Федеральное агентство связи

Государственное бюджетное образовательное учреждение высшего образование

Ордена Трудового Красного Знамени

«Московский технический университет связи и информатики»

Кафедра «МКиИТ»

дисциплина «СиАОД»

Отчет по Лабораторной работе №4

Подготовила студентка

группы БВТ1901: Нкурикийе Х.

Проверил: Мелехин А.

Москва 2021

Задания:

Реализовать следующие структуры данных:

● Стек (stack): операции для стека: инициализация, проверка на пустоту, добавление нового элемента в начало, извлечение элемента из начала;

● Дек (двусторонняя очередь, deque): операции для дека: инициализация, проверка на пустоту, добавление нового элемента в начало, добавление нового элемента в конец, извлечение элемента из начала, извлечение элемента из конца.

1. Отсортировать строки файла, содержащие названия книг, в алфавитном порядке с использованием двух деков

2. Дек содержит последовательность символов для шифровки сообщений. Дан текстовый файл, содержащий зашифрованное сообщение. Пользуясь деком, расшифровать текст. Известно, что при шифровке каждый символ сообщения заменялся следующим за ним в деке по часовой стрелке через один.

3. Даны три стержня и n дисков различного размера. Диски можно надевать на стержни, образуя из них башни. Перенести n дисков со стержня A на стержень C, сохранив их первоначальный порядок. При переносе дисков необходимо соблюдать следующие правила: - на каждом шаге со стержня на стержень переносить только один диск; - диск нельзя помещать на диск меньшего размера; - для промежуточного хранения можно использовать стержень B. Реализовать алгоритм, используя три стека вместо стержней A, B, C. Информация о дисках хранится в исходном файле.

4. Дан текстовый файл с программой на алгоритмическом языке. За один просмотр файла проверить баланс круглых скобок в тексте, используя стек.

5. Дан текстовый файл с программой на алгоритмическом языке. За один просмотр файла проверить баланс квадратных скобок в тексте, используя дек.

6. Дан файл из символов. Используя стек, за один просмотр файла напечатать сначала все цифры, затем все буквы, и, наконец, все остальные символы, сохраняя исходный порядок в каждой группе символов.

7. Дан файл из целых чисел. Используя дек, за один просмотр файла напечатать сначала все отрицательные числа, затем все положительные числа, сохраняя исходный порядок в каждой группе.

8. Дан текстовый файл. Используя стек, сформировать новый текстовый файл, содержащий строки исходного файла, записанные в обратном порядке: первая строка становится последней, вторая – предпоследней и т.д.

9. Дан текстовый файл. Используя стек, вычислить значение логического выражения, записанного в текстовом файле в следующей форме: < ЛВ > ::= T | F | (N) | (A) | (X) | (O), где буквами обозначены логические константы и операции: T – True, F – False, N – Not, A – And, X – Xor, O – Or

10. Дан текстовый файл. В текстовом файле записана формула следующего вида: ::= | M(,) | N(Формула>,) < Цифра > ::= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 где буквами обозначены функции: M – определение максимума, N – определение минимума. Используя стек, вычислить значение заданного выражения.

11. Дан текстовый файл. Используя стек, проверить, является ли содержимое текстового файла правильной записью формулы вида: < Формула > ::= < Терм > | < Терм > + < Формула > | < Терм > - < Формула > < Терм > ::= < Имя > | (< Формула >) < Имя > ::= x | y | z

Код вызова функций:

```
using System;
using System.Collections.Generic;
using System.IO;

namespace Лабораторная_работа__4
{
   class Program
   {

      static MyDeque<Char> MyDec2 = new MyDeque<char>(34);


      static int size = 10;
      static void Main(string[] args)
      {
```

```
Console.WriteLine("Task1 ");
List<String> Books = new List<string>();
using (StreamReader sr = new StreamReader("task1.txt"))
{
   for (int i = 0; i < size; i++)
   {
      Books.Add(sr.ReadLine());
   }
}
Task1(Books);

Console.WriteLine("\r\n" + "Task2");
using (StreamReader sr = new StreamReader("task2.txt"))
{
   String MyStr2 = sr.ReadToEnd();
   CreateDec2();
   for (int i = 0; i < MyStr2.Length; i++)
   {
      Console.Write(Decode(MyStr2[i]));
   }
}



Console.WriteLine("\r\n" + "Task3");
int[] Disks;
using (StreamReader sr = new StreamReader("task3.txt"))
{
```

```
            Disks = Array.ConvertAll(sr.ReadToEnd().Split(new char[] { ' ' },
StringSplitOptions.RemoveEmptyEntries), new Converter<String, int>(Convert.ToInt32));
        }

        MyStack<int> Stack3A = new MyStack<int>(Disks.Length + 1);
        MyStack<int> Stack3B = new MyStack<int>(Disks.Length + 1);
        MyStack<int> Stack3C = new MyStack<int>(Disks.Length + 1);

        for (int i = Disks.Length-1; i >= 0; i--)
        {
           Stack3A.Push(Disks[i]);
        }

        while (Stack3C.Count != Disks.Length)
        {
           move(Stack3A, Stack3B);
           move(Stack3A, Stack3C);
           move(Stack3B, Stack3C);
        }

        while (Stack3C.Count != 0)
        {
           Console.WriteLine(Stack3C.Pop());
        }

        Console.WriteLine("\r\n" + "Task4");
        String Str4 = "";
        using (StreamReader sr = new StreamReader("task4.txt"))
        {
```

```
            Str4 = sr.ReadToEnd();
        }
        Task4(Str4);

        Console.WriteLine("\r\n" + "Task5");
        String Str5 = "";
        using (StreamReader sr = new StreamReader("task5.txt"))
        {
            Str5 = sr.ReadToEnd();
        }
        Task55(Str5);

        Console.WriteLine("\r\n" + "Task6");
        String Str6 = "";
        using (StreamReader sr = new StreamReader("task6.txt"))
        {
            Str6 = sr.ReadToEnd();
        }
        Task6(Str6);

        Console.WriteLine("\r\n" + "\r\n" + "Task7");
        int[] Numbers;
        using (StreamReader sr = new StreamReader("task7.txt"))
        {
            Numbers = Array.ConvertAll(sr.ReadToEnd().Split(new char[] { ' ' },
StringSplitOptions.RemoveEmptyEntries), new Converter<String, int>(Convert.ToInt32));
            Task7(Numbers);
        }
```

```
Console.WriteLine("\r\n" + "\r\n" + "Task8");
String[] Strings;
using (StreamReader sr = new StreamReader("task8.txt"))
{
   Strings = sr.ReadToEnd().Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);
}
Task8(Strings);

Console.WriteLine("\r\n" + "Task9");
String String9 = "";
using (StreamReader sr = new StreamReader("task9.txt"))
{
   String9= sr.ReadToEnd();
}

String op = "XOAN";
MyStack<bool> Zn = new MyStack<bool>(String9.Length);
MyStack<char> Ops = new MyStack<char>(String9.Length);

for (int i = String9.Length - 1; i >= 0; i--)
{
   if (op.Contains(Convert.ToString(String9[i])))
   {
      if (String9[i + 1] == '(') Ops.Push(Char.ToLower(String9[i]));
      else Ops.Push(String9[i]);
   }
   else if (String9[i] == 'T')
   {
      Zn.Push(true);
```

```
    }
    else if (String9[i] == 'F')
    {
        Zn.Push(false);
    }
}

Console.WriteLine(!(true & false));
Console.WriteLine(Task9(Zn,Ops));



Console.WriteLine("\r\n" + "Task10");
String String10 = "";
using (StreamReader sr = new StreamReader("task10.txt"))
{
    String10 = sr.ReadToEnd();
}
MyStack<char> St10 = new MyStack<char>(String10.Length+1);
if (String10[0] == '(' && String10[String10.Length - 1] == ')') String10 = String10.Substring(1,
String10.Length - 2);

for(int i = 0; i < String10.Length; i++)
{
    if(String10[i]==',' && (String10[i-1] == ')' || String10[i + 1] == '('))
    {
        String str1 = String10.Substring(0, i);
        String str2 = String10.Substring(i+1, String10.Length-i-1);
        String10 = str1 + str2;
```

```
    }
}

for (int i = String10.Length - 1; i >= 0; i--)
{
    if (String10[i] != ' ') St10.Push(String10[i]);
}
Console.WriteLine(Task10(St10));

Console.WriteLine("\r\n" + "Task11");
String String11 = "";
using (StreamReader sr = new StreamReader("task11.txt"))
{
    String11 = sr.ReadToEnd();
}
MyStack<char> St11 = new MyStack<char>(String11.Length);
if(String11[0]=='(' && String11[String11.Length-1] == ')')
{
    String11 = String11.Substring(1, String11.Length - 2);
}
for(int i=String11.Length-1;i>=0;i--)
{
    if (String11[i] != ' ') St11.Push(String11[i]);
}
Task11(St11);
Console.ReadLine();
```

```
    Console.ReadLine();
}
```

Задание 1

```
#region Task1
//заполняем первый стек, что меньше то влево, что больше то вправо
//если необходимо вставить элемент то в цикле вытаскиваем все элементы, вставляем и возвращаем остальные

static void Task1(List<String> books)
{
    MyDeque<String> q1 = new MyDeque<string>(books.Count + 1);
    MyDeque<String> q2 = new MyDeque<string>(books.Count + 1);

    for(int i = 0; i < books.Count; i++)
    {
        if (q1.Count == 0) { q1.PushBack(books[i]); }
        else if (String.Compare(books[i], q1.PeekBack()) == 1)
        {
            q1.PushBack(books[i]);
        }
        else if (String.Compare(books[i], q1.PeekFront()) == -1)
        {
            q1.PushFront(books[i]);
        }
        else
        {
            while(String.Compare(books[i], q1.PeekFront()) == 1)
```

```
            {
                q2.PushBack(q1.PopFront());
            }

            q1.PushFront(books[i]);

            while (q2.Count != 0)
            {
                q1.PushFront(q2.PopBack());
            }
        }
    }
    q1.PrintDec();
}

#endregion
```

Задание 2

```
#region Task2
```

//перебираем все символы в спомогательном деке, когда находи нужный, перекидываем его вперед и возвращаем следующий за ним

```
static void CreateDec2( )
{
    String Str = "qwertyuiopasdfghjklzxcvbnm";
    for (int i = 0; i < Str.Length; i++)
    {
        if(Char.IsLetter(Str[i]))
            MyDec2.PushBack(Str[i]);
    }
```

```
}

static Char Decode(char ch)
{
   for (int i = 0; i < MyDec2.Count; i++)
   {
      Char x = MyDec2.PopBack();
      if (ch == x)
      {
         MyDec2.PushFront(x);
         Char Value = MyDec2.PopBack();
         MyDec2.PushFront(Value);
         return Value;
      }

      MyDec2.PushFront(x);
   }

   return ch;
}
#endregion
```

Задание 3

```
#region Task3

//в цикле по три действия перекидываем с 1 на 2, на 3 и с 2 на 3
static void move(MyStack<int >a, MyStack<int> b)
{
```

```
    if (a.Count > 0 && b.Count == 0) b.Push(a.Pop());
    else if (a.Count == 0 && b.Count > 0) a.Push(b.Pop());
    else if (a.Count > 0 && b.Count > 0 )
    {
        if (a.Peek() < b.Peek()) b.Push(a.Pop());
        else a.Push(b.Pop());
    }
}



#endregion
```

Задание 4

```
#region Task4
// добавляем в стек одну скобку, когда находим другую удаляем
static void Task4(String str)
{
    MyStack<char> Stack3 = new MyStack<char>(str.Length);

    for(int i = 0; i < str.Length; i++)
    {
        if (str[i] == '(')
        {
            Stack3.Push('(');
        }
        else if(str[i] == ')')
        {
            if (Stack3.Count == 0)
            {
```

```
            Console.WriteLine("Quantity'(' != quantity ')'");
            return;
          }
          Stack3.Pop();
        }
      }

      if (Stack3.Count == 0) Console.WriteLine("quantity'(' = quantity ')'");
      else Console.WriteLine("quantity '(' != quantity ')'");
    }

    #endregion
```

Задание 5

```
    #region Task5

    static void Task5(String str)
    {
      MyDeque<char> Deque3 = new MyDeque<char>(str.Length+1);

      for (int i = 0; i < str.Length; i++)
      {
        if (str[i] == '[')
        {
          Deque3.PushBack('[');
        }
        else if (str[i] == ']')
        {
          if (Deque3.Count == 0)
```

```
            {
                Console.WriteLine("quantity '[' != quantity ']'");
                return;
            }
            Deque3.PopBack();
        }
    }

    if (Deque3.Count == 0) Console.WriteLine("quantity '[' != quantity ']'");
    else Console.WriteLine("quantity '[' != quantity ']'");
}



static void Task55(String str)
{
    MyDeque<char> Deque3 = new MyDeque<char>(str.Length + 1);

    for (int i = 0; i < str.Length; i++)
    {
        if (str[i] == '[') Deque3.PushFront('[');
        else if (str[i] == ']') Deque3.PushBack(']');
    }

    int count = 0;

    while (true)
    {
        char ch = Deque3.PopFront();
        if (ch == '[') count++;
```

```
            else break;
        }

        if (count == str.Length-count) Console.WriteLine("quantity '[' = quantity ']'");
        else Console.WriteLine("quantity '[' != quantity ']'");
    }

    #endregion
```

Задание 6

```
    #region Task6
    // разбрасываем данные по трем стекам и выводим
    static void Task6(String str)
    {
        MyStack<char> Digits = new MyStack<char>(str.Length);
        MyStack<char> Letters = new MyStack<char>(str.Length);
        MyStack<char> Others = new MyStack<char>(str.Length);

        for (int i = str.Length-1; i >= 0; i--)
        {
            if(str[i]>='0' && str[i] <= '9')
            {
                Digits.Push(str[i]);
            }
            else if ((str[i] >= 'a' && str[i] <= 'z') || (str[i] >= 'A' && str[i] <= 'Z') || (str[i] >= 'а' && str[i] <= 'я')
|| (str[i] >= 'А' && str[i] <= 'Я'))
            {
                Letters.Push(str[i]);
```

```
            }
            else
            {
                Others.Push(str[i]);
            }
        }



        while (Digits.Count > 0)
        {
            Console.Write(Digits.Pop());
        }

        while (Letters.Count > 0)
        {
            Console.Write(Letters.Pop());
        }

        while (Others.Count > 0)
        {
            Console.Write(Others.Pop());
        }
    }

    #endregion
```

Задание 7

```
#region Task7
//заполняем дек числами отрицательными и положительными, переворачиваем и выводим
static void Task7(int[] arr)
{
   MyDeque<int> Dec4 = new MyDeque<int>(arr.Length+1);

   for(int i =0; i < arr.Length; i++)
   {
      if (arr[i] < 0)
      {
         Dec4.PushFront(arr[i]);
      }
      else Dec4.PushBack(arr[i]);
   }

   for (int i=1;i<arr.Length;i++)
   {
      int x = Dec4.PopFront();
      if (x < 0)
      {
         Dec4.PushBack(x);
      }
      else
      {
         Dec4.PushFront(x);
         break;
      }
   }
```

```
    while (true)
    {
        int x = Dec4.PopBack();
        if (x < 0) Console.Write(x + " ");
        else
        {
            Dec4.PushBack(x);
            break;
        }
    }

    while (Dec4.Count > 0)
    {
        Console.Write(Dec4.PopFront()+" ");
    }

}

#endregion
```

Задание 8

```
#region Task8
// просто заполняем и выводим
static void Task8(String[] arr)
{

    MyStack<String> Stack8 = new MyStack<String>(arr.Length+1);

    for(int i = 0; i < arr.Length; i++)
```

```
        {
            Stack8.Push(arr[i]);
        }

        while (Stack8.Count != 0)
        {
            Console.WriteLine(Stack8.Pop());
        }

    }

    #endregion
```

Задание 9

```
    #region Task9
    //преусматриваются случаи, когда отрицание перед буквой и перед скобкой, когда простое действие или межу двуся скобками
    //если обычное отрицание перед буквой , вытаскиваем ее, выполняем действие,добавляем результат
    //если перед скобкой, вытаскиваем две буквы и следующую операцию, выполняем 2 операции, добавляем результат назад
    //если обычная операция, вытаскиваем две буквы, выполняем, добавляем назад
    //если операция перед скобкой выполняем рекрсию
    static bool Task9(MyStack<bool> Zn, MyStack<char> Ops)
    {
        if (Zn.Count == 1)
        {
            return Zn.Pop();
        }
        else
```

```
{
   String op = "XOAN";

   while (Ops.Count != 0)
   {
      char CurOp = Ops.Pop();

      if (CurOp == 'N')
      {
         bool res = !Zn.Pop();
         Zn.Push(res);
      }
      else if(CurOp == 'n')
      {
         char NextOp = Ops.Pop();
         bool one = Zn.Pop();
         bool two = Zn.Pop();
         bool result=true;

         if (NextOp == 'A') result = !(one & two);
         else if(NextOp == 'O') result = !(one | two);
         else if(NextOp == 'X') result = !(one ^ two);

         Zn.Push(result);
      }
      else if(op.Contains(Convert.ToString(CurOp)))
      {
         bool one = Zn.Pop();
         bool two = Zn.Pop();
```

```
            bool result = true;

            if (CurOp == 'A') result = one & two;
            else if (CurOp == 'O') result = one | two;
            else if (CurOp == 'X') result = one ^ two;

            Zn.Push(result);
        }
        else
        {
            bool one= Zn.Pop();
            bool two = Task9(Zn, Ops);
            bool result = true;

            if (CurOp == 'a') result = one & two;
            else if (CurOp == 'o') result = one | two;
            else if (CurOp == 'x') result = one ^ two;
            Zn.Push(result);
        }
      }

      return Zn.Pop();
   }


}

#endregion
```

Задание 10

```
#region Task10
//рекурсивно находим значение и возвращаем
static int Task10(MyStack<char> St)
{

   if (St.Count == 1 && Char.IsDigit(St.Peek())) return St.Pop();
   else
   {
      int num1, num2, result;
      char op = St.Pop();
      St.Pop();// убираем скобку перед операцией



      if (Char.IsDigit(St.Peek()))
      {
         int c = (int)St.Pop()- (int)'0';
         num1 = c;
         St.Pop();
      }
      else
      {
         num1 = Task10(St);
      }



      if (Char.IsDigit(St.Peek()))
```

```
{
   int c = (int)St.Pop() - (int)'0';
   num2 = c;
   St.Pop();
}
else num2 = Task10(St);



int min, max;

if (num1 > num2)
{
   max = num1;
   min = num2;
}
else
{
   max = num2;
   min = num1;
}

if (op == 'M')
{
   return max;
}
else
{
   return min;
}
```

```
    }

}

#endregion
```

Задание 11

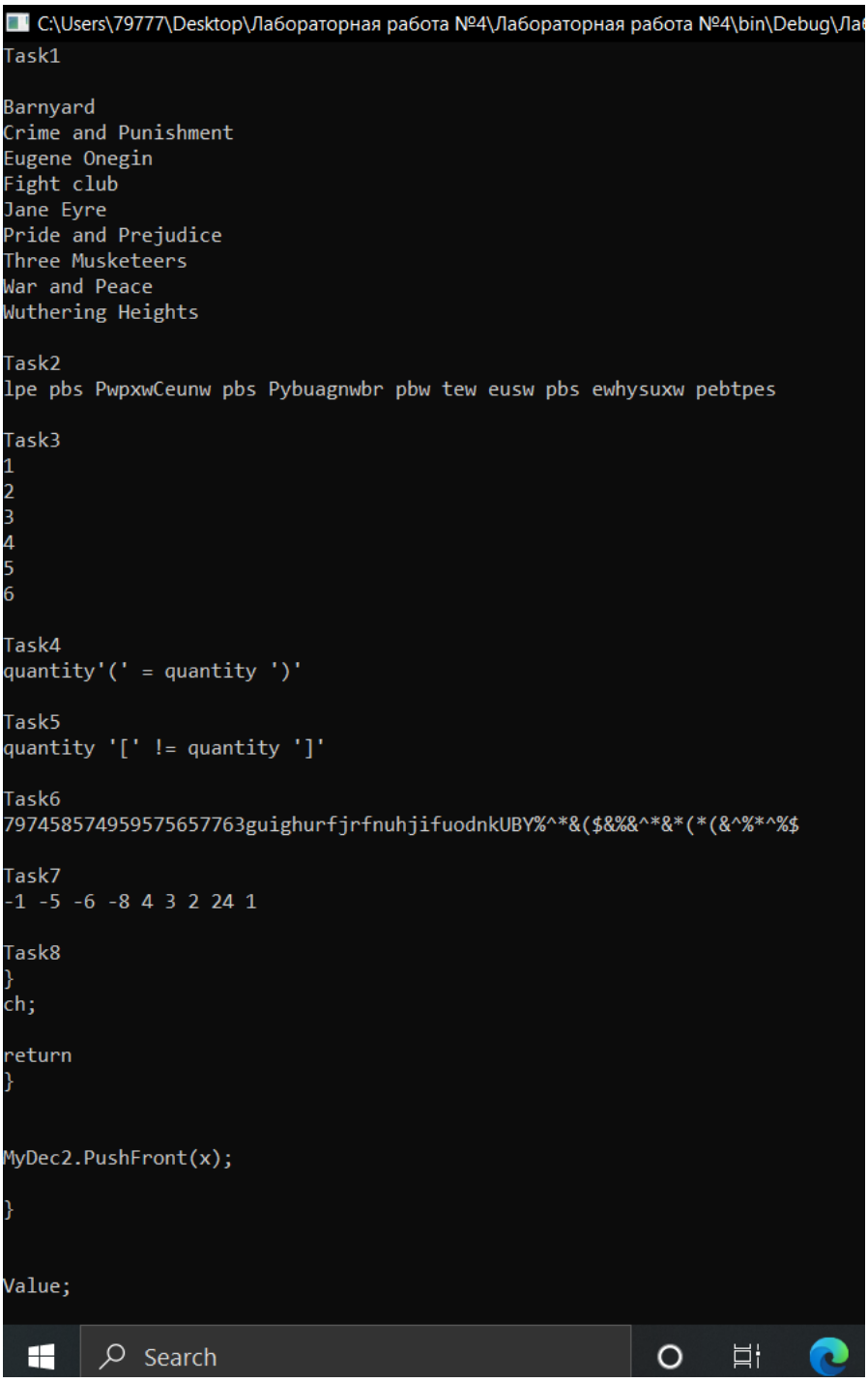
```
#region Task11
//проверяем правильность того, что находитмя внутри скобок и отдельно проверяем правильность того, что не в скобках
static void Task11(MyStack<char> St)
{

    String op = "+-";
    String chars = "xyz";

    while (St.Count != 0)
    {
        char ch = St.Pop();

        if (ch == '(' && St.Count>=4){

            char ch1 = St.Pop();
            char ch2 = St.Pop();
            char ch3 = St.Pop();
            char ch4 = St.Pop();
```

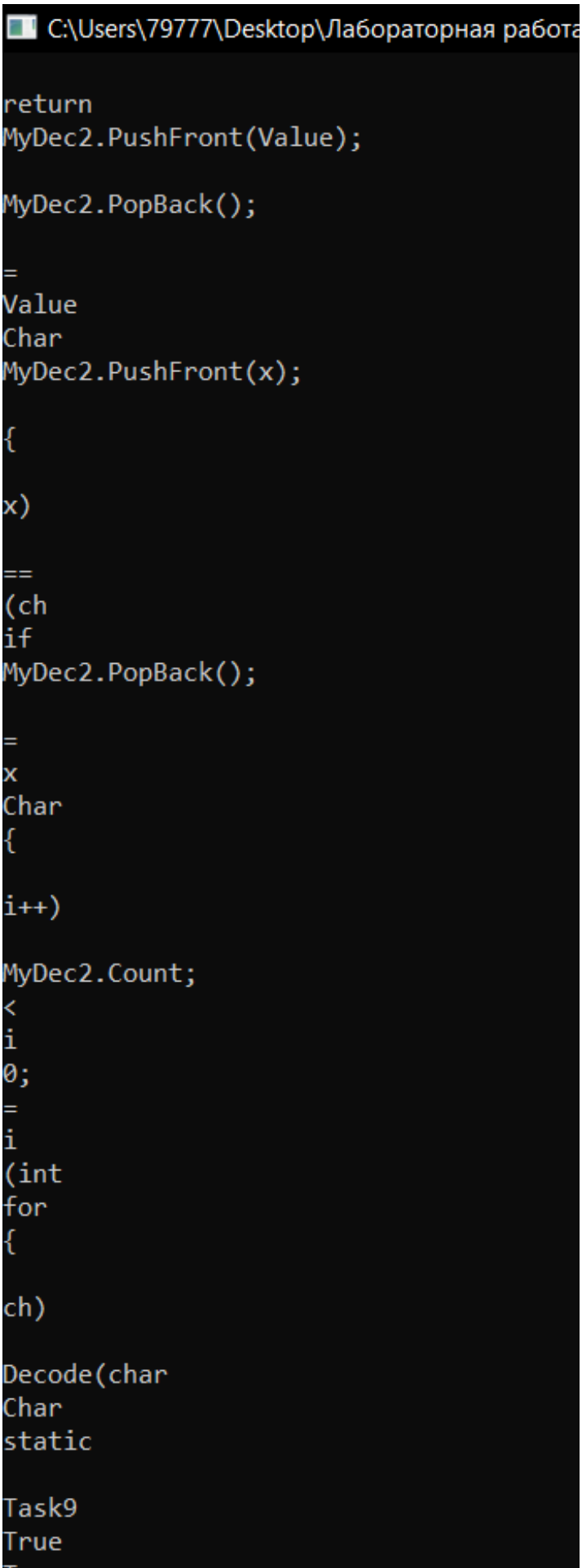
```
            if(!(chars.Contains(Convert.ToString(ch1)) && op.Contains(Convert.ToString(ch2)) &&
chars.Contains(Convert.ToString(ch3)) && ch4 == ')'))
            {
                Console.WriteLine(false);
                return;
            }
        }
        else if(ch == '(' && St.Count < 4)
        {
            Console.WriteLine(false);
            return;
        }

        else
        {
            String prov = "";
            prov += ch;

            while(St.Count != 0 )
            {
                char ch1 = St.Pop();
                prov += ch1;

                if (ch1 == '(' || St.Count==0)
                {
                    if (ch1 == '(')
                    {
                        prov = prov.Substring(0, prov.Length - 1);
```

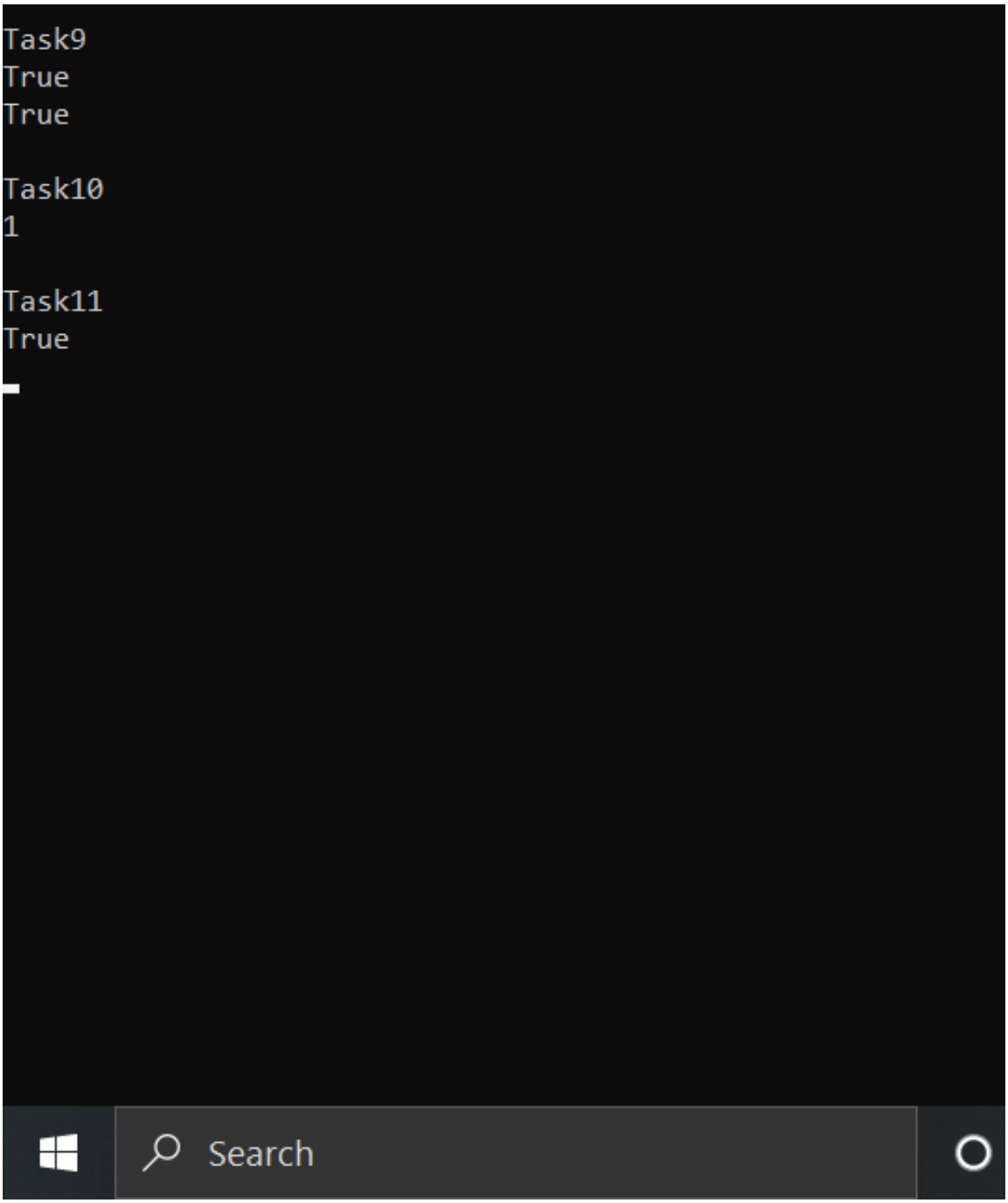
```
                St.Push(ch1);
            }

            if (prov.Length != 0)
            {
                if (!(prov.Length == 2 &&( (chars.Contains(Convert.ToString(prov[1])) &&
op.Contains(Convert.ToString(prov[0])))|| chars.Contains(Convert.ToString(prov[0])) &&
op.Contains(Convert.ToString(prov[1])))) && !(prov.Length == 1 &&
op.Contains(Convert.ToString(prov[0]))))
                {
                    Console.WriteLine( false);
                    return;
                }
            }

            break;
        }
        else
        {
            if (!chars.Contains(Convert.ToString(ch1)) && !op.Contains(Convert.ToString(ch1)))
            {
                Console.WriteLine(false);
                return;
            }
        }


    }
  }
}
```

```
            Console.WriteLine(true);


        }
        #endregion
    }
}
```

Результат работы программы:


C:\Users\79777\Desktop\Лабораторная работа №4\Лабораторная работа №4\bin\Debug\Ла
Task1
Barnyard
Crime and Punishment
Eugene Onegin
Fight club
Jane Eyre
Pride and Prejudice
Three Musketeers
War and Peace
Wuthering Heights
Task2
lpe pbs PwpxwCeunw pbs Pybuagnwbr pbw tew eusw pbs ewhysuxw pebtpes
Task3
1
2
3
4
5
6
Task4
quantity'(' = quantity ')'
Task5
quantity '[' != quantity ']'
Task6
7974585749595756577 63guighurfjrfnuhjifuodnkUBY%^*&($&%&^*&*(*(&^%*^%$
Task7
-1 -5 -6 -8 4 3 2 24 1
Task8
}
ch;
return
}
MyDec2.PushFront(x);
}
Value;
Search


C:\Users\79777\Desktop\Лабораторная работа
return
MyDec2.PushFront(Value);
MyDec2.PopBack();
=
Value
Char
MyDec2.PushFront(x);
{
x)
==
(ch
if
MyDec2.PopBack();
=
x
Char
{
i++)
MyDec2.Count;
<
i
0;
=
i
(int
for
{
ch)
Decode(char
Char
static
Task9
True



Вывод: реализовала стек и дек, выполнила все задания.