Міністерство освіти і науки України
Національний університет “Львівська політехніка”
Інститут комп’ютерних наук та інформаційних технологій
Кафедра програмного забезпечення

Звіт

Про виконання лабораторної роботи №7

на тему:

«ВКАЗІВНИКИ НА ФУНКЦІЇ. РЕКУРСИВНІ ФУНКЦІЇ.»

з дисципліни «Основи програмування»

Лектор:
ст. викл. каф. ПЗ
Муха Т.О.

Виконав:
ст. гр. ПЗ-11
Морозов О.Р..

Прийняв:
асист. каф. ПЗ
Дивак І.В.

« __ » _____ 2021 р.

∑ = _____ _____ .

Львів – 2021

Тема: вказівники на функцію, рекурсивні функції.
Мета: поглиблене вивчення можливостей функцій в мові С з використанням механізмів рекурсії та вказівників.

ЗАВДАННЯ

Варіант 17

Завдання 1: Використовуючи вищенаведені функції swap та qs_sort, які реалізують алгоритм швидкого сортування масиву, написати програму мовою С для порівняння ефективності алгоритмів сортування масивів великих обсягів (наприклад, 100000 елементів). Програма повинна також реалізувати один з класичних алгоритмів сортування масиву згідно з варіантом індивідуального завдання. У програмі використати два одинакових масиви, які заповнити випадковими числами, здійснити перевірку впорядкованості елементів масиву, перевірку ідентичності масивів до і після сортування, а також за допомогою стандартної функції time, оцінити час виконання реалізованих алгоритмів сортування.
Сортування в порядку спадання “бульбашковим” методом з додатковою перевіркою чи масив вже відсортований.

Завдання 2: Написати мовою С три функції, щоб протабулювати, задану згідно варіанту, функцію на проміжку [a , b] з кроком h, використавши:
а) для першої функції оператор циклу for;
б) для другої – оператор циклу while;
в) для третьої – – оператор циклу do…while.

17. $f = x e^{-x}, a = 0, b = \ln 2;$

Завдання 3:

Вивести у зворотньому порядку заданий рядок символів, використовуючи рекурсію.

ТЕКСТ ПРОГРАМИ

Завдання 1:

Файл lab7_1.c:

```
#include "Includer.h"

char isSortedAsc(int[], int);
char areEqual(int[], int[], int);

int main(void) {
     double tm1, tm2;
     int mas1[MAS], mas2[MAS], i;
     srand(time(NULL));
     for (i = 0; i < MAS; i++)
     {
           mas1[i] = rand();
```

```
            mas1[i] *= rand();
            mas2[i] = mas1[i];
      }

      printf("Arrays are %sequal before sorting.\n", (areEqual(mas1, mas2, MAS) ? "" :
"not"));

      tm1 = clock() / (float)CLOCKS_PER_SEC;
      qs_sort(mas1, 0, MAS - 1);
      tm2 = clock() / (float)CLOCKS_PER_SEC;

      printf("First array is %s. Quick-sorting took %.3f seconds.\n", (isSortedAsc(mas1,
MAS) ? "sorted" : "not sorted"), (tm2 - tm1));

      tm1 = clock() / (float)CLOCKS_PER_SEC;
      bubblesort(mas2);
      tm2 = clock() / (float)CLOCKS_PER_SEC;

      printf("Second array is %s. Bubble-sorting took %.3f seconds.\n",
(isSortedAsc(mas2, MAS) ? "sorted" : "not sorted"), (tm2 - tm1));
      printf("Arrays are %sequal after sorting.\n", (areEqual(mas1, mas2, MAS) ? "" :
"not "));

      /*for (int i = 0; i < MAS; i++) {
            printf("%d\n", mas2[i]);
      }*/

      return 0;
}

char isSortedAsc(int mas[], int sz)
{
      for (int i = 1; i < sz; i++)
            if (mas[i - 1] < mas[i])
                  return 0;

      return 1;
}

char areEqual(int mas1[], int mas2[], int sz)
{
      for (int i = 0; i < sz; i++)
            if (mas1[i] - mas2[i])
                  return 0;

      return 1;
}
```

Файл bubblesort.c:

```
#include "Includer.h"

void bubblesort(int* array)
{
    for (int c = 0; c < MAS - 1; c++)
    {
        int swapped = 0;
        for (int d = 0; d < MAS - c - 1; d++)
        {
            if (array[d] < array[d + 1])
```

```
            {
                int swap = array[d];
                array[d] = array[d + 1];
                array[d + 1] = swap;
                swapped = 1;
            }
        }
        if (swapped == 0) {
            break;
        }
    }
}
```

Файл qsort.c:

```
#include "Includer.h"

void swap(int array[], long pos1, long pos2)
{
     long tmp;
     tmp = array[pos1];
     array[pos1] = array[pos2];
     array[pos2] = tmp;
}

void qs_sort(int* array, long start, long end)
{
    long head = start, tail = end - 1;
    long diff = end - start;
    long pe_index;
    // якщо залишилося менше двох елементів — кінець рекурсії
    if (diff < 1) return;
    if (diff == 1)
        if (array[start] < array[end])
        {
            swap(array, start, end);
            return;
        }
    // пошук індексу розділяючого елементу pe_index
    long m = (start + end) / 2;
    if (array[start] >= array[m])
    {
        if (array[m] >= array[end])
            pe_index = m;
        else
            if (array[end] >= array[start])
                pe_index = start;
            else
                pe_index = end;
    }
    else
    {
        if (array[start] >= array[end])
            pe_index = start;
        else
            if (array[end] >= array[m])
                pe_index = m;
            else
                pe_index = end;
    }
```

```
    long pe = array[pe_index]; // сам розділяючий елемент
    swap(array, pe_index, end);
    while (1)
    {
        while (array[head] > pe)
            ++head;
        while (array[tail] < pe && tail > start)
            --tail;
        if (head >= tail) break;
        swap(array, head++, tail--);
    }
    swap(array, head, end);
    long mid = head;
    qs_sort(array, start, mid - 1); // рекурсивний виклик для 1-ої підмножини
    qs_sort(array, mid + 1, end); // рекурсивний виклик для 2-ої підмножини
}
```

Файл Includer.h:

```
#pragma once
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define MAS 100000

#ifndef QSORT_H
#define QSORT_H

void swap(int array[], long pos1, long pos2);

void qs_sort(int array[], long start, long end);

#endif; // !QSORT_H

#ifndef BUBBLESORT_H
#define BUBBLESORT_H

void bubblesort(int* array);

#endif; // !BUBBLESORT_H
```

Завдання 2:

```
#include <stdio.h>
#include <math.h>

double data(double);
void tabulationWithFor(double, double, double);
void tabulationWithWhile(double, double, double);
void tabulationWithDoWhile(double, double, double);
void (*getResult) (double, double, double);

int main(void) {
    double a = 0, b = log(2), step;
    int option;
    printf("Enter step: ");
    scanf_s("%lf", &step);
```

```
    printf("Choose tabulation type (1 - for, 2 - while, 3 - do...while): ");
    scanf_s("%d", &option);

        switch (option)
        {
        case 1:
            getResult = &tabulationWithFor;
            break;
        case 2:
            getResult = &tabulationWithWhile;
            break;
        case 3:
            getResult = &tabulationWithDoWhile;
            break;
        default:
            break;
        }

        getResult(a, b, step);
}

double data(double x) {
    return x * exp(-x);
}

void tabulationWithFor(double a, double b, double step) {
    printf("Tabulation type: for\n");
    printf("|  x  |  f(x) |\n");
    for (double x = a; x < b; x+= step)
    {
        printf("|%0.3lf|%0.5lf|\n", x, data(x));
    }
}
void tabulationWithWhile(double a, double b, double step) {
    printf("Tabulation type: while\n");
    printf("|  x  |  f(x) |\n");
    double x = a;
    while (x <= b) {
        printf("|%0.3lf|%0.5lf|\n", x, data(x));
        x += step;
    }
}
void tabulationWithDoWhile(double a, double b, double step) {
    printf("Tabulation type: do.....while\n");
    printf("|  x  |  f(x) |\n");
    double x = a;
    do {
        printf("|%0.3lf|%0.5lf|\n", x, data(x));
        x += step;
    } while (x<b);
}
```

Завдання 3:

```
#include <stdio.h>

void printer(char*, int, int);

int main(void) {
     char s[21];

     printf("Enter 20 sumbols: ");
     fgets(s, sizeof(s), stdin);
     printer(s, strlen(s), strlen(s));
     return 0;
}

void printer(char* s, int n, int m)
{
     if (n > 0)
     {
          printer(s, n - 1, m);
          printf("%c", s[m - n]);
     }
     return;
}
```

РЕЗУЛЬТАТИ

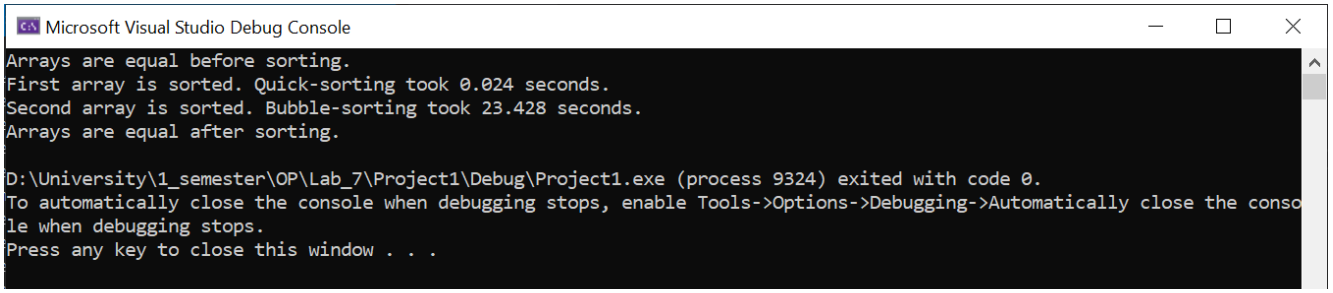

Рис 1. Результат виконання програми 1

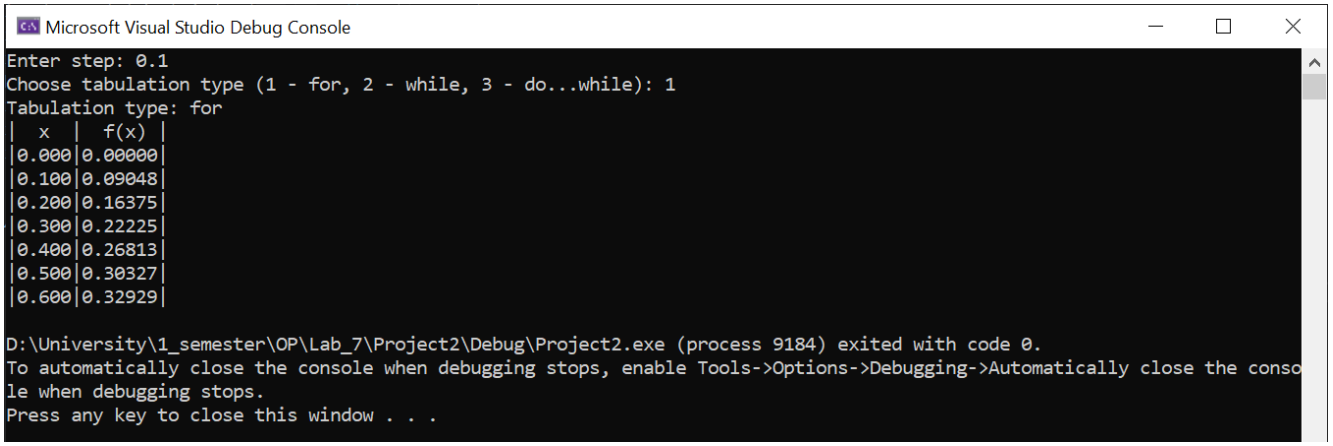

Рис 2. Результат виконання програми 2

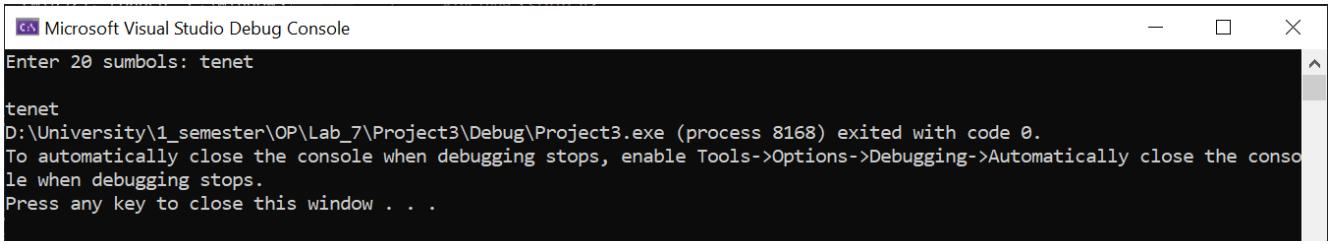

Рис 3. Результат виконання програми 3

ВИСНОВКИ

Виконуючи лабораторну роботу №7, я покращив свої знання про можливості функцій в мові С з використанням механізмів рекурсії та вказівників.