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             А. К. Хрищанович

Проверил                          Н. Ю. Гриценко

Минск 2024

СОДЕРЖАНИЕ

[1 Постановка задачи 4](#_Toc157960226)

[2 Определение модели языка программирования С++ 5](#_Toc157960227)

[2.1 Переменные и константы 5](#_Toc157960228)

[2.2 Типы данных 6](#_Toc157960229)

[2.3 Структуры данных 8](#_Toc157960230)

[2.4 Типы операторов 9](#_Toc157960231)

[2.5 Функции 12](#_Toc157960232)

[2.6 Классы 13](#_Toc157960233)

[2.7 Подключение библиотек 13](#_Toc157960234)

[3 Определение инструментальной языковой среды 14](#_Toc157960235)

[Выводы 15](#_Toc157960236)

[Список использованных источников 16](#_Toc157960237)

[Приложение А (обязательное) Пример реализации программ на языке   
                            программирования С++ 17](#_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 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 С++

## 2.1 Переменные и константы в языке программирования С++

Для хранения данных в программе на языке С++ используются переменные, которые представляют собой именованные участки памяти. Переменная имеет тип, который определяет, какие именно данные может хранить переменная, имя и значение.[1]

Имя переменной – это последовательность алфавитных-цифровых символов и знака подчеркивания. При этом имя переменной не может начинаться с цифр, специальных знаков или подчеркивания. Кроме того,   
в качестве имени переменной нельзя использовать ключевые слова языка С++.

В общем случае определение переменной в языке программирования С++ представлено 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 ей некоторое значение. Присвоение переменной называется инициализацией. В С++ есть три вида инициализации:

– нотация присваивания;

– функциональная нотация;

– инициализация в фигурных скобках.

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



Рисунок 2.2 – Пример нотации присваивания на языке программирования С++

Инициализация в фигурных скобках организовывается при помощи указания значения переменной в фигурных скобках после ее названия. Пример инициализации в фигурных скобках на языке программирования С++ представлен на рисунке 2.3.



Рисунок 2.3 – Пример инициализации в фигурных скобках на языке программирования С++

При функциональной нотации после названия переменной в круглых скобках указывается ее значение. Пример функциональной нотации на языке программирования С++ представлен на рисунке 2.4.



Рисунок 2.4 – Пример инициализации в фигурных скобках на языке программирования С++

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

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.

## 2.2 Типы данных в языке программирования С++

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.[2]

**2.2.1** Логический тип

Логический тип bool может хранить одно из двух значений:

– true;

– false.

Логический тип в основном применяется в условных выражениях. Значением по умолчанию для переменных этого типы является значение false.

**2.2.2** Целочисленные типы

Целые числа в языке программирования С++ представлены следующими типами:

– знаковыми;

– беззнаковыми.

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. Различие signed char от unsigned char заключается в том,   
что signed char хранит любое значение из диапазона от -128 до 127, а unsigned char хранит любой значение из диапазона от 0 до 255. Тип данных char представляет один символ в кодировке ASCII и занимает 1 байт. Тип 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 предоставления чисел в языке программирования С++ применяются целочисленные литералы со знаком или без него.

**2.2.3** Числа с плавающей точкой

Для хранения дробных чисел в С++ применяются числа с плавающей точкой. Число с плавающей точкой состоит из двух частей: мантиссы   
и показателя степени. Они 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.

**2.2.4** Символьные типы

К символьным типа данных в языке программирования С++ относятся char, wchar\_t, char8\_t, char16\_t, char32\_t.

Переменные типы char хранят числовой код одного символа и занимает один байт. Обычно для символов char используется кодировка ASCII.   
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. Значения w\_char определяются также, как и символы char. Единственное исключение,   
что они предваряются символом «L».

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.

**2.2.5** Строковый тип данных

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.

**2.2.6** Спецификатор auto

В случае затруднительного определения типа выражения следует применять спецификатор auto. При определении переменной   
со спецификатором auto, эта переменная должна быть обязательно инициализировано каким-либо значением.

## 2.3 Структуры данных в языке программирования С++

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

– массивы;

– стандартные контейнеры библиотеки STL;

– структуры.

Массив представляет собой последовательность элементов одного типа, расположенных в памяти друг за другом. Размер массива определяется   
при его объявлении и не может быть изменен во время выполнения программы.

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

1 Контейнер vector: данный тип контейнера представляет собой динамический массив, который автоматически расширяется при добавлении новых элементов. Организовывает быстрый доступ к элементам по индексу   
и эффективное добавление и удаление элементов в конец, а также реализуется все функции массива, но имеет дополнительные методы.

2 Контейнер list: данный тип контейнера представляет двусвязный список, где каждый элемент содержит указатели на предыдущий и следующий элементы. Обеспечивает быстрое добавление и удаление элементов   
в любом месте списка.

3 Контейнер deque: данный тип контейнера представляет   
собой двустороннюю очередь, которая обеспечивает быстрое добавление   
и удаление элементов с обоих концов. Контейнер deque очень похож на vector, но имеет более эффективные операции добавления и удаления в начале   
и середине контейнера.

4 Контейнер queue: данный тип контейнера представляет   
собой реализацию очереди, то есть организовывает FIFO структуру данных.

5 Контейнер stack: данный тип контейнера реализует LIFO структуру данных.

6 Контейнер set: данный тип контейнера представляет   
собой 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,   
который представляет собой упорядоченное множество элементов,   
которое может содержать дубликаты.

7 Контейнер map: данный тип контейнера представляет   
собой 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.

8 Контейнер bitset: данный тип контейнера представляет 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++

В языке программирования С++ операторы используются   
для выполнения различных операция над данными. Операторы могут выполнять арифметические, логические, сравнительные операции.[3]

**2.4.1** Арифметические операторы

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 +, -, \*, /, %.

Также есть две унарные арифметические операции, к которым относятся операция инкремента (++) и операция декремента (--). Каждая из операций имеет две разновидности: префиксная и постфиксная. При префиксной унарной операции инкремент или декремент указываются вначале операнда. При постфиксной унарной операции инкремент или декремент указываются после операнда.

**2.4.2** Логические операторы

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.

**2.4.3** Операторы сравнения

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.

**2.4.4** Побитовые операторы

К побитовым операторам относятся:

– побитовый И (&);

– побитовый ИЛИ (|);

– побитовое исключающее ИЛИ (^);

– побитовое отрицание (~);

– побитовый сдвиг влево (<<);

– побитовый сдвиг вправо (>>).

Данные операторы используются для работы с битами в целочисленных типах данных.

**2.4.5** Операторы присваивания

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 (^=).

Все вышеперечисленные операции присваивания являются сочетанием простой операции присваивания с другими операциями.

**2.4.6** Операторы доступа к элементам

К операторам доступа к элементам в языке программирования С++ относятся:

– операторы квадратных скобок ([]);

– оператор точки (.);

– оператор стрелки (->);

– оператор двойной точки (::).

Оператор квадратных скобок используется для доступа к элементам массива по индексу.

Оператор точки используется для доступа к членам структуры   
или класса по имени.

Оператор стрелки используется для доступа к членам класса   
через указатель на объект класса.

Оператор двойной точки используется для доступа к статическим членам класса или глобальным переменным или функциям извне класса.

**2.4.7** Операторы цикла

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.

**2.4.8** Условные операторы

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 2.5.



Рисунок 2.5 – 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 в языке программирования С++

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. Вначале указывается возвращаемый тип функции. Если функция не возвращает никакого значения, то используется тип void.

Для возвращения результата функция применяет оператор return. Если функция имеет в качестве возвращаемого типа любой тип, кроме void, то она должна обязательно с помощью оператора return возвращать какое-либо значение.

## 2.6 Классы в языке программирования С++

Одной из наиболее полезных особенностей языка C++ является возможность определять собственные типы данных, которые будут лучше соответствовать в решении конкретных проблем.[5]

В объектно-ориентированном программировании типы данных могут содержать не только данные, но и функции, которые будут работать с этими данными. Для определения такого типа данных в языке C++ используется **ключевое слово class**. Использование ключевого слова class определяет новый **пользовательский тип данных – класс.**

В языке C++ переменная класса называется **экземпляромкласса**. Помимо хранения данных, классы могут содержать и функции. Функции, определенные внутри класса, называются **методами.**

## 2.7 Подключение библиотек в языке программирования С++

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.

# 3 ОПРЕДЕЛЕНИЕ ИНСТРУМЕНТАЛЬНОЙ ЯЗЫКОВОЙ             СРЕДЫ

Для разработки транслятора был выбран язык программирования Python 3.12.1. Python – это современный и мощный язык программирования общего назначения, который отличается высокой производительностью разработки   
и читаемостью кода.

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 также имеет динамическую строгую типизацию. Данный язык программирования обладает динамическим разрешением имен, что означает, что имена методов и переменных связываются с их объектами во время выполнения программы.

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.

В качестве интегрированной среды разработки был выбран PyCharm   
от JetBrains.

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 С++. В результате проведенного анализа были описаны переменные и константы языка программирования С++, типы данных, структуры данных, типы операторов и подключение библиотек. В результате определения модели языка было определено, что язык программирования С++ предоставляет широкие возможности 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: 28.01.2024.
2. Типы данных [Электронный ресурс]. – Режим доступа: <https://metanit.com/cpp/tutorial/2.3.php>. – Дата доступа: 28.01.2024.
3. Операторы в С++ [Электронный ресурс]. – Режим доступа: <https://learn.microsoft.com/ru-ru/cpp/c-operators>. – Дата доступа: 04.02.2024
4. Функции С++ [Электронный ресурс]. – Режим доступа: https://metanit.com/cpp/tutorial/3.1.php. – Дата доступа: 04.02.2024.
5. Классы С++ [Электронный ресурс]. – Режим доступа: <https://ravesli.com/urok-113-klassy-obekty-i-metody-klassov/>. – Дата доступа: 04.02.2024.
6. The C++ Standard Template Library (STL) [Электронный ресурс]. – Режим доступа: https://www.geeksforgeeks.org/the-c-standard-template-library-stl/. – Дата доступа: 30.01.2024.

## ПРИЛОЖЕНИЕ А

## (обязательное)

## Пример реализации программ на языке программирования С++

Листинг 1 – Программная реализация сортировки пузырьком

#include <iostream>

#include <vector>

int main() {

std::vector<int> arr = {64, 25, 12, 22, 11};

int n = arr.size();

for (int i = 0; i < n - 1; ++i) {

for (int j = 0; j < n - i - 1; ++j) {

if (arr[j] > arr[j + 1]) {

std::swap(arr[j], arr[j + 1]);

}

}

}

std::cout << "Sorted array: ";

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

std::cout << arr[i] << " ";

}

std::cout << std::endl;

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 Rectangle

#include <iostream>

class Rectangle {

private:

int width;

int height;

public:

Rectangle(int w, int h) : width(w), height(h) {}

void setWidth(int w) { width = w; }

void setHeight(int h) { height = h; }

int getWidth() const { return width; }

int getHeight() const { return height; }

int area() const { return width \* height; }

};

int main() {

Rectangle rect(5, 3);

std::cout << "Width: " << rect.getWidth() << std::endl;

std::cout << "Height: " << rect.getHeight() << std::endl;

std::cout << "Area of the rectangle: " << rect.area() << std::endl;

rect.setWidth(7);

rect.setHeight(4);

std::cout << "Updated area of the rectangle: " << rect.area() << std::endl;

return 0;

}