Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра автоматизированных систем управления (АСУ)

“Стеки и очереди”

Отчет по лабораторной работе №1

По дисциплине

«Структуры и алгоритмы обработки данных в ЭВМ»

Студент гр. 431-3

_____ Д.П. Андреев

«___»_____ 2022 г.

Проверил: профессор кафедры АСУ, д.т.н.

_____ А.Н. Горитов

«___»_____ 2022 г.

Томск 2022

1.Задание на лабораторную работу

Вариант 1,Задание 4:Пусть даны две очереди X и Y, содержащие вещественные числа. Из каждой очереди одновременно извлекается по одному числу x и y, соответственно. Если x<y, то число (x+y) помещается в конец очереди X, иначе число(x-y) помещается в конец очереди Y.Вычисления заканчиваются, когда одна из очередей становится пустой. Подсчитайте число шагов, через которое одна из очередей станет пустой. Для реализации АТД Очередь использовать массив. Начальное заполнение очередей X и Y считываются из файла.

2.Алгоритм решения задачи

Сначала мы заполняем очереди X и Y из файлов “X“ и “Y” соответственно. После чего создаём цикл while с условием (пока одна из очередей не будет пуста). Далее создаём два условия. При выполнении первого условия(x<y) выполняется операция сложения элементов и добавления этой суммы в очередь X. При выполнении второго условия(x>y) выполняется операция вычитания элементов и добавления этой разности в очередь Y. Так же в цикле есть счётчик, который считает количество операций. Когда одна из очередей будет пуста, цикл прекратит свою работу, после чего выводится на экран количество операций, которое понадобилось для обнуления этой очереди.

3.Листинг программы

```
#include <stdio.h>
#include <stdlib.h>
#include <windows.h>
#include <locale.h>

#define size 100
struct queue
{
   float qu[size];
   int rear, frnt;
};

void init(struct queue *q)
{
   q->frnt = -1;
   q->rear = -1;
   return;
}

int isempty(struct queue *q)
{
   if(q->frnt==-1)
   {
      return 1;
   }
   else
   {
      return 0;
   }
}

void add(struct queue *q, float x)
{
   if (q->frnt == -1||q->frnt == 0)
   {
      q->frnt = 0;
      q->rear = (q->rear + 1) % size;
      q->qu[q->rear]=x;
   }

}

void delit(struct queue *q)
{
   if (isempty(q))
   {
      printf("\n Очередь пуста \n");
      return (-1);
   }
   else
   {
      if (q->frnt == q->rear)
      {
         q->frnt = -1;
         q->rear = -1;
      }
      else
      {
         q->frnt = (q->frnt + 1) % size;
      }
   }
}

void print(struct queue *q)
{
   int h;
   if(isempty(q)==1)
   {
      printf("Очередь пуста!\n");
      return;
   }
   for(h = q->frnt; h<= q->rear; h++)
   {
      printf("%g ",q->qu[h]);
   }
 return;
}

int main()
{
   setlocale(LC_ALL, "Rus");
   float n;
   int countX,countY;
   float ch, pre = EOF;
   struct queue *x;
   struct queue *y;
   x = (struct queue*)malloc(size*sizeof(struct
queue));
   y = (struct queue*)malloc(size*sizeof(struct
queue));
   init(x);
   init(y);
   FILE *fx,*fy,*fx1,*fy1;
   fx = fopen("X.txt", "r");
   fy = fopen("Y.txt", "r");
   if ((fx = fopen("X.txt", "r")) == NULL)
   {
      printf("Не удалось открыть файл X");
      getchar();
      return 0;
   }
   if ((fy = fopen("Y.txt", "r")) == NULL)
   {
      printf("Не удалось открыть файл X/");
      getchar();
      return 0;
   }

   while(fscanf(fx, "%*[^\n]%*c") != EOF)
   {
      countX++;
   }
   while(fscanf(fy, "%*[^\n]%*c") != EOF)
   {
      countY++;
   }
   countY=countY-16;
   fclose(fx);
   fclose(fy);
   fx1 = fopen("X.txt", "r");
```

```
fy1 = fopen("Y.txt", "r");
for(int i=0;i<countX;i++)//X
{
   fscanf(fx1, "%g", &n);
   add(x, n);
}
for(int i=0;i<countY;i++)//Y
{
   fscanf(fy1, "%g", &n);
   add(y, n);
}
int cout=0;
float out;
while(isempty(x)!=1&&isempty(y)!=1)
{
   if(x->qu[x->frnt]<y->qu[y->frnt])
   {
      out=x->qu[x->frnt]+y->qu[y->frnt];
      add(x,out);
      delit(x);
      delit(y);
      printf("Очередь X:");
      print(x);
      printf(" \n");
      printf("Очередь Y:");
      print(y);
      printf(" \n");
   }
   if(x->qu[x->frnt]>y->qu[y->frnt])
   {
      out=x->qu[x->frnt]-y->qu[y->frnt];
      add(y,out);
      delit(x);
      delit(y);
      printf("Очередь X:");
      print(x);
      printf(" \n");
      printf("Очередь Y:");
      print(y);
      printf(" \n");
   }
   cout++;
}
printf("Кол.итерации= %d",cout);
fclose(fx);
fclose(fy);
getchar();
return 0;

}
```

4.Пример решения

Входные данные можно увидеть на рисунке 4.1.

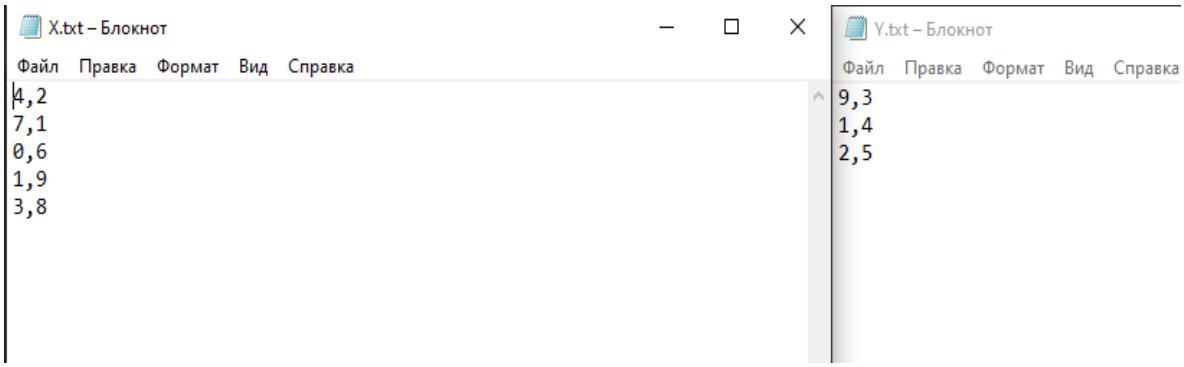

Рисунок 4.1— Входные данные

Результат работы программы можно увидеть на рисунке 4.2

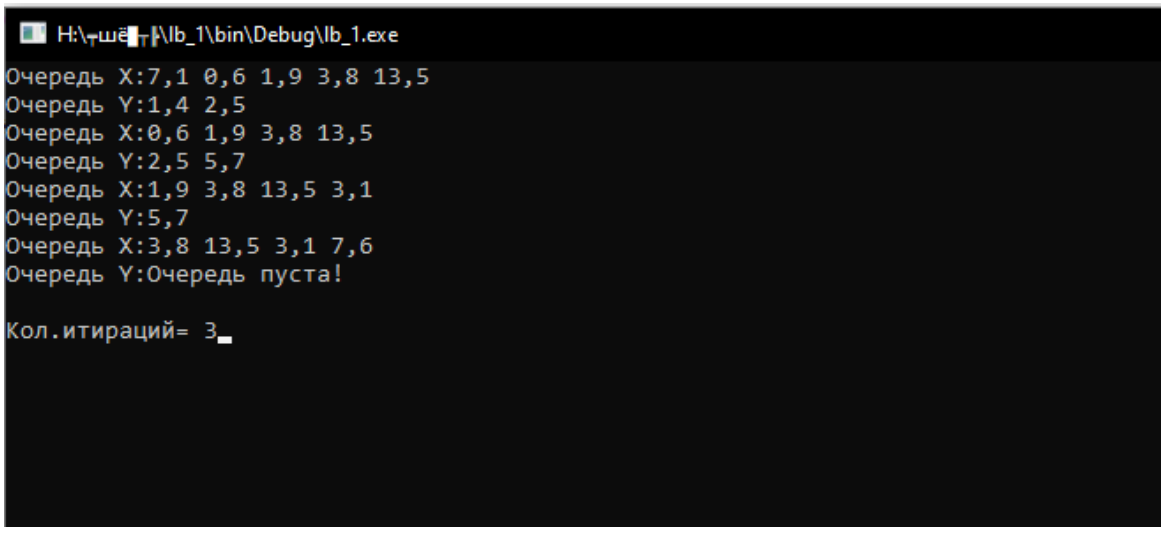

Рисунок 4.2 —Результат работы программы

5.Вывод

В результате лабораторной работы был изучен такой вид АТД как очередь и работа с ним.