|  |
| --- |
| **МИНОБРНАУКИ РОССИИ**  **САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  **ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  **«ЛЭТИ» ИМ. В. И. УЛЬЯНОВА (ЛЕНИНА)**  **Кафедра РЭС** |
| **ОТЧЕТ по лабораторной работе №6  по дисциплине «Информатика» Тема: Использование классов для создания телефонной книги.** |
| |  |  | | --- | --- | | Студент гр. 1183 | Чаминов Д. А. | | Преподаватель | Ситников И. Ю. | |

# Содержание

[Содержание 2](#_Toc86833440)

[Спецификация задания 3](#_Toc86833441)

[Описание алгоритма 4](#_Toc86833442)

[Блок-схема алгоритма 5](#_Toc86833443)

[Выбор и обоснование типов переменных 8](#_Toc86833444)

[Вводимые и выводимые параметры и их типы 9](#_Toc86833445)

[Структура проекта 10](#_Toc86833446)

[Текст программы 11](#_Toc86833447)

[Копии экрана при работе программы 12](#_Toc86833448)

[Контрольный пример, сравнение с эталоном 13](#_Toc86833449)

[Результаты 14](#_Toc86833450)

# Спецификация задания

Разрабатываемая программа должна

* разрабатываться на языке C++,
* разрабатываться в среде MS Visual Studio,
* работать в операционных системах MS Windows 7 и выше,
* использовать хотя бы 2 пользовательских класса,
* принимать данные контактов телефонной книги (имя и номер), сортировать их и выводить в виде таблицы.

# Описание алгоритма

Для хранения телефонной книги создается экземпляр класса Phonebook, который содержит всего одно поле типа std::vector<PhoneRecord>. В этом классе определены функция бинарного поиска по записям в книге, функции добавления записи в книгу, которые при помощи функции двоичного поиска находят место для вставки (реализуя таким образом сортировку вставками), функции ввода телефонной книги с потока ввода или из файла, использующие функцию добавления записи, пока не считают терминальный элемент (нулевой номер или конец файла), функция распечатки, которая вызывает распечатку для каждой записи в векторе.

Класс PhoneRecord содержит член – структуру, содержащую два поля – телефонный номер (выделено 50 бит, так как этого хватит для представления 15-значного числа) и индикатор того, начинается ли номер с символа ‘+’. В этом классе определены операторы сравнения, функция фильтрации строки телефонного номера, которая проверяет каждый символ строки на то, является он ведущим плюсом или цифрой, «get’теры» и «set’теры» для имени и номера телефона, так как они находятся в поле спецификатора доступа protected, конструктор по умолчанию и инициализирующий конструктор, функция ввода записи с потока ввода, функция вывода записи, использующая функцию расширения строки inflate\_string.

inflate\_string добавлет указанные символы к строке так, чтобы исходная строка выравнивалась либо по левому, либо по правому краю, любо по центру.

При старте программа создает телефонную книгу, вызывает функцию ввода через стандартный поток ввода с консоли, затем вызывает функцию вывода телефонной книги на стандартный поток вывода на консоль.

# Блок-схема алгоритма

Начало / конец

условие

подпрограмма

Ввод / вывод

действие

Блок-схема нужна для представления алгоритма в виде последовательности операций и переходов между ними.

Основные вычислительные алгоритмы содержат повторяющиеся действия, при этом повторение не должно быть бесконечным – тогда программа никогда бы не заканчивалась и не позволяла получить результат. Существуют и длительно выполняющиеся программы - например программы игр или операционных систем, которые работают все время пока их не выключают. Это алгоритмы другого типа, по сути, не являющиеся вычислительными. В нашем курсе мы будем писать программы конечными, предназначенными для решения конкретной задачи, и когда результат получен, вычислительный алгоритм тоже заканчивается.

В наших блок-схемах появляются блоки НАЧАЛО и ЗАВЕРШЕНИЕ, по правилам оформления блок-схем [] это прямоугольники со скругленными углами. У НАЧАЛА один выход и ни одного входа, у ЗАВЕРШЕНИЯ – один вход и ни одного выхода. Другие блоки содержат один вход и как минимум один выход. Если у блока нет ни одного выхода, он «висит в воздухе», то программа, попав в этот блок останется там навсегда – проверяйте ваши схемы на наличие блоков, висящих в воздухе.

Блоки соединяются линиями, показывающими последовательность действий. Поэтому линия, соединяющая блоки, должна быть со стрелкой, показывающей откуда и куда переходит действие.

Для начала вычислений нужно ввести исходные данные, а в конце работы напечатать результат. Блоки ввода и вывода изображаются параллелограммами.

Наконец, в алгоритме бывают ветвления, то есть выполнение действий может идти, например по двум путям (простое условие). Блок проверки условия содержит описание проверки (что проверяется) и два выхода, которые подписываются «ДА» и «НЕТ» и соответствуют результату проверки.

# Выбор и обоснование типов переменных

В программе вычисления выполняются над переменными - числами, записанными в памяти компьютера, называемыми операндами. Например, С=А+В – простое выражение, которое означает взять число из ячейки А сложить его с числом из ячейки В и записать результат в ячейку С. В этой записи, несмотря на ее простоту, есть неопределенность для программы, заключающаяся в том, что числа в программе могут быть представлены в различных форматах, проще говоря занимать разное по размеру место в памяти машины, и это может существенно влиять на результат. В дальнейшем мы рассмотрим детально эти представления и правила преобразования типов, однако отчет должен точно описывать используемые в программе типы переменных для правильной работы с обоснованием использования выбранного типа. Например А, В. С – имеют целочисленный тип со знаком int.

# Вводимые и выводимые параметры и их типы

В этом разделе приводится полный перечень данных, которые программа получает, от оператора, в нашем случае, и через ввод с клавиатуры. Вывод результатов также должен быть описан, какие условные обозначения, сообщения и другую вспомогательную информацию может выдавать программа с необходимыми пояснениями. Например: в программе вводятся слагаемые А и В, оба представляют собой целые числа со знаком, не превышающие по модулю значения 200000000. Результат расчета выводится в строке «А+В=» и программа ожидает нажатия клавиши оператором для завершения.

# Структура проекта

Программа разрабатывается в современной среде разработки Microsoft (r) Visual Studio. Это очень хороший набор программ и вспомогательных средств для профессионального программирования. Процесс написания программы – это написание текста на одном из языков программирования, мы будем использовать языки С и С++. Сам текст является только средством для облегчения восприятия человеком последовательности машинных команд, которые «видит» машина в программе. Поэтому текст с языка С должен быть «переведен» на язык машинных команд, и поскольку этот перевод должен быть не вольным, а точным, процесс называется компиляцией, а выполняет его специальная программа – компилятор. Для нас важно, что тексты в программе – это текстовые файлы на языке С или С++, их имя может быть любым (избегайте использования кириллицы), а расширение .cpp для текста С++ или для классического С .c. Кроме файлов текста, которые обычно содержат тексты выполняемых операций в проекте есть файлы заголовков, которые в основном являются пояснениями компилятору, какие программист использует определения в своих текстах. Файлы заголовков имеют расширение .h от английского headers. Ну, и раз текстов уже несколько, должен быть файл проекта, который содержит в себе как минимум перечень файлов текста, файлов заголовков и еще инструкции в какую форму наш проект должен исходные файлы собрать. Расширение этого файла проекта .vcxproj от Visual C Extended Project. Этап сборки скомпилированных файлов и приводит в случае успеха к созданию исполняемого файла, с расширением .exe (executable). Исполняемый файл не универсален, он строится с учетом особенностей операционной системы и как правило не работает в другой ОС.

# Текст программы

//мой код:

# Копии экрана при работе программы

\*скриншоты (лучше win+shift+s в режиме окна)\*

# Контрольный пример, сравнение с эталоном

Ну тут типа информация по прохождению теста / тестов

# Результаты

К результатам работы относятся характеристики разработанной программы, такие, как: • тип ОС, для которой разработано приложение • тип приложения (консольное, оконное…) • путь, имя и размер исполняемого .exe файла • ограничения на исходные данные • результаты проверки выходных значений по независимому источнику • достоинства и недостатки программы