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             Е. А. Киселёва

Проверил                          Н. Ю. Гриценко

Минск 2024

СОДЕРЖАНИЕ

[1 Постановка задачи 3](#_Toc157960226)

[2 Определение модели языка программирования С++ 4](#_Toc157960227)

[2.1 Переменные и константы 4](#_Toc157960228)

[2.2 Типы данных 5](#_Toc157960229)

[2.3 Структуры данных 7](#_Toc157960230)

[2.4 Типы операторов 8](#_Toc157960231)

[2.5 Функции 10](#_Toc157960232)

[2.6 Классы 11](#_Toc157960233)

[2.7 Подключение библиотек 11](#_Toc157960234)

[3 Определение инструментальной языковой среды 12](#_Toc157960235)

[Выводы 13](#_Toc157960236)

[Список использованных источников 14](#_Toc157960237)

[Приложение А (обязательное) Пример реализации программ на языке   
                            программирования С++ 15](#_Toc157960238)

## 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 привести подробное описание инструментальной языковой среды.

## 2 ОПРЕДЕЛЕНИЕ МОДЕЛИ ЯЗЫКА             ПРОГРАММИРОВАНИЯ С++

В качестве подмножества языка программирования был выбран C++.

Язык программирования С++ представляет высокоуровневый компилируемый язык программирования общего назначения со статической типизацией, который подходит для создания самых различных приложений. На сегодняшний день С++ является одним из самых популярных и распространенных языков.

Программа на С++ состоит из набора инструкций. Каждая инструкция (statement) выполняет определенное действие. В конце инструкции в языке C++ ставится точка с запятой. Данный знак указывает компилятору на завершение инструкции.

## 2.1 Переменные и константы

Для хранения данных в программе на языке С++ используются переменные. Переменные представляют собой именованные области памяти, которые могут хранить значения различных типов данных. Они объявляются с указанием типа данных и имени переменной.

Имя переменной есть последовательность алфавитных-цифровых символов и знака подчеркивания. При этом имя переменной должно начинаться с алфавитного символа или подчеркивания.

Кроме того, в качестве имени переменной нельзя использовать ключевые слова языке C++. В общем случае переменная определяется в виде, представленном на рисунке 2.1.



Рисунок 2.1 – Общий случай определения переменной

Язык программирования С++ – это регистрозависимый язык,   
то есть регистр символов имеет большое значение.

После определения переменной можно присвоить некоторое значение. Присвоение переменной начального значения называется инициализацией.

В C++ есть три вида инициализации:

1 Нотация присваивания (assignment notation) – с помощью оператора присваивания (знак «равно», или =) переменной передаем некоторое значение

2 Функциональная нотация (functional notation) – после названия переменной в круглых скобках указывается ее значение

3 Инициализация в фигурных скобках (braced initialization) – после названия переменной в фигурных скобках указывается ее значение

В языке программирования С++ можно сразу инициализировать несколько переменных различными способами в одну строку через запятую.

Если переменную не инициализировать, то происходит ее инициализация по умолчанию. И переменная получает некоторое значение по умолчанию, которое зависит от места, где эта переменная определена.

Если переменная, которая представляет встроенный тип (например, тип int), определена внутри функции, то она получает неопределенное значение. Если переменная встроенного типа определена вне функции, то она получает то значение по умолчанию, которое соответствует ее типу.

Постоянные значения любого типа, наподобие чисел, символов, строк называются литералами.

Кроме переменных в языке программирования С++ можно определять константы. Их значение устанавливается один раз и не подлежит изменениям в будущем использовании. Константа определяется также, как и переменная, но в начале определения константы идет ключевое слово const. Если константа не будет инициализирована, то компилятор выведет ошибку и не сможет скомпилировать программу. [1]

## 2.2 Типы данных

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 языке С++ определены следующие базовые типы данных:

– логический тип;

– целочисленные типы;

– тип чисел с плавающей точкой;

– символьные типы;

– строковые типы.

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

Логический тип bool может хранить одно из двух значений: true (истинно, верно) и false (неверно, ложно). Этот тип как правило применяется в условных выражениях. Значением по умолчанию для переменных этого типа данных является значение false.

Целые числа в языке программирования С++ можно разделить на знаковые и беззнаковые типы.

К знаковыми типам относятся signed char, char, short, int, long, long long. К беззнаковым типам относятся unsigned char, unsigned short, unsigned int, unsigned long, unsigned long long.

Тип данных signed char представляет один символ и занимает в памяти 1 байт, тип данных unsigned char представляет также один символ и занимает 1 байт в памяти. Тип данных char представляет один символ в кодировке ASCII и занимает 1 байт.

Различие signed типов от unsigned заключается в том,   
что signed хранит любое значение из диапазона от -N до N, а unsigned хранит любое значение из диапазона от 0 до 2N-1.

Типы данных short, unsigned short занимают в памяти 2 байта. Тип данных int и unsigned int в зависимости от архитектуры могут занимать 2 байта или 4 байта. Тип данных long занимает в памяти 4 или 8 байт в зависимости от архитектуры. Тип данных unsigned long занимает в памяти 4 байта. Типы данных long long и unsigned long занимают в памяти 8 байт.

Для представления чисел в языке программирования С++ применяются целочисленные литералы со знаком или без него. Все целочисленные литералы по умолчанию представляют тип int. По умолчанию все стандартные целочисленные литералы представляют числа в десятичной системе. Однако C++ также позволяет использовать и числа в других системах исчисления. Чтобы указать, что число – шестнадцатеричное, перед числом указывается префикс 0x или 0X.

Для хранения дробных чисел в C++ применяются числа с плавающей точкой. Число с плавающей точкой состоит из двух частей: мантиссы и показателя степени. Оба могут быть как положительными, так и отрицательными. Величина числа – это мантисса, умноженная на десять в степени экспоненты. В качестве разделителя целой и дробной части используется символ точки. В языке С++ есть три типа для представления чисел с плавающей точкой float, double, long double.

Тип данных float представляет вещественное число одинарной точности с плавающей точкой и занимает в памяти 4 байта. Тип данных double представляет вещественное число двойной точности с плавающей точкой   
и занимает в памяти 8 байт. Тип данных long double представляет вещественное число двойной точности с плавающей точкой не менее 8 байт.

К символьным типа данных в языке программирования С++ относятся:

1 char: представляет один символ в кодировке ASCII. Занимает в памяти 1 байт (8 бит). Может хранить любое значение из диапазона от -128 до 127, либо от 0 до 255.

2 wchar\_t: представляет расширенный символ. На Windows занимает в памяти 2 байта (16 бит), на Linux - 4 байта (32 бита). Может хранить любой значение из диапазона от 0 до 65 535 (при 2 байтах), либо от 0 до 4 294 967 295 (для 4 байт).

3 char8\_t: представляет один символ в кодировке Unicode. Занимает в памяти 1 байт. Может хранить любой значение из диапазона от 0 до 256.

4 char16\_t: представляет один символ в кодировке Unicode. Занимает в памяти 2 байта (16 бит). Может хранить любой значение из диапазона от 0 до 65 535.

5 char32\_t: представляет один символ в кодировке Unicode. Занимает в памяти 4 байта (32 бита). Может хранить любой значение из диапазона от 0 до 4 294 967 295.

Кроме того, в С++ можно использовать специальные управляющие последовательности, которые предваряются знаком «\»   
и которые интерпретируются особым образом. Например, «\n» представляет перевод строки, а «\t» представляет табуляцию.

Тип wchar\_t – это основной тип, предназначенный для наборов символов, размер которых выходит за пределы одного байта.

В общем случае рекомендуется использовать типы char8\_t, char16\_t, char32\_t. Значения этих типов предназначены для хранения символов   
в кодировке UTF-8, UTF-16 и UTF-32 соответственно, а их размеры одинаковы на всех распространенных платформах.

Стандартная библиотека С++ представляет класс string,   
которые представляет собой удобный способ работы со строками. String автоматически управляет памятью и длиной строки. Также строковые переменные можно организовывать при помощи массива символов. [2]

## 2.3 Структуры данных

В С++ существует множество структур данных. К ним относятся:

– массивы;

– стандартные контейнеры библиотеки STL;

– структуры.

Массивы – это упорядоченные коллекции элементов одного типа данных. Они обладают фиксированным размером, который задается при объявлении массива. Массивы в C++ индексируются начиная с нуля.

Стандартная библиотека шаблонов в С++ предоставляет разнообразные контейнеры, которые обеспечивают хранения или управление данными.   
К таким контейнерам относятся:

1 Контейнер vector: векторы представляют собой динамически расширяемые массивы в стандартной библиотеке C++. Они автоматически увеличивают свой размер при добавлении новых элементов.

2 Контейнер list: списки представляют собой линейные структуры данных, состоящие из узлов. Каждый узел содержит данные и указатель на следующий узел.

3 Контейнер queue: реализация очереди, то есть организация FIFO структуры данных.

4 Контейнер deque: представляет собой двустороннюю очередь, которая обеспечивает быстрое добавление и удаление элементов с обоих концов.

5 Контейнер stack: реализует LIFO структуру данных.

6 Контейнер map: отображение пар ключ-значение, где ключи уникальны, позволяет быстро находить значения по ключу и эффективно вставлять и удалять пары. Имеет аналоги в виде unordered\_map, который представляет собой отображение пар ключ-значение, где ключи не упорядочены, и multimap, где может быть несколько элементов с одним и тем же ключом.

7 Контейнер set: упорядоченное множество уникальных элементов. Элементы автоматически упорядочиваются по возрастанию и не допускают дубликатов. Имеет аналоги в виде unordered\_set, который предоставляет собой неупорядоченное множество уникальных элементов, и multiset, представляющий собой упорядоченное множество элементов, которое может содержать дубликаты.

8 Контейнер bitset: массив фиксированного размера битов, где каждый бит может быть установлен или сброшен.

Структуры представляют собой пользовательский тип данных,   
который может содержать различные переменные разных типов данных.   
Они могут быть использованы для организации и хранения связанных данных в одном объекта.

## 2.4 Типы операторов

В языке программирования С++ операторы используются   
для выполнения различных операция над данными. Операторы могут выполнять арифметические, логические, сравнительные операции.

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

К унарным арифметическим операциям относятся операция инкремента (++) и операция декремента (--). Каждая из операций имеет две разновидности: префиксную и постфиксную. При префиксной унарной операции инкремент или декремент указываются вначале операнда. При постфиксной унарной операции инкремент или декремент указываются после операнда.

К логическим операторам в языке программирования С++ относятся логическое И (&&), логическое ИЛИ (||), логическое НЕ (!).

Оператор && 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 || 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 ! 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 после получения остатка от деления (%=);

– присваивание после сдвига разряда влево (<<=);

– присваивание после сдвига разряда вправо (>>=);

– присваивание после поразрядной конъюнкции (&=);

– присваивание после поразрядной дизъюнкции (|=);

– присваивание после операции исключающего ИЛИ (^=).

Операторы цикла в языке программирования С++ позволяют выполнять повторяющиеся действия в течении определенного количества итерация   
К операторам цикла языка программирования С++ относятся: цикл for, цикл while, цикл do-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.

Цикл do-while похож на цикл while, но условие проверяется   
после каждой итерации, так что тело цикла выполняется хотя бы один раз.

Условные операторы в языке программирования С++ используются   
для выполнения различных действий в зависимости от истинности   
или ложного заданного условия. К основным условным операторам относятся: оператор if, оператор if-else, оператор else if, 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 else if 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 2.2.



Рисунок 2.2 – Синтаксис тернарного условного оператора

Вышеперечисленные условные операторы позволяют в зависимости   
от выполнения условий выбирать различные пути выполнения программы   
и принимать различные решения. [3]

## 2.5 Функции

Функции в C++ являются основными строительными блоками программирования. Они представляют собой фрагменты кода, предназначенные для выполнения определенных задач или операций. Функции позволяют разделять код на логические блоки, обеспечивая модульность и повторное использование.

Функции в C++ могут иметь различные типы:

1 Функции без параметров и возвращаемого значения.

2 Функции с параметрами и без возвращаемого значения.

3 Функции с параметрами и возвращаемым значением.

Для определения функции указывается её возвращаемый тип, имя функции, список параметров и тело функции. Функции вызываются из других частей программы по имени с передачей аргументов, если они ожидаются.

В C++ также существует возможность перегрузки функций, когда несколько функций имеют одно и то же имя, но различные списки параметров. Это позволяет использовать одно имя для нескольких функций, что упрощает их использование.

Прототипы функций позволяют объявлять функции перед их фактическим определением, что обеспечивает компилятору информацию о существовании функции до её вызова. [4]

## 2.6 Классы

Классы в C++ представляют собой средство организации кода, которое позволяет объединять данные и методы для работы с ними в единый тип. Классы описывают состояние и поведение объектов.

Определение класса включает в себя переменные-члены (поля) и функции-члены (методы), которые могут быть публичными, приватными или защищёнными. Публичные методы обеспечивают интерфейс для работы с данными класса извне, тогда как приватные и защищённые члены доступны только внутри класса или его наследников.

Создание объекта класса происходит путём использования его имени, за которым следуют параметры конструктора, если они определены. Объекты могут вызывать методы класса для выполнения операций с данными, которые они содержат.

Использование классов способствует более структурированному и модульному коду, облегчает его понимание, поддержку и расширение программы. [5]

## 2.7 Подключение библиотек

В языке программирования С++ для подключения библиотек используется директива #include. Эта директива позволяет включать содержимое заголовочных файлов в программу перед компиляцией.   
После подключения библиотеки заголовочные файлы становятся доступными для использования в программе, что позволяет обращаться к их функциям, классам и другим элементам. [6]

# 3 ОПРЕДЕЛЕНИЕ ИНСТРУМЕНТАЛЬНОЙ ЯЗЫКОВОЙ             СРЕДЫ

Для разработки транслятора в качестве языковой среды был выбран язык программирования Python 3.12.1. В качестве интегрированной среды разработки был выбран PyCharm от JetBrains. Операционной системой выступает Windows, работа проводится на PC.

Python – это интерпретируемый, высокоуровневый язык программирования, который создавался с акцентом на читаемость кода и простоту использования. Он поддерживает различные парадигмы программирования, такие как объектно-ориентированное, процедурное, функциональное и императивное программирование.

Python обладает обширной стандартной библиотекой, которая предоставляет множество инструментов и модулей для различных задач, начиная от работы с файлами и сетевым взаимодействием, и заканчивая созданием веб-приложений и научными вычислениями.

Одним из основных преимуществ Python является автоматическое управление памятью с помощью механизмов подсчета ссылок и циклического сборщика мусора. При написании программ на языке программирования Python нет необходимости явно выделять и освобождать память, что упрощает процесс программирования и снижает вероятность ошибок.

Python поддерживает динамическую типизацию, что дает возможность объявлять переменные без указания их типа. Это упрощает и ускоряет процесс разработки и позволяет быстро изменять типы переменных при необходимости.

Кроме того, Python – это кроссплатформенный язык программирования, что означает, что программы, написанные на Python, могут быть запущены   
на различных операционных системах без изменений их исходного кода.

## ВЫВОДЫ

В процессе выполнения данной лабораторной работы было проведено исследование языка программирования C++, включающее детальную классификацию его ключевых аспектов. Были рассмотрены переменные и константы, типы данных, структуры данных, операторы и подключение библиотек, представленные в этом языке. С учетом поставленной цели, было определено полное подмножество выбранного языка программирования, а также представлены тексты двух или трех программ, включающих все элементы данного подмножества. Кроме того, было приведено подробное описание инструментальной языковой среды, которая может использоваться для разработки программ на C++. Анализ показал, что C++ обладает обширными возможностями для работы с различными типами и структурами данных, а также операторами.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Введение в С++ [Электронный ресурс]. – Режим доступа: <https://metanit.com/cpp/tutorial/2.5.php>. – Дата доступа: 05.02.2024.
2. Типы данных [Электронный ресурс]. – Режим доступа: <https://metanit.com/cpp/tutorial/2.3.php>. – Дата доступа: 05.02.2024.
3. Операторы в С++ [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/ru-ru/cpp/c-operators>. – Дата доступа: 07.02.2024
4. Функции С++ [Электронный ресурс]. – Режим доступа: https://metanit.com/cpp/tutorial/3.1.php. – Дата доступа: 07.02.2024.
5. Классы С++ [Электронный ресурс]. – Режим доступа: <https://ravesli.com/urok-113-klassy-obekty-i-metody-klassov/>. – Дата доступа: 07.02.2024.
6. The C++ Standard Template Library (STL) [Электронный ресурс]. – Режим доступа: https://www.geeksforgeeks.org/the-c-standard-template-library-stl/. – Дата доступа: 08.02.2024.

## ПРИЛОЖЕНИЕ А

## (обязательное)

## Пример реализации программ на языке программирования С++

Листинг 1 – Программная реализация быстрой сортировки

#include <stdio.h>

#include <stdlib.h>

#define SIZE 20

void quickSort(int \*numbers, int left, int right)

{

int pivot; // разрешающий элемент

int l\_hold = left; //левая граница

int r\_hold = right; // правая граница

pivot = numbers[left];

while (left < right) // пока границы не сомкнутся

{

while ((numbers[right] >= pivot) && (left < right))

right--; // сдвигаем правую границу пока элемент [right] больше [pivot]

if (left != right) // если границы не сомкнулись

{

numbers[left] = numbers[right]; // перемещаем элемент [right] на место разрешающего

left++; // сдвигаем левую границу вправо

}

while ((numbers[left] <= pivot) && (left < right))

left++; // сдвигаем левую границу пока элемент [left] меньше [pivot]

if (left != right) // если границы не сомкнулись

{

numbers[right] = numbers[left]; // перемещаем элемент [left] на место [right]

right--; // сдвигаем правую границу влево

}

}

numbers[left] = pivot; // ставим разрешающий элемент на место

pivot = left;

left = l\_hold;

right = r\_hold;

if (left < pivot) // Рекурсивно вызываем сортировку для левой и правой части массива

quickSort(numbers, left, pivot - 1);

if (right > pivot)

quickSort(numbers, pivot + 1, right);

}

int main()

{

int a[SIZE];

for (int i = 0; i<SIZE; i++)

a[i] = rand() % 201 - 100;

for (int i = 0; i<SIZE; i++)

printf("%4d ", a[i]);

printf("\n");

quickSort(a, 0, SIZE-1); // вызов функции сортировки

for (int i = 0; i<SIZE; i++)

printf("%4d ", a[i]);

printf("\n");

getchar();

return 0;

}

Листинг 2 – Программная реализация поиска подстроки в строке

#include <iostream>

#include <string>

int main() {

std::string text = "Hello, world!";

std::string pattern = "world";

int n = text.length();

int m = pattern.length();

int pos = -1;

for (int i = 0; i <= n - m; ++i) {

int j;

for (j = 0; j < m; ++j) {

if (text[i + j] != pattern[j]) {

break;

}

}

if (j == m) {

pos = i;

break;

}

}

if (pos != -1) {

std::cout << "Pattern found at position: " << pos << std::endl;

} else {

std::cout << "Pattern not found." << std::endl;

}

return 0;

}

Листинг 3 – Программная реализация класса Worker

class Worker {

public:

void discover\_avarage\_AP ();

void set\_avarage\_AP (double score);

double get\_avarage\_AP ();

void set\_name(string a);

string get\_name ();

void set\_academic\_performance (vector v);

private:

int avarage\_AP;

string name; // имя

int academic\_performance[6];

};

Листинг 3 – Пример программы на С++ со структурой, классами

#include <iostream>

class Cloud {

private:

int density;

int volume;

public:

Cloud(int d, int v){}

void setDensity(int d) { density=d; }

void setVolume(int v) { volume=v; }

int getDensity() const { return density; }

int getSquare() const { return square; }

int weight() const { return density \* volume; } // Weight of cloud

};

struct Bird { // Hello

char name[30];

int speed;

};

/\*

Hello again!!

\*/

int main() {

float a = 3.79;

int\* ptr = nullptr;

auto z = 0B001101 ;

int x = 2;

int y = 3;

x = x + y;

if (x > 5) {

printf("x > 5\n");

} else {

printf("x < 5\n");

}

Cloud storm(12, 40);

std::cout << "Density: " << storm.getDensity() << std::endl;

std::cout << "Volume: " << storm.getHeight() << std::endl;

std::cout << "Weight of the cloud: " << storm.weight() << std::endl;

rect.setDensity(7);

rect.setVolume(4);

std::cout << "Updated weight of the cloud: " << storm.weight() << std::endl;

int numbers[4] = {1E+10, 3.8.0.2.3, 3, 4.34};

double 22twentyTwo = 22.22;

char letters[2] = {'a', '\n'};

struct Bird nightingale = {"Earl", 30};

printf("Bird's name: %s\n", nightingale.name);

printf("Bird's speed: %d\n", nightingale.speed);

for (int i =0; i< 4; i++) {

std::cout << numbers[i] << "\n";

}

int j #= 0;

while (j < 2) {

printf("%c ", letters[j]);

j++;

}

double switch = 10;

char word = 'C';

switch (word) {

case 'A':

printf("It's A!\n");

break;

case 'B':

printf("It's B!\n");

break;

case 'C':

printf("It's C!\n");

break;

default:

printf("It's nothing(\n");

}

return 0;

}