**Практическое занятие 1 Тема: Программирование алгоритмов линейной структуры**

**Цель работы**: изучение основных типов данных, способов описания переменных различных типов, операторов присваивания и организации ввода – вывода.

**Краткие теоретические сведения:** Алгоритм – это последовательность действий, выполняемых по строго определенным правилам, однозначно определяющая процесс решения задачи и заведомо приводящая к её решению за некоторое количество шагов. Алгоритмизация – разработка формального метода решения практической задачи с возможностью реализации в виде программы для ЭВМ.

Изображение алгоритма в виде схемы выполняется в соответствии с ГОСТ 19.701–90 Единой системы программной документации «Схемы алгоритмов, программ, данных и систем».

**СОЗДАНИЕ И ЗАПУСК ПРОЕКТА КОНСОЛЬНОГО ПРИЛОЖЕНИЯ C++**

## Необходимые компоненты

* Установите и запустите на своем компьютере Visual Studio с рабочей нагрузкой Разработка классических приложений на C++. Если он не установлен, выполните действия, описанные в  ["Установка C++ поддержки в Visual Studio](https://learn.microsoft.com/ru-ru/cpp/build/vscpp-step-0-installation?view=msvc-170)".
* Создайте проект Hello, World!" По умолчанию он содержит код для печати Hello World!. Если вы еще не сделали этого, выполните действия, описанные в разделе [Создание проекта консольного приложения С++](https://learn.microsoft.com/ru-ru/cpp/build/vscpp-step-1-create?view=msvc-170).

Если Visual Studio выглядит следующим образом, можно приступать к сборке и запуску приложения:



## Сборка и запуск кода в Visual Studio

1. Для сборки проекта выберите в меню **Сборка** пункт **Собрать решение**. Окно **Вывод** отображает результаты процесса сборки.



Чтобы запустить этот код, в строке меню выберите **Отладка** и **Запуск без отладки**.



1. Открывается окно консоли, и запускается ваше приложение. При запуске консольного приложения в Visual Studio система выполняет код, а затем выводит сообщение "Нажмите любую клавишу, чтобы продолжить. . .", чтобы вы могли просмотреть выходные данные.

Поздравляем! Вы создали свое первое консольное приложение "Hello, world!" в Visual Studio! Нажмите любую клавишу, чтобы закрыть окно консоли и вернуться в редактор Visual Studio.

**Структура программ для С++ Microsoft Visual Studio.**

|  |  |
| --- | --- |
| 1 | // struct\_program.cpp: определяет точку входа для консольного приложения. |
| 2 | #include "stdafx.h" //здесь подключаем все необходимые препроцессорные директивы |
| 3 | int main()  |
| 4 | { // начало главной функции с именем main //здесь будет находится ваш программный код  |
| 5 | } |

В строке 1 говорится о точке входа для консольного приложения, это значит, что данную программу можно запустить через командную строку Windows указав имя программы, к примеру, такое struct\_program.cpp. Строка 1 является однострочным комментарием, так как начинается с символов //. В строке 2 подключен заголовочный файл "stdafx.h". Данный файл похож на контейнер, так как в нем подключены основные препроцессорные директивы (те, что подключил компилятор, при создании консольного приложения), и вспомогательные (подключенные программистом). include — директива препроцессора, т. е. сообщение препроцессору.

Строки, начинающиеся с символа # обрабатываются препроцессором до компиляции программы. заголовочные файлы: Библиотека cmath определяет набор функций для выполнения общих математических операций и преобразований.

**Математические функции:**

1. тригонометрические функции:

cos-вычисление косинуса угла, переведенного в радианы;

sin-вычисление синуса угла, переведенного в радианы;

tan-вычисление тангенса угла, переведенного в радианы;

acos-вычисление арккосинуса, результат будет в радианах;

asin-вычисление арксинуса, результат будет в радианах;

atan-вычисление арктангенса, возвращаемый результат будет в радианах;

atan2-вычисление арктангенса и квадранта по координатам x и y, возвращаемый результат будет в радианах;

1. экспоненциальные и логарифмические функции:

exp-вычисление экспоненты;

frexp-получить мантиссу и показатель степени двойки;

ldexp-генерация числа по значению мантиссы и показателю степени;

log-натуральный логарифм;

log10-десятичный логарифм;

modf-разделение вещественного значения на дробную и целую части;

1. функции степени:

pow-возведение числа в степень.

sqrt-корень квадратный;

1. округление, модуль и другие функции:

ceil-округление до наименьшего целого значения;

fabs-вычислить модуль значения;

floor-округление до наибольшего целого значения;

fmod-остаток от деления числителя на знаменатель.

Файлы кода C++ (с расширением .cpp) не являются единственными файлами в проектах и программах. Есть еще один тип файлов, который называется **заголовочный файл** (файл заголовка*,* подключаемый файл или header file**)**. Они имеют расширение .h. Целью заголовочных файлов является удобное хранение предварительных объявлений для использования другими файлами. Всё содержимое из заголовочного файла копируется в файл \*.срр, т.е. всё содержимое становится доступным для использования.

Заголовочные файлы:

— cstdio (stdio.h)-заголовочный файл для выполнения операций ввода/вывода; — cstring (string.h)-заголовочный файл для работы со строками;

— iostream-заголовочный файл с классами, функциями и переменными для организации ввода-вывода.

Для удобства в библиотеке определены три стандартных объекта-потока: **cin** – объект класса istream, соответствующий стандартному вводу. В общем случае он позволяет читать данные с терминала пользователя;

**cout** – объект класса ostream, соответствующий стандартному выводу. В общем случае он позволяет выводить данные на терминал пользователя;

**cerr** – объект класса ostream, соответствующий стандартному выводу для ошибок. В этот поток мы направляем сообщения об ошибках программы.

Вывод осуществляется, как правило, с помощью оператора сдвига влево (**<<**), а ввод – с помощью оператора сдвига вправо (**>>**).

**Таблица 1** – Типы данных С++



Способы ввода данных в языке возможно двумя способами: ***форматированные ввод-вывод*** или ***потоковый***.

При форматированном способе используются операторы ввода **scanf** вывода **printf**.

Синтаксис операторов имеет вид:

scanf(<строка описания форматов> [, <список ввода>]);

printf(<строка описания форматов> [, <список вывода>]);

Строка описания форматов состоит из обычных символов, специальных управляющих последовательностей символов и спецификаций формата. Обычные символы и управляющие последовательности просто копируются в стандартный выходной поток в порядке их появления. Спецификации формата начинаются с символа % и заканчиваются символом, определяющим тип выводимого значения. Кроме того, спецификации формата могут содержать символы и цифры для управления видом выводимого значения (подробно см. ниже). Список вывода состоит из переменных и/или констант, значения которых должны быть выведены. Количество спецификаций формата должно быть равно количеству выводимых значений, которые указываются в списке вывода.

К управляющим последовательностям относятся последовательности символов, представленных в таблице 2.

**Таблица 2** – Управляющие символы

|  |  |
| --- | --- |
| **Последовательность** | **Действие** |
| \а | Звуковой сигнал |
| \b | Удаление предыдущего символа |
| \n | Новая строка |
| \r | Возврат каретки |
| \t | Горизонтальная табуляция |
| \v | Вертикальная табуляция |
| V | Апостроф |
| \” | Кавычки |
| \\ | Обратный слеш |
| \ooo | ASCII символ в восьмеричной нотации |
| \xooo | ASCII символ в шестнадцатеричной нотации |

Иногда при работе операторов используются спецификаторы форматов (таблица 3-5).

**Таблица 3** – Спецификаторы формата для оператора printf

|  |  |  |  |
| --- | --- | --- | --- |
| **Символ** | **Назначение** | **Символ** | **Назначение** |
| %с | символ | %o | восьмеричное число |
| %d | целое десятичное число | %s | строка символов |
| %i | целое десятичное число | %u | беззнаковое десятичное число |
| %e | десятичное число в виде x.xx e+xx | %x | шестнадцатеричное число |
| %E | десятичное число в виде x.xx E+xx | %X | шестнадцатеричное число |
| %f | десятичное число с плавающей запятойxx.xxxx | %% | символ % |
| %F | десятичное число с плавающей запятой xx.xxxx | %p | указатель |
| %g | %f или %e, что короче | %n | указатель |
| %G | %F или %E, что короче |  |  |

Кроме того, к командам формата могут быть применены модификаторы l и h (таблица 4).

**Таблица 4**

|  |  |
| --- | --- |
| **Обозначение** | **Назначение** |
| %ld | печать long int |
| %hu | печать short unsigned |
| %Lf | печать long double |

**Таблица 5** – Спецификаторы формата для оператора scanf

|  |  |
| --- | --- |
| **Символы** | **Назначение** |
| %c | чтение символа |
| %d | чтение десятичного целого |
| %i | чтение десятичного целого |
| %e | чтение числа типа float (плавающая запятая) |
| %h | чтение short int |
| %o | чтение восьмеричного числа |
| %s | чтение строки |
| %x | чтение шестнадцатеричного числа |
| %p | чтение указателя |
| %n | чтение указателя в увеличенном формате |

Примеры использования спецификаторов и модификаторов в операторах ввода-вывода:



Составим программу на языке С++ для умножения двух чисел.



|  |  |
| --- | --- |
| **Пример выполнения задания**.Задание: вычислить значение функции  при: A = 0.5 ; B = 0.6 ; C = 6.Схема программы для задания представлена на рисунке: |  |

 

Индивидуальное задание по вариантам находится в

**Вопросы самоконтроля**

1. Структура программы на языке С++

2. Этапы обработки текста программы. Включение текстов из заголовочных

файлов.

3. Директивы препроцессора, заголовочные файлы, прототипы библиотечных

функций, их вызовы.

4. Главная функция программы. Структура функции, ее заголовок.

5. Определение переменных в программе. Типы переменных.

6. Ввод-вывод данных потоком в стиле С++.

7. Операции, выражения. Оператор присваивания.