**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**КАФЕДРА САПР**

отчет

**по лабораторной работе №1**

**по дисциплине «Программирование»**

Тема: **«если бы я знал)»**

|  |  |  |
| --- | --- | --- |
| Студент гр. 1335 |  | Максимов Ю.Е. |
| Преподаватель |  | Калмычков В.А. |

Санкт-Петербург

2023

Исходная формулировка задания

Заданы целочисленные значения. Вывести из файла в L1-список на основе массива, отсортировать на четные и нечетные, передать их двухсвязной очереди и вывести в файл.

1. Постановка задачи:
2. Разработать способ хранения целочисленного значения
3. Организовать в памяти динамическое размещение данных в виде L1-список на основе массива, двунаправленной очереди.
4. Обеспечить эффективное выполнение операция по сортировки на четное и нечетное
5. Реализовать передачу информации из одного объекта хранения в другой
6. Анализ вариантов хранения и обработки информации.

Так как целью является реализация задачи представления виде L1-список на основе массива, двунаправленной очереди, то вариант хранения элементов строки в статическом массиве, как мне кажется, уже будет являться неправильным решение. Наиболее удобным с точки зрения пользователя и программиста будет представить L1-список на основе массива, двунаправленной очереди в виде int динамического размера.

Задача: реализовать класс List, Vector, L1-список на основе массива, двунаправленной очереди в виде int динамического размера.

1. Описание методов.
   * 1. Vector.isEmpty() Проверка является ли контейнер пустым.
     2. Vector.size() Возвращает количество элементов в контейнере.
     3. Vector.capacity() Возвращает размерность контейнера.
     4. Vector.pushBack() добавляет значение в конец контейнера
     5. Vector.remove() удаляет значение по передаваемому индексу.
     6. Vector. pushFront() добавляет значение в начало контейнера.
     7. Vector.popFront() удаляет значение в начале контейнера.
     8. Vector.popBeck() удаляет значение в конце контейнера.
     9. Vector.insert() добавляет значение по индексу в контейнер.
     10. List.Move() Метод, возвращающий элемент в заданной позиции,
     11. List.CorrectIndex() Метод, проверяющий корректность позиии (индекса) в списке
     12. List.Del() Удалить элемент в заданной позиции
     13. List.GetElemenet() Получить элемент списка по индексу
     14. List.AddEnd() Добавить элемент в конец списка
     15. List.AddBegin() Добавить элемент в начало списка
     16. List.PopFront() Удалить из начала.
     17. List.PopBack() Удалить из конца.
     18. List.Clear() Очистка списка.
     19. List.Count() Получить количество элементов списка.
2. Тестирование
3. Файл строк *in.txt*

*“1*

*2*

*3*

*4*

*5*

*6*

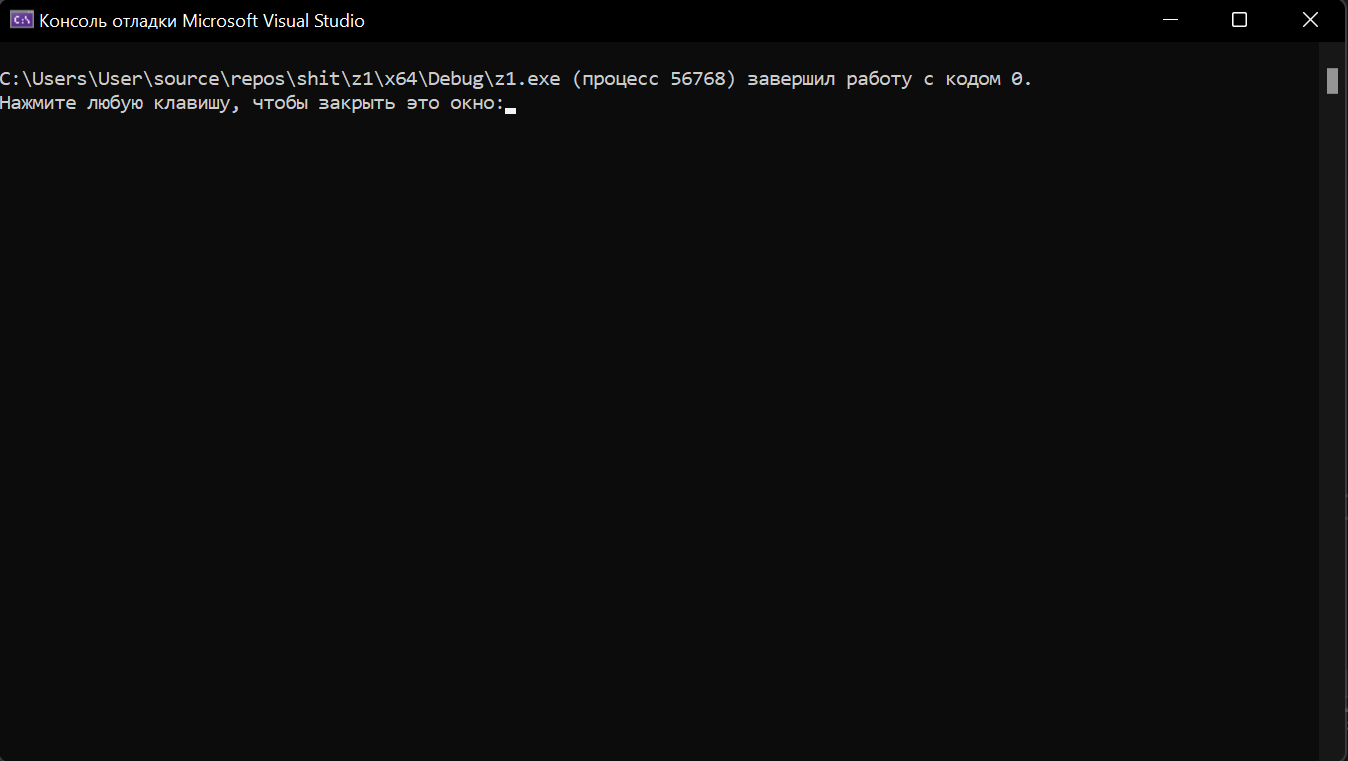
*7*

*8*

*9*

*10”*

Результат:



1. Запись результата в файл *out\_even\_numbers.txt и out\_odd\_numbers.txt*
2. Вывод:
3. Разработан способ хранения целочисленного типа.
4. Организовано динамическое размещение данных в виде L1-список на основе массива, двунаправленной очереди.
5. Обеспечено эффективное выполнение хранение и передачи информации (Vector, List)
6. Реализовано размещение любого количества данных L1-список на основе массива (void addMemory()).