МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського

«Харківський авіаційний інститут»

факультет програмної інженерії та бізнесу

кафедра інженерії програмного забезпечення

**Лабораторна робота № 1**

з дисципліни « Алгоритми і структури даних »

*назва дисципліни*

на тему: «БАЗОВІ ДИНАМІЧНІ СТРУКТУРИ ДАНИХ. ЛІНІЙНІ СПИСКИ»

Виконав: студент 2 курсу групи № 622п

освітньої програми

121 інженерія програмного забезпечення

(шифр і назва ОП)

Зайченко Ярослав Ігорович

(прізвище й ініціали студента)

Прийняв: канд. техн. наук, доцент на кафедрі 603 Волобуєва Ліна Олексіївна

(посада, науковий ступінь, прізвище й ініціали)

Кількість балів:

Харків – 2023

**Мета роботи** − познайомитися з базовими структурами даних (список, черга, стек) та отримати навички програмування алгоритмів, що їх обробляють.

Постановка задачі

1. Описати вказаний абстрактний тип даних (АТД) і основні операції роботи з ним на абстрактному і логічному рівнях:

* початкова ініціалізація структури (список порожній).
* додавання елементу в список (з урахуванням порядку, заданого у варіанті).
* видалення елементу зі списку (з урахуванням порядку, заданого у варіанті).
* перегляд першого елементу в списку.
* перевірка списку - порожній він або не порожній
* виведення структури на екран.
* виконання підрахунку числа елементів.
* реалізація функції вказаної у варіанті.

1. Відповідно до варіанту створити структуру даних для організації лінійного списку з використанням покажчиків. Створити додаток, який демонструє роботу операцій, перелічених вище. Всі операції потрібно визначати із заповненої структури даних. Дозволяється використовувати лише ті операції, що притаманні заданій структурі. Наприклад, заборонено отримувати доступ до елементу із довільною позицією у черзі.
2. Використовувати готові реалізації структур даних категорично забороняється.
3. Оформити звіт.

Зміст звіту

Зміст звіту з лабораторної роботи №1 має включати наступні підрозділи:

1. Абстрактне представлення структури.
2. Представлення даних в графічному виді.
3. Проектування необхідних класів і структур.
4. Алгоритм для кожного методу, використовуючи один з наступних видів представлення алгоритмів: псевдокод або блок-схему.
5. Лістинг програми та екранні форми результату роботи програми.
6. Висновки

Варіант 9 індивідуального завдання до лабораторної роботи 1:

Лінійний двонаправлений список символів. Вставку і видалення елементів робити за принципом черги. Реалізувати процедуру знаходження найбільшого з елементів.

хід роботи

Опис програми:

Мова програмування: С#, операційна система Windows 10 Prо, Версія 22H2, Збірка ОС 22621.1325, процесор: Apple Silicon M1 Pro 3.20 GHz (ядер: 4), компілятор: Microsoft Visual Studio Community 2022 (64-розрядна версія ARM).

Абстрактне представлення структури

* 1. Додавання елементу (EnQueue): ∅ → Head x Tail;
  2. Видалення елементу (DeQueue): Tail x el → Head x Tail;
  3. Перевірка існування (Empty): Head → bool;
  4. Перший елемент (Peek): Head → el (char)
  5. Вивести список на екран: Head to Tail → string
  6. Підрахувати кількість елементів: Head to Tail → int
  7. Знайти найбільший елемент: Head to Tail → Max (char)

Графічне представлення черги

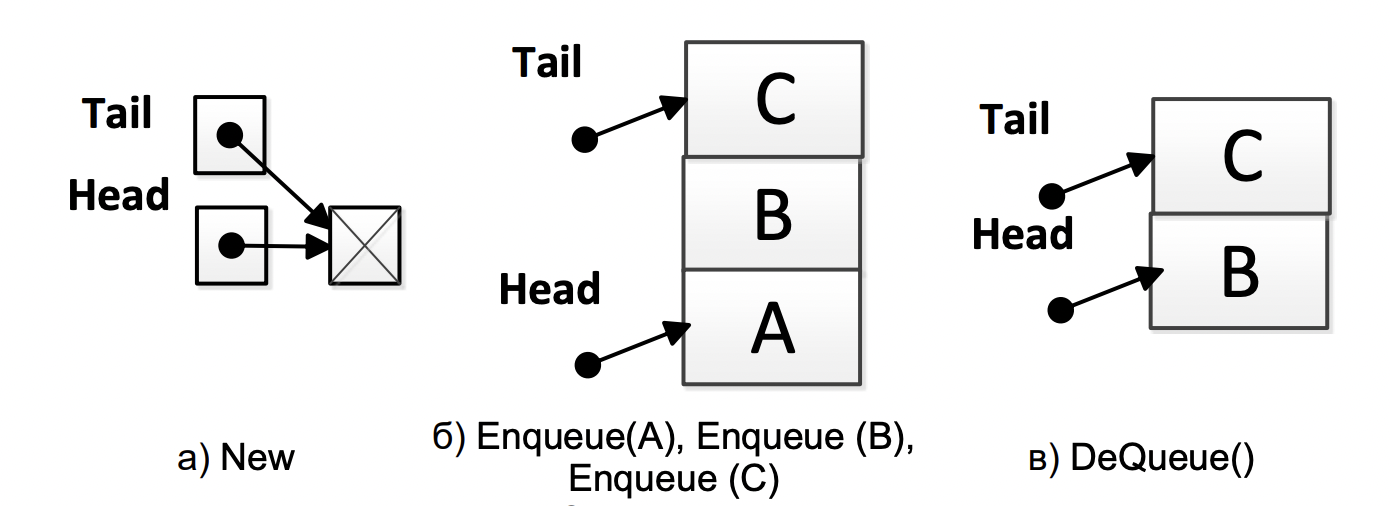


Рисунок 1 – Основні операції з чергою

Проектування необхідних класів і структур.

Клас Elem потрібен для ініціалізації елементів в черзі; містить інформацію про попередній та наступний елемент.

public class Elem

{

public Elem \_L, \_R;

public char \_inf;

public Elem(char inf)

{

this.\_inf = inf;

}

}

Клас Queue містить в собі інформацію про перший та останній елементи черги. Також містить методи для реалізації дій із даними, які зазначені у постановці завдання (фрагмент коду наведено нижче).

public class Queue

{

Elem \_Tail;

Elem \_Head;

public void EnQueue(char inf)

public char DeQueue()

public char FindMax()

public override string ToString()

public bool IsEmpty()

public int Count()

public char Peek()

}

Методи зображені у вигляді блок-схем

(Див. рисунки 2 – ). Вони мають назви відповідно до їхнього функціоналу.

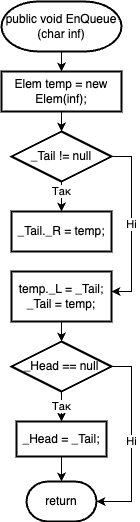


Рисунок 2 – Додавання символу в чергу

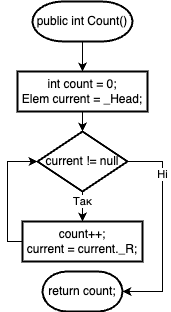


Рисунок 3 – метод для рахунку кількості символів у черзі

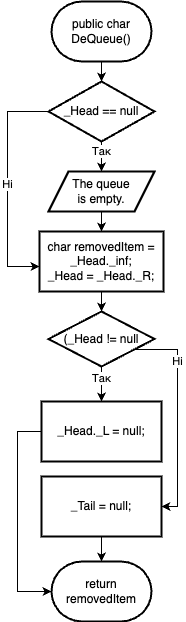


Рисунок 4 – Видалення символу з черги

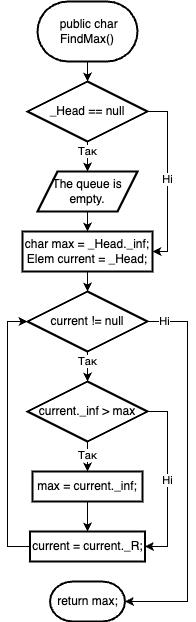


Рисунок 5 – Знаходження максимального елементу в черзі

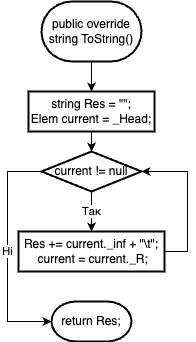


Рисунок 6 – Метод для перетворення елементів черги в рядковий тип

Тестування програми

Додавання елементів (1 a 2 b 3 c) до черги:

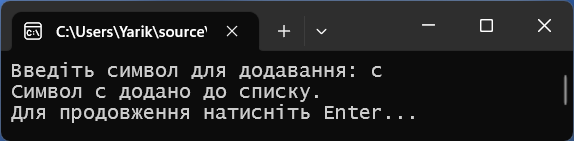


Рисунок 7 – Процес додавання елементів до черги

Переглянемо створену чергу символів. Рисунок 8.

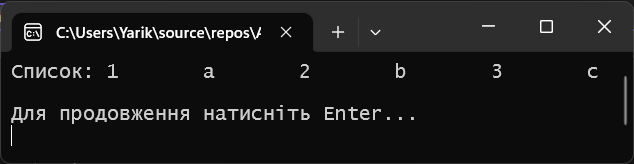


Рисунок 8 – Виведення на екран створеної черги символів

Видаляємо перший елемент в черзі та виводимо її знову на екран (Рисунки 9 - 10)

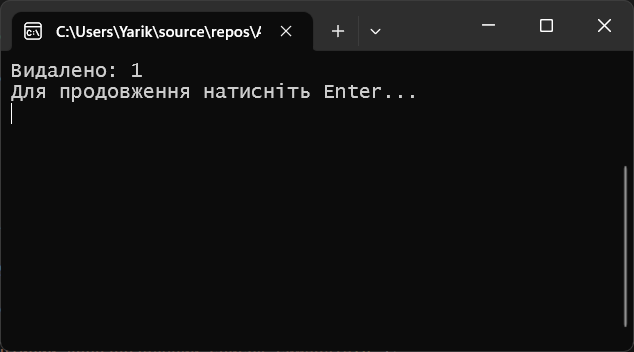


Рисунок 9 – Видалення першого елементу

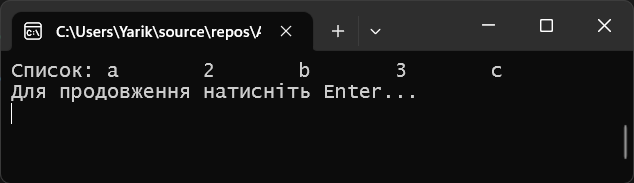


Рисунок 10 – Нова черга з видаленим першим елементом

Перевіримо, який найбільший елемент в черзі. Результат на Рисунку 11

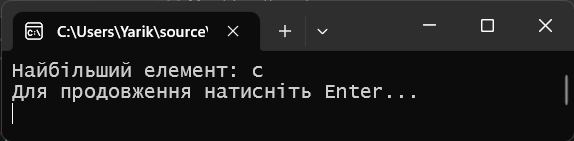


Рисунок 11 – Виведений на екран найбільший елемент черги

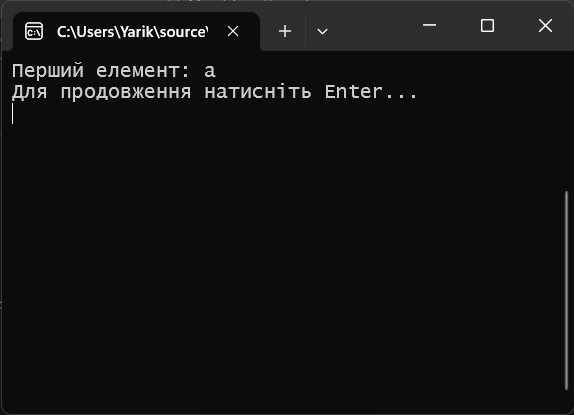


Рисунок 12 – Виведений на екран перший елемент черги без видалення

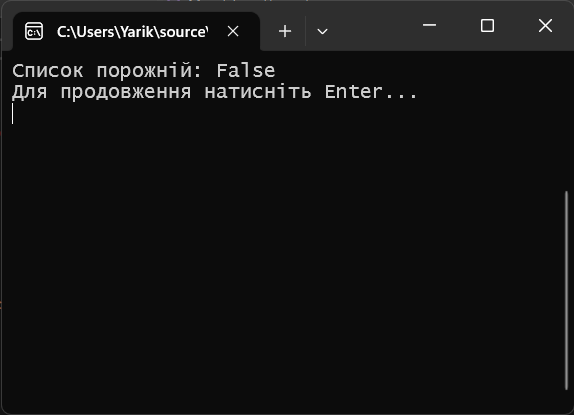


Рисунок 13 – Виведений на екран результат перевірки наповненості черги

Порахуємо кількість елементів в списку. Результат на Рисунку 14.

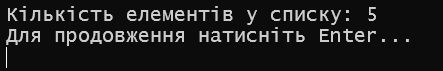


Рисунок 14 – кількість елементів в черзі

Результат співпадає з ручним рахунком.

Після видалення всіх елементів черги при перевірці наповнення програма повідомляє, що черга порожня. Демонстрація на рисунку 15.

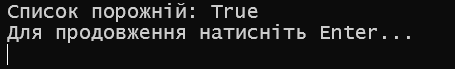


Рисунок 15 – перевірка елементів в пустій черзі

ВИСНОВОК

У ході виконання лабораторної роботи я:

* Ознайомився з базовими структурами даних;
* Отримав навички програмування алгоритмів, що їх оброблюють;
* Навів приклади роботи з чергою;
* Написав програму для виконання операцій з чергою;
* Виконав тестування написаної програми;
* Написав звіт.

ДОДАТОК А

Машинний лістинг програми

Program.cs:

using System;

using System.Text;

namespace ASD\_Lab\_1

{

internal class Program

{

static void PressEnter()

{

while (true)

{

Console.WriteLine("Для продовження натисніть Enter...");

var key = Console.ReadKey(intercept: true);

if (key.Key == ConsoleKey.Enter)

{

Console.Clear();

break; // Выход из цикла, если нажата клавиша Enter

}

else Console.WriteLine("Була натиснута інша клавіша");

Console.Clear();

}

}

static void Main(string[] args)

{

Encoding.RegisterProvider(CodePagesEncodingProvider.Instance);

Console.InputEncoding = System.Text.Encoding.GetEncoding(1251);

Console.OutputEncoding = System.Text.Encoding.GetEncoding(1251);

Queue list = new Queue();

char choice;

do

{

Console.WriteLine("Лінійний двонаправлений список символів");

Console.WriteLine("\nМеню:");

Console.WriteLine("1. Додати символ до списку");

Console.WriteLine("2. Видалити перший символ зі списку");

Console.WriteLine("3. Знайти найбільший символ у списку");

Console.WriteLine("4. Переглянути перший символ у списку");

Console.WriteLine("5. Перевірити, чи список порожній");

Console.WriteLine("6. Вивести структуру на екран");

Console.WriteLine("7. Підрахувати кількість елементів у списку");

Console.WriteLine("0. Вийти");

Console.Write("Введіть номер операції: ");

choice = Console.ReadKey().KeyChar;

Console.WriteLine();

Console.Clear();

try

{

switch (choice)

{

case '1':

Console.Write("Введіть символ для додавання: ");

char itemToAdd = Console.ReadKey().KeyChar;

list.EnQueue(itemToAdd);

Console.WriteLine("\nСимвол " + itemToAdd + " додано до списку.");

PressEnter();

break;

case '2':

char removedItem = list.DeQueue();

Console.WriteLine("Видалено: " + removedItem);

PressEnter();

break;

case '3':

char maxItem = list.FindMax();

Console.WriteLine("Найбільший елемент: " + maxItem);

PressEnter();

break;

case '4':

char firstItem = list.Peek();

Console.WriteLine("Перший елемент: " + firstItem);

PressEnter();

break;

case '5':

bool isEmpty = list.IsEmpty();

Console.WriteLine("Список порожній: " + isEmpty);

PressEnter();

break;

case '6':

Console.Write("Список: ");

Console.WriteLine(list.ToString());

PressEnter();

break;

case '7':

int itemCount = list.Count();

Console.WriteLine("Кількість елементів у списку: " + itemCount);

PressEnter();

break;

case '0':

break;

default:

Console.WriteLine("Неправильний вибір. Спробуйте ще раз.");

break;

}

}

catch (InvalidOperationException ex)

{

Console.WriteLine("Помилка: " + ex.Message);

}

} while (choice != '0');

}

}

}

ДОДАТОК Б

Машинний лістинг створеного класу

Class1.cs

using System;

namespace ASD\_Lab\_1

{

public class Elem

{

public Elem \_L, \_R;

public char \_inf;

public Elem(char inf)

{

this.\_inf = inf;

}

}

public class Queue

{

Elem \_Tail;

Elem \_Head;

public void EnQueue(char inf)

{

Elem temp = new Elem(inf);

if (\_Tail != null)

{

\_Tail.\_R = temp;

}

temp.\_L = \_Tail;

\_Tail = temp;

if (\_Head == null)

{

\_Head = \_Tail;

}

}

public char DeQueue()

{

if (\_Head == null)

{

throw new InvalidOperationException("The queue is empty.");

}

char removedItem = \_Head.\_inf;

\_Head = \_Head.\_R;

if (\_Head != null)

{

\_Head.\_L = null;

}

else

{

\_Tail = null;

}

return removedItem;

}

public char FindMax()

{

if (\_Head == null)

{

throw new InvalidOperationException("The queue is empty.");

}

char max = \_Head.\_inf;

Elem current = \_Head;

while (current != null)

{

if (current.\_inf > max)

{

max = current.\_inf;

}

current = current.\_R;

}

return max;

}

public override string ToString()

{

string Res = "";

Elem current = \_Head;

while (current != null)

{

Res += current.\_inf + "\t";

current = current.\_R;

}

return Res;

}

public bool IsEmpty()

{

return \_Head == null;

}

public int Count()

{

int count = 0;

Elem current = \_Head;

while (current != null)

{

count++;

current = current.\_R;

}

return count;

}

public char Peek()

{

if (\_Head == null)

{

throw new InvalidOperationException("The queue is empty.");

}

return \_Head.\_inf;

}

}

}