БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет Информационных Технологий

Дисциплина “Основы программной инженерии”

Отчёт к лабораторной работе №6 “Модульное программирование”

Выполнил:

Студент группы ПИ-7 Пастухов Александр Васильевич

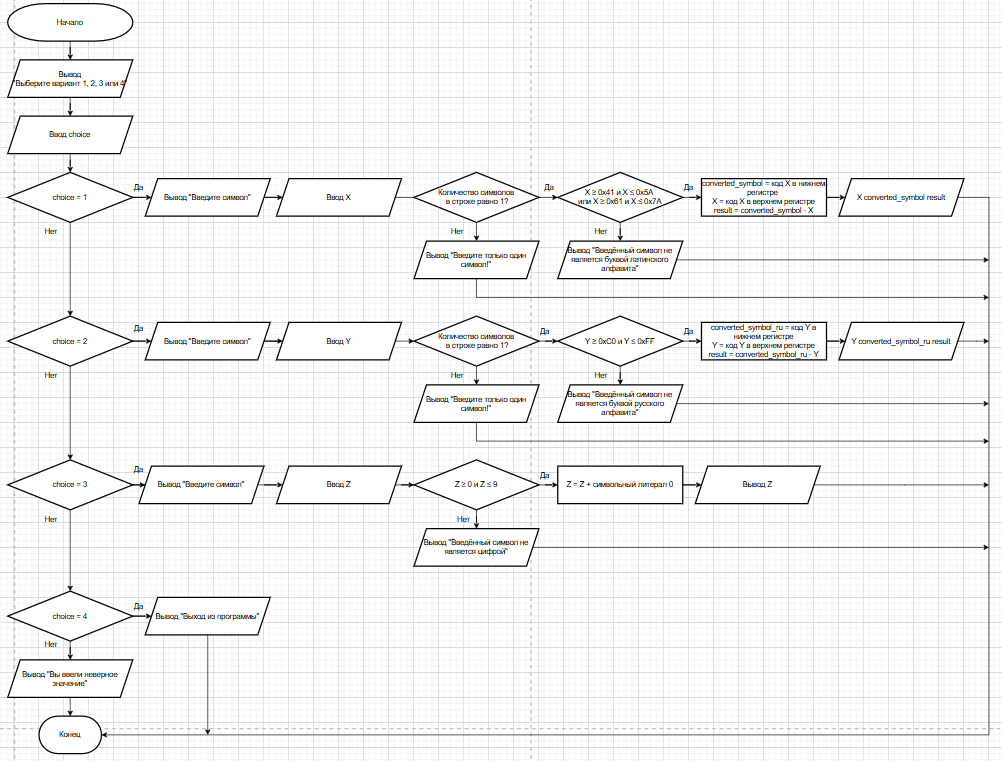
**2. Постановка задачи:**

**Среда разработки**: создать приложение на языке программирования С++ в интегрированной среде разработки Visual Studio.

**Задача:** написать диалоговую программу, которая в зависимости от выбранного варианта использования выполняет действия пунктов 1, 2, 3 для любого введенного с клавиатуры символа. Также необходимо предусмотреть возможность ввода нескольких символов последовательно.

**3. Входная информация**: программа принимает 2 символа из стандартного входного потока.

**Выходная информация**: программа выводит символ в верхнем регистре, символ в нижнем регистре и разницу значений кодов верхнего и нижнего регистра для букв латинского и русского алфавитов, а также выводит код символа, соответствующего введённой цифре.

**4. Блок-схема:**

**6. Модуль main.cpp**

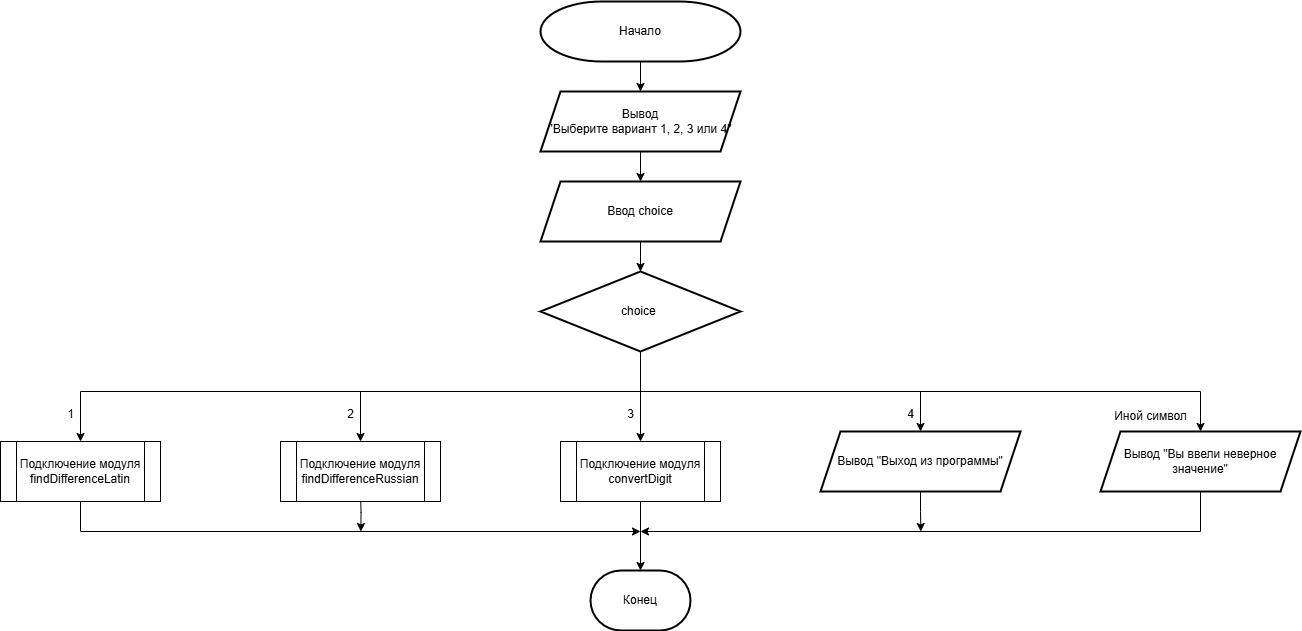
**Состав:** заголовочный файл main.h, модули findDifferenceLatin, findDifferenceRussian, convertDigit, оператор switch.

**Назначение:** предоставить пользователю консольный интерфейс для выбора одной из четырёх функций и выполнить соответствующую логику, реализованную в этих функциях.

**Входные данные:** модуль принимает 2 символа из стандартного входного потока.

**Выходные данные:** выходные данные находятся в других модулях.

**Алгоритм:**



**Модуль findDifferenceLatin**

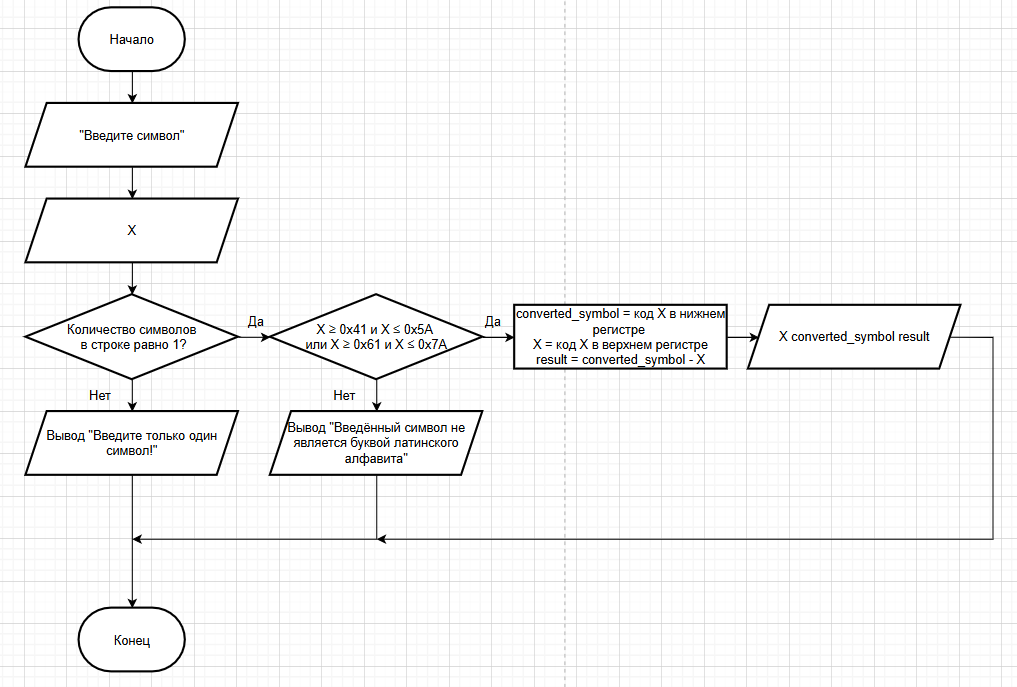
**Состав:** переменные X, converted\_symbol, result, функция peek, условный оператор if else, условный оператор if, заголовочный файл findDifferenceLatin.h.

**Назначение:** преобразовать букву латинского алфавита в нижний и верхний регистр и вычислить разницу их кодов в кодировке Windows-1251.

**Входные данные:** модуль принимает символ из стандартного входного потока.

**Выходные данные:** символ в нижнем и верхнем регистрах, а также разница нижнего и верхнего регистра в кодировке Windows-1251.

**Алгоритм:**



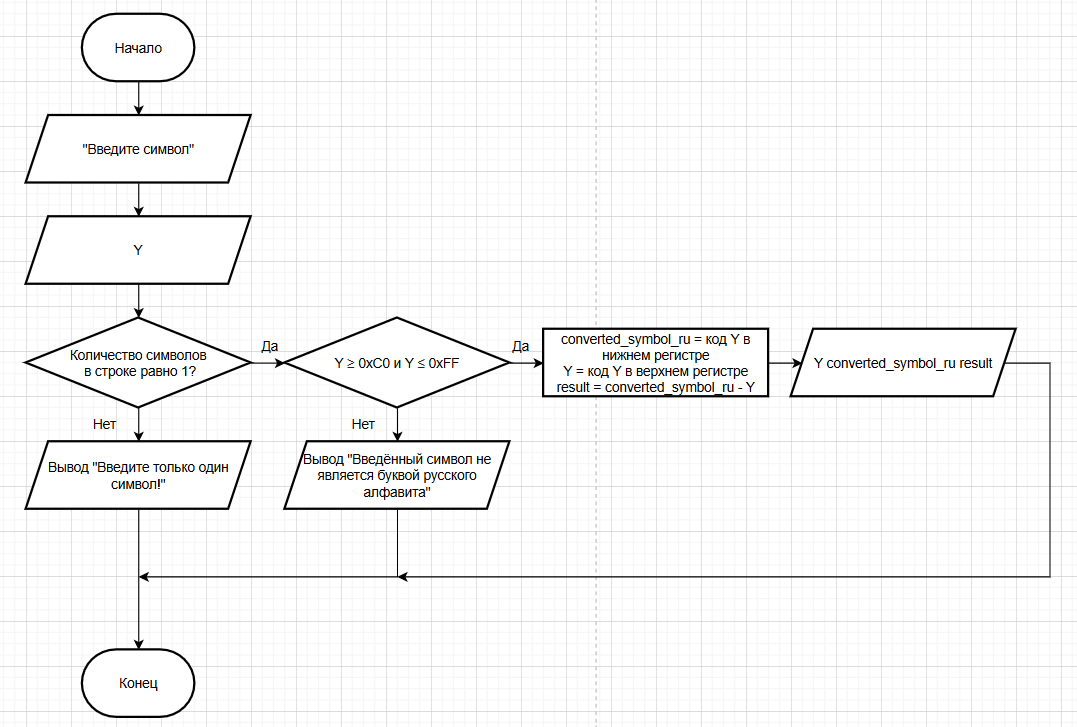
**Модуль findDifferenceRussian**

**Состав:** переменные Y, converted\_symbol\_ru, result, функция peek, условный оператор if else, условный оператор if, заголовочный файл findDifferenceRussian.h.

**Назначение:** преобразовать букву русского алфавита в нижний и верхний регистр и вычислить разницу их кодов в кодировке Windows-1251.

**Входные данные:** модуль принимает символ из стандартного входного потока.

**Выходные данные:** символ в нижнем и верхнем регистрах, а также разница нижнего и верхнего регистра в кодировке Windows-1251.

**Алгоритм:**

**Модуль convertDigit**

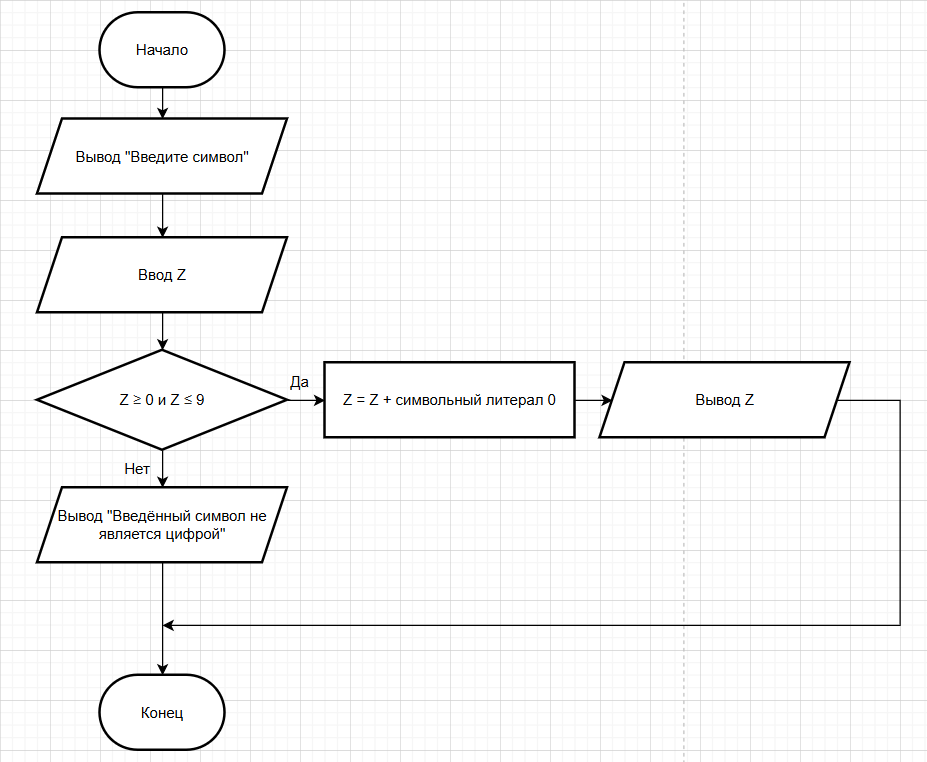
**Состав:** переменная Z, условный оператор if else, условный оператор if, заголовочный файл convertDigit.h.

**Назначение:** вывод в консоль кода символа, соответствующего введённой цифре.

**Входные данные:** модуль принимает символ из стандартного входного потока.

**Выходные данные:** код символа, соответствующий введённой цифре.

**Алгоритм:**



**8. Модульная схема программы:**

main

convertDigit

findDifferenceLatin

findDifferenceRussian

**Псевдокод:**

НАЧАЛО  
 ВЫВОД “Выберите вариант 1, 2, 3 или 4”  
 ВВОД choice  
 ЕСЛИ choice = 1, ТО ПОДКЛЮЧИТЬ findDifferenceLatin

ИНАЧЕ ЕСЛИ choice = 2, ТО ПОДКЛЮЧИТЬ findDifferenceRussian

ИНАЧЕ ЕСЛИ choice = 3, ТО ПОДКЛЮЧИТЬ convertDigit  
 ИНАЧЕ ЕСЛИ choice = 4, ТО ВЫВОД “Выход из программы”

ИНАЧЕ ВЫВОД “Вы ввели неверное значение”  
КОНЕЦ

**Ответы на вопросы:**

1. Парадигма программирования – это совокупность идей и понятий, определяющих стиль написания компьютерных программ (подход к программированию). Императивное программирование (Структурное программирование и Объектно-ориентированное программирование), Декларативное программирование (Функциональное программирование и Логическое программирование), Визуальное программирование.
2. Методология разработки программного обеспечения – совокупность методов, применяемых на различных стадиях жизненного цикла программного обеспечения.
3. **Императивное программирование** описывает **как** выполнить задачу, задавая последовательность шагов для достижения результата, а **Декларативное программирование** описывает **что** должно быть сделано, сосредотачиваясь на желаемом результате, а не на способе его достижения.
4. Цели: повысить надёжность программ, создание понятной, читаемой программы, которая исполняется, тестируется, конфигурируется, сопровождается и модифицируется без участия автора.  
   Основные принципы: программирование “сверху-вниз”, модульное программирование с иерархическим упорядочением связей между модулями/подпрограммами «От общего к частному».
5. Следование, ветвление, цикл.
6. Нисходящее проектирование, пошаговое проектирование, структурное проектирование, одновременное проектирование алгоритма и данных, модульное проектирование, модульное, нисходящее, пошаговое тестирование.
7. «Разделяй и властвуй».
8. Функциональная декомпозиция задачи – разбиение большой задачи на ряд более мелких, функционально самостоятельных подзадач – модулей.
9. Один вход и один выход, функциональная завершенность, логическая независимость, слабые информационные связи с другими программными модулями, обозримый по размеру и сложности программный элемент.
10. Методы нисходящего проектирования и методы восходящего проектирования.