА ПРОГРАММИРОВАНИЯ**

В качестве подмножества для работы был выбран язык программирования С++. Это компилируемый, статически типизированный язык программирования общего назначения. С++ имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и [алгоритмы](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC), ввод-вывод, регулярные выражения, поддержку [многопоточности](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B3%D0%BE%D0%BF%D0%BE%D1%82%D0%BE%D1%87%D0%BD%D0%BE%D1%81%D1%82%D1%8C) и другие возможности. C++ сочетает свойства как [высокоуровневых](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%81%D0%BE%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F" \o "Высокоуровневый язык программирования), так и [низкоуровневых языков](https://ru.wikipedia.org/wiki/%D0%9D%D0%B8%D0%B7%D0%BA%D0%BE%D1%83%D1%80%D0%BE%D0%B2%D0%BD%D0%B5%D0%B2%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F" \o "Низкоуровневый язык программирования).

C++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание [операционных систем](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0" \o "Операционная система), разнообразных прикладных программ, [драйверов](https://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B0%D0%B9%D0%B2%D0%B5%D1%80" \o "Драйвер) устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также компьютерных игр.

## **2.1 Типы констант и переменных**

Каждая переменная имеет определенный тип. И этот тип определяет, какие значения может иметь переменная, какие операции с ней можно производить и сколько байт в памяти она будет занимать. В C++ доступны следующие встроенные типы:

1. символьные: используются для представления символов в приложении. Переменная типа char представляет один символ в кодировке ASCII. Занимает в памяти 1 байт. Также существует тип wchar\_t, который представляет расширенный символ. На Windows он занимает в памяти 2 байта, на Linux - 4 байта;
2. целочисленные: используются для представления целых чисел в приложении. Существуют знаковые и беззнаковые целочисленные типы данных. К знаковым относятся: int (2 либо 4 байта в зависимости от архитектуры), short (2 байта), long (4 байта), long long (8 байт). Беззнаковые типы данных включают в себя только неотрицательные целые числа. К ним относятся: unsigned short (2 байта), unsigned int (2 либо 4 байта в зависимости от архитектуры), unsigned long (4 байта), unsigned long long (8 байт);
3. с плавающей точкой: используются для представления дробных чисел в приложении. Всего существует 3 типа данных с плавающей точкой: float, double, long double. Float представляет вещественное число ординарной точности с плавающей точкой в диапазоне. В памяти занимает 4 байта (32 бита). Double представляет вещественное число двойной точности с плавающей точкой в диапазоне. В памяти занимает 8 байт (64 бита). Long double представляет вещественное число двойной точности с плавающей точкой не менее 8 байт (64 бит). В зависимости от размера занимаемой памяти может отличаться диапазон допустимых значений.

Следует отметить, что С++ является статически типизированным языком программирования. То есть если для переменной определен какой-то тип данных, то в последующем этот тип изменить нельзя. Соответственно переменная может получить значения только того типа, который она представляет.

## **2.2 Типы операторов**

Оператор – это символ, который сообщает компилятору выполнить определенные математические или логические манипуляции. C ++ богат встроенными операторами, описанные в разделах ниже.

### **2.2.1 Арифметические операторы**

Арифметические операции производятся над числами. Значения, которые участвуют в операции, называются операндами. В языке программирования C++ арифметические операции бинарными (производятся над двумя операндами) и унарными (выполняются над одним операндом). К бинарным операциям относятся: +, -, \*, /, %. Пример сложения чисел:

*int x = 5, y = 7;*

*int z = x + y; //12*

Также есть две унарные арифметические операции, которые производятся над одним числом: ++ (инкремент) и – (декремент). Каждая из операций имеет две разновидности: префиксная и постфиксная. Пример постфиксного инкремента:

*int a = 8;*

*int b = a++; //8*

Пример префиксного декремента:

*int a = 8;*

*int b = --a; //7*

### **2.2.2. Побитовые операторы**

Побитовые операции выполняются над отдельными разрядами или битами чисел. Данные операции производятся только над целыми числами.

Каждое целое число в памяти представлено в виде определенного количества разрядов. И операции сдвига позволяют сдвинуть битовое представление числа на несколько разрядов вправо или влево. Операции сдвига применяются только к целочисленным операндам.

Побитовый сдвиг влево (<<) cдвигает битовое представление числа, представленного первым операндом, влево на определенное количество разрядов, которое задается вторым операндом. Пример:

*int a = 2 << 2; // 10 на два разрядов влево = 1000 – 8*

Побитовый сдвиг вправо (>>) сдвигает битовое представление числа вправо на определенное количество разрядов. Пример:

*int b = 16 >> 3; // 10000 на три разряда вправо = 10 – 2*

### **2.2.3. Операторы присваивания**

Операции присваивания позволяют присвоить некоторое значения. Эти операции выполняются над двумя операндами, причем левый операнд может представлять только модифицируемое именованное выражение, например, переменную. Базовая операция присваивания = позволяет присвоить значение правого операнда левому операнду. Например, x = 2.

Все остальные операции присваивания являются сочетанием простой операции присваивания с другими операциями:

1. +=: присваивание после сложения;
2. -=: присваивание после вычитания;
3. \*=: присваивание после умножения;
4. /=: присваивание после деления;
5. %=: присваивание после деления по модулю;
6. <<=: присваивание после сдвига разрядов влево;
7. >>=: присваивание после сдвига разрядов вправо;
8. &=: присваивание после поразрядной конъюнкции;
9. |=: присваивание после поразрядной дизъюнкции;
10. ^=: присваивание после операции исключающего ИЛИ.

Примеры:

*int a = 5 | 2; // 101 | 010 = 111 - 7*

*int b = 6 & 2; // 110 & 010 = 10 - 2*

*int c = 5 ^ 2; // 101 ^ 010 = 111 – 7*

*int d = ~9; // -10*

### **2.2.4. Условные операторы**

Условный оператор if – самый простой оператор для принятия решения. Он используется для определения того, будет ли выполняться определенный оператор или блок операторов, т.е. если определенное условие истинно, тогда блок оператора выполняется иначе нет.

*if (условие){*

*//инструкции;*

*}*

*else{*

*//инструкции;*

*}*

Операторы break, continue, return используются в **C++** для безусловного потока управления через функции в программе.

Break – это управляющий параметр цикла используется для завершения цикла. Как только оператор прерывания  встретится внутри цикла, итерации цикла прекращаются, и управление возвращается из цикла сразу же к первому оператору после окончания цикла.

Оператор continue противоположен оператору break, вместо завершения цикла он заставляет выполнить следующую итерацию цикла.

Return C++ возвращает поток выполнения функции, из которой он вызывается. Это утверждение не обязательно требует каких-либо условных операторов. Как только оператор выполняется, поток программы немедленно останавливается и возвращает элемент управления, откуда он был вызван.

### **2.2.5. Операторы цикла**

В С++ существует 3 вида операторов цикла:

1. цикл for (цикл с параметром): выполняется до тех пор, пока значение выражение равно true. Как только значение выражение станет false, выполнение цикла прекращается и выполняется оператор, который следует за циклом for;

*for (инициализация; выражение; прирост)*

*{*

*// последовательность операторов*

*}*

1. цикл while (с предусловием):  последовательность операторов выполняется до тех пор, пока условное выражение возвращает значение true. Как только выражение становится равным false, выполнение цикла while прекращается и управление передается следующему за циклом while оператору;

*while (*выражение*)*

*{*

*// последовательность операторов*

*}*

1. цикл do…while (с постусловием): цикл целесообразно использовать в случаях, когда итерацию нужно сделать хотя бы 1 раз. В отличие от циклов for и while, в цикле do…while условие проверяется при выходе из цикла (а не при входе в цикл). Сначала происходит выполнение тела цикла. Потом проверяется значение выражения (условное выражение). Если значение выражение есть истинна (true), выполняется снова тело цикла. Как только значение выражение станет false, выполнение цикла прекращается.

*do*

*{*

*// последовательность операторов*

*}*

*while (*выражение*);*

## **2.3 Типы функций**

В С++ существует два типа функций: библиотечные и пользовательские. Программисты могут использовать библиотечные функции, вызывая их напрямую, поэтому им не нужно писать их самим. Некоторые распространённые функции С++:

1. sqrt() – извлечение квадратного корня из числа;
2. abs() – возвращает абсолютную величину числа;
3. pow() – возведение числа в степень.

Зачастую программистами используются пользовательские функции. Пользовательская функция группирует код для выполнения определенной задачи, и этой группе кода дается имя (идентификатор). Когда функция вызывается из любой части программы, она выполняет коды, определенные в теле функции.

# **3 ИНСТРУМЕНТАЛЬНАЯ ЯЗЫКОВАЯ СРЕДА**

Для разработки интерпретатора был выбран язык программирования Python. Это высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык использует динамическую типизацию вместе с подсчётом ссылок и циклический сборщик мусора для менеджмента памяти. Также есть динамические разрешения имен (динамическое связывание), которые связывают имена методов и переменных во время выполнения программы.

В качестве интегрированной среды разработки был выбран Microsoft Visual Studio Code, операционной системой выступает Mac OS, вся работа производится на PC.

# **ПРИЛОЖЕНИЕ А**

**(обязательное)**

**Пример реализации программ на языке C++**

1. Решение задачи бинарного поиска элемента в отсортированном массиве.

*#include <iostream>*

*#include <vector>*

*using namespace std;*

*int main(){*

*int nums[] = {-1,0,3,5,9,12};*

*const int size = 6;*

*int target = 12;*

*int a = 0, b = size - 1;*

*while(a <= b){*

*int mid = (a + b) / 2;*

*if(nums[mid] == target){*

*cout << mid << endl;*

*return 0;*

*}*

*if(nums[mid] < target){*

*a = ++mid;*

*}*

*else{*

*b = --mid;*

*}*

*}*

*cout << -1 << endl;*

*return 0;*

*}*

1. Учитывая отсортированный массив различных целых чисел и целевое значение, верните индекс, если цель найдена. Если нет, верните индекс туда, где он был бы, если бы он был вставлен по порядку. Функция для решения задачи:

*int searchInsert(vector<int>& nums, int target) {*

*int right = nums.size(), left = 0, middle;*

*while(left < right){*

*middle=(left+right)/2;*

*if(nums[middle] < target){*

*left = middle + 1;*

*}*

*else if(nums[middle] > target){*

*right = middle;*

*}*

*else if(nums[middle] == target){*

*return middle;*

*}*

*}*

*return right;*

*}*