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. А. Сивый

Проверил Н. Ю. Гриценко

Минск 2024**СОДЕРЖАНИЕ**

1 Цели работы3

2 Подмножество языка программирования4

2.1 Числовые и строковые константы4

2.2 Типы переменных4

2.3 Операторы цикла6

2.4 Условные операторы6

2.5 Структуры данных7

2.6 Типы операторов8

2.7 Подпрограммы9

2.8 Библиотеки9

3 Инструментальная языковая среда10

Вывод11

Список использованных источников12

Приложение А (обязательное) Листинг программного кода13

**1 ЦЕЛИ РАБОТЫ**

1 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)

2 Определить инструментальную языковую среду т.е. язык программирования и операционную систему для разработки.

**2 ПОДМНОЖЕСТВО ЯЗЫКА ПРОГРАММИРОВАНИЯ**

В качестве подмножества языка программирования был выбран язык C++.

С++ – высокоуровневый компилируемый язык программирования общего назначения со статической типизацией, который подходит для создания самых различных приложений.

Данный язык программирования поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы [1].

**2.1 Числовые и строковые константы**

Константы относятся к фиксированным значениям, которые программа не может изменить, они называются литералами. Константы могут относиться к любому из основных типов данных и могут быть разделены на целые числа, числа с плавающей запятой, символы, строки и логические значения.

Целочисленные литералы могут быть представлены в десятичной, восьмеричной или шестнадцатеричной системе. Префикс определяет базу или систему, например, 0x – для шестнадцатеричного числа, 0b – для двоичного и 0 для восьмеричного.

Литералы с плавающей запятой имеют целую, дробную и экспоненциальную часть. Данный тип литералов может быть представлен как в десятичной, так и в экспоненциальной форме.

Строковые литералы заключаются в двойные кавычки. Строка содержит простые символы, escape-последовательности и универсальные символы. Длинную строку можно разбить на несколько строк, используя строковые литералы и разделить их пробелами. [2]

**2.2 Типы переменных**

В языке С++ каждая переменная имеет определенный тип. Этот тип определяет, какие значения может иметь переменная, какие операции с ней можно производить и сколько байт памяти она будет занимать. Существуют следующие базовые типы данных: логический тип bool, целочисленные типы, типы чисел с плавающей точкой, символьные типы.

Целочисленные типы данных в С++ предназначены для хранения целых чисел без дробной части. Они используются для представления значений, которые не должны содержать десятичные дроби. В С++ целочисленные типы представлены следующими ключевыми словами:

1 Тип int. Этот тип представляет обычные целые числа. Обычно занимает 4 байта в большинстве современных систем.

2 Тип short. Используется для коротких целых чисел и обычно занимает 2 байта.

3 Тип long. Используется для представления длинных целых чисел и обычно занимает 4 байта на 32-битных системах и 8 байт на 64-битных системах.

4 Тип long long. Этот тип представляет большую длину для целых чисел и обычно занимает 8 байт даже на 32-битных системах.

Также целочисленные типы могут быть signed (со знаком) или unsigned (без знака). Когда тип не указан явно, то тип считается signed. Ключевое слово unsigned используется для представления только положительных значений и удваивает диапазон значений для типа.

Типы данных с плавающей точкой в С++ предназначены для представления чисел с десятичной точкой или чисел, содержащих дробную часть. Эти типы данных используются, когда необходима более точная репрезентация вещественных чисел, чем целочисленные. В С++ типы с плавающей точкой представлены ключевыми словами:

1 Тип float. Он используется для представления чисел с плавающей запятой одинарной точности. Обычно занимает 4 байта.

2 Тип double. Предоставляет большую точность по сравнению с float и используется для представления чисел с плавающей точкой двойной точности. Обычно занимает 8 байт.

3 Тип long double. Обеспечивает еще большую точность по сравнению с double. Занимает обычно 8 байт, но может быть более точным на некоторых системах.

Логический тип может хранить одно из двух значений: true (истинно, верно) и false (неверно, ложно).

Также в С++ имеются следующие символьные типы данных:

1 Тип char. Представляет один символ в кодировке ASCII. Занимает в памяти 1 байт. Может хранить любое значение из диапазона от -128 до 127, либо от 0 до 255.

2 Тип wchar\_t. Представляет расширенный символ. На Windows занимает в памяти 2 байта, на Linux – 4 байта. Может хранить любое значение из диапазона от 0 до 64535 (2 байта), либо от 0 до 4294967295 (для 4 байт).

3 Тип char8\_t. Представляет один символ в кодировке Unicode. Занимает в памяти 1 байт. Может хранить любое значение из диапазона от 0 до 256.

4 Тип char16\_t. Представляет один символ в кодировке Unicode. Занимает в памяти 2 байта. Может хранить любое значение из диапазона от 0 до 65535.

5 Тип char32\_t. Представляет один символ в кодировке Unicode. Занимает 4 байта. Может хранить любое значение из диапазона от 0 до 4294967295.

Для работы со строками в С++ есть тип данных string, который является частью стандартной библиотеки С++ (STL) и представляет удобный способ работы со строками переменной длины [3].

**2.3 Операторы цикла**

Циклы позволяют выполнять некоторый набор инструкций множество раз, пока соблюдается определенное условие. В языке С++ имеются циклы for, while, do…while.

Цикл while выполняет некоторый код, пока его условие истинно, то есть возвращает true. После ключевого слова while в скобках идет условное выражение, которое возвращает true или false. Затем в фигурных скобках идет набор инструкций, которые составляют тело цикла. И пока условие возвращает true, будут выполняться инструкции в теле цикла.

Цикл for представляет собой конструкцию, которая состоит из ключевого слова, после которого в скобках указываются инициализатор, условие и итерация. Дальше после скобок идут фигурные скобки, которые содержат набор инструкций, представляющих тело цикла. Так же, как и в цикле while, действие цикла for происходит пока выполняется условие. В некоторых случаях инициализатор и итерация при объявлении цикла могут быть пропущены.

Существует особая форма цикла for, которая предназначена для работы с последовательностями значений. Эта форма задается следующим образом: сначала пишется ключевое слово for, после чего в скобках через двоеточие указываются тип переменной и последовательность.

В цикле do сначала выполняется код цикла, а потом происходит проверка условия в инструкции while. И пока это условие истинно, то есть не равно 0, то цикл повторяется.

Помимо операторов цикла, в С++ есть операторы break и continue. Оператор break используется для выхода из тела цикла, а continue – прерывает одну итерацию в цикле [4].

**2.4 Условные операторы**

В С++ существует несколько типов условных конструкций, которые позволяют выполнение различных блоков кода в зависимости от условий:

1 Оператор if. Этот оператор используется для выполнения блока кода, если заданное условие истинно.

2 Оператор if-else. Эта конструкция позволяет выполнить один из двух блоков кода в зависимости от условия.

3 Оператор else if. Позволяет проверять несколько условий последовательно.

4 Оператор switch. Используется для выполнения блока кода в зависимости от значения выражения.

5 Тернарный оператор. Это специальный оператор в С++, который позволяет компактно записывать условные выражения.

Условные операторы позволяют программам самостоятельно принимать решения, что делает их выполнение более гибким. [5]

**2.5 Структуры данных**

С++ предоставляет различные встроенные и пользовательские структуры данных для эффективного хранения и организации данных. С++ имеет следующие структуры данных:

1 Массив. Массив представляет собой набор элементов одного типа с фиксированным размером. Элементы массива хранятся в памяти последовательно.

2 Структура (struct). Структура позволяет различные типы данных в одной переменной.

3 Класс (class). Класс также позволяет объединять данные и функции, но в отличие от структуры, он поддерживает инкапсуляцию, наследование и полиморфизм.

4 Список (list). Список представляет собой динамическую структуру данных, в которой каждый элемент (узел) содержит данные и указатель на следующий элемент.

5 Вектор (vector). Вектор является динамическим массивом, который автоматически увеличивается при добавлении новых элементов.

6 Стек (stack). Стек представляет собой структуру данных, которая следует принципу «последний вошел, первый вышел» (Last In, First Out, LIFO). Он работает по принципу добавления элементов на вершину стека и удаления их с вершины.

7 Очередь (queue). В С++ представляет собой структуру данных, которая работает по принципу «первый вошел, первый вышел» (First In, First Out, FIFO). Это означает, что элементы добавляются в конец очереди, а удаление происходит с её начала.

8 Двусторонняя очередь (deque). Данный коллекция обеспечивает быстрое добавление и удаление элементов с обеих сторон. Этот контейнер очень похож на vector, но имеет более эффективные операции добавления и удаления в начале и в середине контейнера.

9 Множество (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.

10 Ассоциативный массив (map). Этот тип контейнера используется для хранения и извлечения данных из коллекции, в которой каждый элемент является парой, обладающей одновременно значением данных и ключом сортировки. [6].

**2.6 Типы операторов**

В языке программирования С++ существует множество операторов, которые выполняют различные операции. Среди них выделяют: арифметические операторы, операторы сравнения, логические операторы, операторы инкремента и декремента, оператор присваивания, операторы побитовых операций.

Арифметические операторы используются для выполнения математических операций над числами. К арифметическим операторам относятся:

– оператор сложения (+);

– оператор вычитания (-);

– оператор умножения (\*);

– оператор деления (/);

– оператор остатка от деления (%);

– оператор инкремента (++);

– оператор декремента (--);

Эти операторы применяются к переменным различных типов данных, таким как целые числа (int), числа с плавающей точкой (float, double), а также к пользовательским типам данных, где определены соответствующие операции. Арифметические операторы играют важную роль в многих аспектах программирования, включая вычисления, манипуляции и формирование выражений.

Операторы сравнения используются для сравнения значений переменных или выражений. Они возвращают логическое значение true или false в зависимости от выполнения условия. К операторам сравнения относятся:

– оператор равенства (==);

– оператор неравенства (!=);

– оператор меньше (<);

– оператор больше (>);

– оператор меньше или равно (<=);

– оператор больше или равно (>=);

Логические операторы в С++ используются для комбинирования и сравнения и логических условий. Они возвращают булево значение (true или false). Логические операторы включают:

– логическое И (&&);

– логическое ИЛИ (||);

– логическое НЕ (!);

Операторы побитовых операций позволяют выполнять манипуляции с битами в представлении чисел. Эти операторы работают на уровне отдельных битов чисел. Побитовые операции бывают:

– побитовое И (&);

– побитовое ИЛИ (|);

– побитовое исключающее ИЛИ (^);

– побитовое НЕ (~);

– побитовый сдвиг влево (<<);

– побитовый сдвиг вправо (>>);

Операторы присваивания в С++ используются для присвоения значений переменным. Они присваивают результат выражения правого операнда переменной слева. В С++ существуют следующие типы операторов:

– присваивание (=)

– присваивание с операцией сложения (+=)

– присваивание с операцией вычитания (-=)

– присваивание с операцией умножения (\*=)

– присваивание с операцией деления (/=)

– присваивание с операцией взятия остатка (%=)

Операторы присваивания позволяют упростить запись выражений, комбинируя операцию и присваивание в одном выражении [7].

**2.7 Подпрограммы**

Подпрограммы (функции) – это блоки кода, выполняющие определенные операции. Если требуется, функция может определять входные параметры, позволяющие вызывающим объектам передавать ей аргументы. При необходимости функция может возвращать значение как выходное. Функции полезны для инкапсуляции основных операций в едином блоке, который может многократно использоваться [8].

**2.8 Библиотеки**

В С++ библиотеки представляют собой набор функций и классов, предназначенных для решения определенных задач. Они обеспечивают возможность повторного использования кода и упрощают разработку программ. Подключение библиотек в С++ осуществляется с использованием директивы препроцессора #include. Эта директива позволяет вставлять содержимое файла заголовка непосредственно в исходный код [9].

**3 ИНСТРУМЕНТАЛЬНАЯ ЯЗЫКОВАЯ СРЕДА**

В качестве языка программирования для реализации программы был выбран язык программирования C# версии 12. C# – это современный объектно-ориентированный и типобезопасный язык программирования. C# позволяет создавать разные типы безопасных и надежных приложений, выполняющихся в .NET. C# относится к широко известному семейству языков C, и покажется знакомым любому, кто работал с C, C++, Java или JavaScript.

Программы C# выполняются в .NET, виртуальной системе выполнения, вызывающей общеязыковую среду (CLR) и набор библиотек классов. Среда CLR – это реализация общеязыковой инфраструктуры языка (CLI), являющейся международным стандартом от корпорации Майкрософт.

Исходный код, написанный на C# компилируется в промежуточный язык (IL), который соответствует спецификациям CLI. Код на языке IL сохраняется в сборке, обычно с расширением .dll. Сборка содержит манифест с информацией о типах, версии, языке и региональных параметрах для этой сборке.

При выполнении программы C# сборка загружается в среду CLR. Среда CLR выполняет JIT-компиляцию из кода на языке IL в инструкции машинного языка. Среда CLR также выполняет другие операции, например, автоматическую сборку мусора, обработку исключений и управление ресурсами. Код, выполняемый средой CLR, иногда называют «управляемым кодом». «Неуправляемый код» компилируется на машинный язык, предназначенный для конкретной платформы [10].

Для написания кода выбрана интегрированная среда разработки JetBrains Rider.

В качестве операционной системы выбрана ОС Windows 10.

**ВЫВОД**

В ходе выполнения лабораторной работы было выбрано подмножество языка программирования, которое включает в себя основные элементы для создания структурированных программ, такие как числовые и текстовые константы, различные типы переменных, операторы цикла, условные операторы, различные структуры данных, функции, а также различные операторы.

Для разработки программы на данном подмножестве языка программирования была выбрана инструментальная языковая среда на основе языка C#, так как он обладает высокой производительностью и широким спектром возможностей.

**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] C++ Constants/Literals [Электронный ресурс] – Режим доступа: <https://www.tutorialspoint.com/cplusplus/cpp_constants_literals.htm> – Дата доступа: 05.02.2024.

[3] Типы данных С++ [Электронный ресурс] – Режим доступа: <https://metanit.com/cpp/tutorial/2.3.php>. – Дата доступа: 05.02.2024.

[4] Циклы С++ [Электронный ресурс] – Режим доступа: <https://metanit.com/cpp/tutorial/2.13.php> – Дата доступа: 05.02.2024.

[5] Ветвление в С++ [Электронный ресурс] – Режим доступа: <https://code-live.ru/post/cpp-if-else/> – Дата доступа: 05.02.2024.

[6] Структуры данных в С++ [Электронный ресурс] – Режим доступа: <https://acmp.ru/asp/do/index.asp?main=topic&id_course=2&id_section=14&id_topic=12> – Дата доступа: 05.02.2024.

[7] Встроенные операторы в С++ [Электронный ресурс] – Режим доступа: <https://learn.microsoft.com/ru-ru/cpp/cpp/cpp-built-in-operators-precedence-and-associativity?view=msvc-170> – Дата доступа: 05.02.2024

[8] Функции в С++ [Электронный ресурс] – Режим доступа: <https://learn.microsoft.com/ru-ru/cpp/cpp/functions-cpp?view=msvc-170> – Дата доступа: 05.02.2024.

[9] Статические и динамические библиотеки [Электронный ресурс] – Режим доступа: <https://ravesli.com/staticheskie-i-dinamicheskie-biblioteki/> – Дата доступа: 05.02.2024.

[10] Документация C# [Электронный ресурс] – Режим доступа: <https://learn.microsoft.com/ru-ru/dotnet/csharp/> – Дата доступа: 05.02.2024.

**ПРИЛОЖЕНИЕ А**

**(обязательное)**

**Листинг программного кода**

Листинг 1 – Пример программы сортировки «пузырьком»

#include <iostream>

#include <vector>

void bubbleSort(std::vector<int>& arr) {

int n = arr.size();

bool swapped;

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

swapped = false;

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

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

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

swapped = true;

}

}

if (!swapped) {

break;

}

}

}

int main() {

std::vector<int> numbers = {5, 2, 8, 1, 7};

std::cout << "Unsorted array: ";

for (const auto& num : numbers) {

std::cout << num << " ";

}

std::cout << std::endl;

// Применяем сортировку пузырьком

bubbleSort(numbers);

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

for (const auto& num : numbers) {

std::cout << num << " ";

}

std::cout << std::endl;

return 0;

}

Листинг 2 – Пример программы двоичного поиска

#include <iostream>

#include <vector>

int binarySearch(const std::vector<int>& array, int target) {

int left = 0;

int right = array.size() - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (array[mid] == target) {

// Элемент найден, возвращаем его индекс

return mid;

} else if (array[mid] < target) {

// Искомый элемент находится в правой половине

left = mid + 1;

} else {

// Искомый элемент находится в левой половине

right = mid - 1;

}

}

// Элемент не найден, возвращаем -1

return -1;

}

int main() {

std::vector<int> sortedArray = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

int target = 6;

int result = binarySearch(sortedArray, target);

if (result != -1) {

std::cout << "Элемент " << target << " найден по индексу " << result << std::endl;

} else {

std::cout << "Элемент " << target << " не найден в массиве" << std::endl;

}

return 0;

}