Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий.

Специальность Программная инженерия.

1 курс, 10 группа, 1 подгруппа

Основы Программной инженерии

Выполнил: Лавринович Егор Владимирович

Лабораторная работа №6

Структурное программирование

**Задача:**

Написать диалоговую программу, которая в зависимости от выбранного варианта использования выполняет действие пунктов 1, 2, 3 для любого введенного символа

**Входные данные:**

- пункт выбора

- символы

**Выходные данные:**

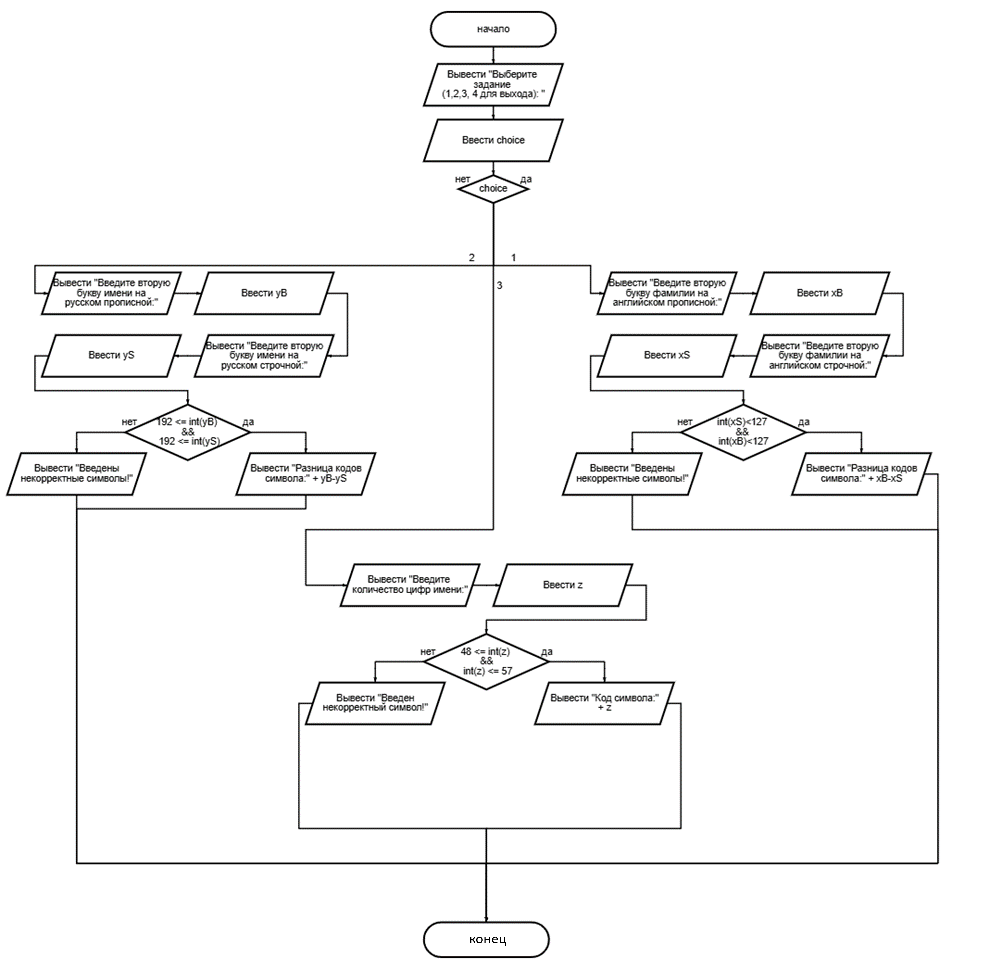
1 – определение разницы значений кодов в ASCII буквы в прописном и строчном написании, если введен символ латинского алфавита, иначе вывод сообщения об ошибке;

2 – определение разницы значений кодов в Windows-1251 буквы в прописном и строчном написании, если введен символ русского алфавита, иначе вывод сообщения об ошибке;

3 – вывод в консоль кода символа, соответствующего введенной цифре, иначе вывод сообщения об ошибке;

4 – выход из программы.

**Блок-схема:**



**Описание:**

- Ввод числа от 1 до 4 (каждое число отвечает за отдельный модуль), если choice не равно 1-4, то закончить программу

В зависимости от выбора choice ввод необходимых символов.

- Назначение: определение дальнейшего пути (какой модуль использовать)

Модуль 1: Проверка принадлежности символов к нужному типу, вывод код строчной введенной буквы минус код заглавной введенной буквы,

Модуль 2: Проверка принадлежности символов к нужному типу, вывод код строчной введенной буквы минус код заглавной введенной буквы

Модуль 3: Проверка принадлежности символов к нужному типу, вывод код введенного символа

**Псевдокод:**

Модуль 0(main):

НАЧАЛО

ВВОД choice

k == 1, ТОГДА ВВОД СИМВОЛОВ, res = модуль 1, ЕСЛИ res != 0, ТО ВЫВОД res, ИНАЧЕ ВЫВОД ошибки

k == 2, ТОГДА ВВОД СИМВОЛОВ, res = модуль 2, ЕСЛИ res != 0, ТО ВЫВОД res, ИНАЧЕ ВЫВОД ошибки

k == 3, ТОГДА ВВОД СИМВОЛА(ИЛИ СТРОКИ СИМВОЛОВ), res = модуль 3, ЕСЛИ res != 0, ТО ВЫВОД res, ИНАЧЕ ВЫВОД ошибки

k == 4 ТОГДА КОНЕЦ

ИНАЧЕ ВЫВОД ошибка

КОНЕЦ

Модуль 1:

НАЧАЛО

ЕСЛИ символы находятся в нужном диапазоне, ТОГДА ВЕРНУТЬ код символ1 – код символ2

ИНАЧЕ ВЕРНУТЬ 0

КОНЕЦ

Модуль 2:

НАЧАЛО

ЕСЛИ символы находятся в нужном диапазоне, ТОГДА ВЕРНУТЬ код символ1 - код символ2

ИНАЧЕ ВЕРНУТЬ 0

КОНЕЦ

Модуль 3:

НАЧАЛО

НАЧАЛО ЦИКЛА ОТ 0 ДО РАЗМЕРА СТРОКИ

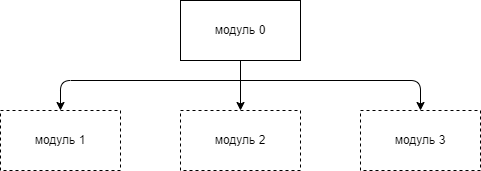
ЕСЛИ символ находятся в нужном диапазоне, ТОГДА ВЫВЕСТИ код символа

ИНАЧЕ ВЫВЕСТИ ошибку

КОНЕЦ ЦИКЛА

КОНЕЦ

**Модульная схема**



**1. Определение парадигмы программирования и основные парадигмы**

**Парадигма программирования** — это совокупность идей, принципов и подходов, определяющих стиль и метод организации программного кода. Основные парадигмы:

* **Императивное программирование** (например, процедурное, структурное)
* **Декларативное программирование** (например, функциональное, логическое)
* **Объектно-ориентированное программирование**
* **Функциональное программирование**
* **Логическое программирование**
* **Событийно-ориентированное программирование**

**2. Методология разработки программного обеспечения**

**Методология разработки ПО** — это структурированный набор методов, техник и принципов, помогающих организовать процесс разработки ПО.

**3. Отличие императивного программирования от декларативного**

* **Императивное программирование** описывает шаги, которые нужно выполнить, чтобы достичь результата. Оно сосредотачивается на последовательности действий.
* **Декларативное программирование** описывает, что должно быть достигнуто, а не как этого достичь. Это подход, где акцент сделан на конечный результат, а не на процесс.

**4. Цели и основные принципы структурного программирования**

**Цели структурного программирования**:

* Упрощение процесса написания и понимания кода.
* Снижение вероятности ошибок.
* Повышение модульности и читаемости.

**Основные принципы**:

* Использование базовых управляющих структур: последовательность, выбор, цикл.
* Избегание использования операторов перехода, таких как goto.

**5. Базовые конструкции для структурного программирования**

* **Последовательность**: линейное выполнение команд.
* **Условие (ветвление)**: выполнение кода в зависимости от условия (например, if, switch).
* **Цикл**: повторение выполнения кода (например, for, while).

**6. Методы структурного программирования**

* **Декомпозиция задачи на подзадачи**.
* **Использование функций и процедур** для разделения кода на модули.
* **Последовательное программирование без goto** для улучшения читаемости.

**7. Принцип модульного проектирования**

Основой модульного проектирования является **разделение системы на независимые или слабо связанные модули**, которые можно разрабатывать и тестировать отдельно.

**8. Функциональная декомпозиция задачи**

**Функциональная декомпозиция задачи** — это процесс разбиения сложной задачи на более мелкие и простые функции, которые легче анализировать и реализовывать.

**9. Основные характеристики модуля**

* **Самодостаточность**: возможность функционировать независимо.
* **Инкапсуляция**: скрытие внутренней реализации.
* **Переиспользуемость**: возможность повторного использования.
* **Низкая связность** и **высокая связность** (внутри модуля).

**10. Методы модульного программирования**

* **Функциональная декомпозиция**.
* **Инкапсуляция** данных и функций.
* **Использование интерфейсов** для взаимодействия между модулями.
* **Повторное использование кода**