Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных средств

ОТЧЕТ

по лабораторной работе №7

«Объединения»

Выполнил
студ. гр.150702 (пг1)
Ефимчик А.И.

Проверил
ст. преп. каф. ЭВС
Демидович Г.Н.

Минск 2022

1 ЦЕЛЬ И ЗАДАЧИ ЛАБОРАТОРНОЙ РАБОТЫ

1.1 Целью лабораторной работы является изучение объединений и формирование практических навыков разработки алгоритмов и компьютерных программ объединений.

1.2 Для достижения поставленной цели необходимо решить следующие задачи:

1) Дополнить и расширить сведения по теме ЛР из учебного пособия [1].

1.3 Выполнить следующие задания по ЛР в соответствии с вариантом №5 , разработав алгоритмы их реализации, запрограммировав их с использованием языка «Си», отладив и представив результаты работы компьютерных программ.

Задание 1 Структура содержит информацию о геометрических фигурах: площадь (число), название (указатель), вложенное объединение – периметр (вещественное число) и цвет (строка фиксированной длины). Найти фигуры с максимальным периметром. Удалить фигуры заданного цвета.

2 РЕЗУЛЬТАТЫ ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

2.1 Результат выполнения задания по фильтрации и удалению фигуры по заданному параметру

2.1.1 На рисунке 1 приведена блок-схема алгоритма для выполнения задания №1 (заполнение массива фигур исходными данными, фильрация данных по периметру, удаление из массива фигур, фигуры заданного цвета).

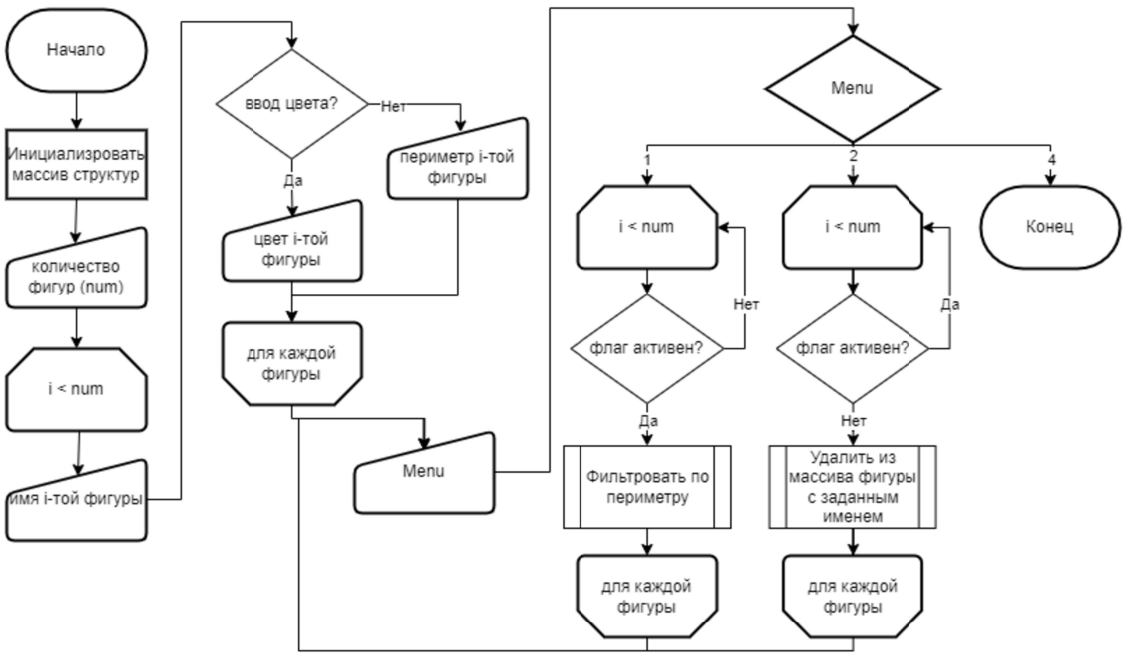


Рисунок 1 - Блок-схема алгоритма заполнения массива фигур исходными данными, фильрации данных по периметру, удаление из массива фигур фигуры заданного цвета

2.1.2 Листинг компьютерной программы по заданию 1 (заполнить массив структур, отфильтровать по параметру "периметр", удалить элемент массива по параметру "цвет") .

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "Strings.h"
#include "Struct.h"

struct figure
{
   int Square;
   char *Name;
   union Params
   {
      float Perimeter;
      char Color[7];
      // #pragma pack(push, 1)
      struct {
        // not working solution ?
         // unsigned : 10;
         // unsigned : 10;
         // unsigned : 10;
         // unsigned : 10;
         unsigned : 32;
         unsigned : 8;
         char flag: 8;
      };
   } params;
};

int main()
{
   printf("LAB 2.2 by Efimchik Alexandr from GROUP 150702 \n");
   printf("Enter figures num: ");
   int num;
   scanf("%d", &num);
   struct figure *figures;
   figures = calloc(num, sizeof(struct figure));
   for (int i = 0; i < num; i++)
   {
```

```
    printf("Enter %d figure name: ", i + 1);
    figures[i].Name = inputString();
    printf("What would you like to enter: \n1)Perimeter\n2)Color\n?\n");

    int d;
    scanf("%d", &d);
    switch (d)
    {
    case 1:
        printf("Enter %d figure perimeter: ", i + 1);
        scanf("%f", &figures[i].params.Perimeter);
        break;
    case 2:
        printf("Enter %d figure color: ", i + 1);
        char buffer[7];
        while (1)
        {
            fflush(stdin);
            scanf("%s", buffer);
            if (strlen(buffer) != 6)
            {
                printf("Invald color (need to be hex)\n");
                continue;
            } else {
                strcpy(figures[i].params.Color, buffer);
                figures[i].params.Color[6] = '\0';
                break;
            }
        }
        printf("Entered: %s\n", figures[i].params.Color);
        break;
    default:
        printf("Invalid option\n");
        break;
    }
}
int menu;

while (1)
{
    printf("\nSuperStructSoftware\n");
    printf("1) Filter figures by perimeter\n");
    printf("2) Delete figures by color\n");
    printf("3) View structs\n");
```

```
        printf("4) Exit\n");
        scanf("%d", &menu);
        switch (menu)
        {
        case 1:
            printf("Enter filter perimeter: ");
            float filter_perimeter;
            scanf("%f", &filter_perimeter);
            printf("Figures, with perimeter smaller than %f: \n", filter_perimeter);
            printf(" ================================\n");
            for (int i = 0; i < num; i++)
            {
                if (figures[i].params.flag)
                    continue;
                if (figures[i].params.Perimeter < filter_perimeter)
                {
                    printf("| Name: %6s | Perimeter: %5.5f |\n", figures[i].Name,
figures[i].params.Perimeter);
                }
            }
            printf(" ================================\n");
            break;
        case 2:
            printf("Specify delete filter: ");
            char buffer[7];
            while (1)
            {
                fflush(stdin);
                scanf("%s", buffer);
                if (strlen(buffer) != 6)
                {
                    printf("Invald color (need to be hex)\n");
                } else {
                    buffer[6] = '\0';
                    break;
                }
            }
            for (int i = 0; i < num; i++)
            {
                if (!figures[i].params.flag)
                    continue;
                if (!strcmp(buffer, figures[i].params.Color))
                {
                    for (int j = i + 1; j < num; j++)
```

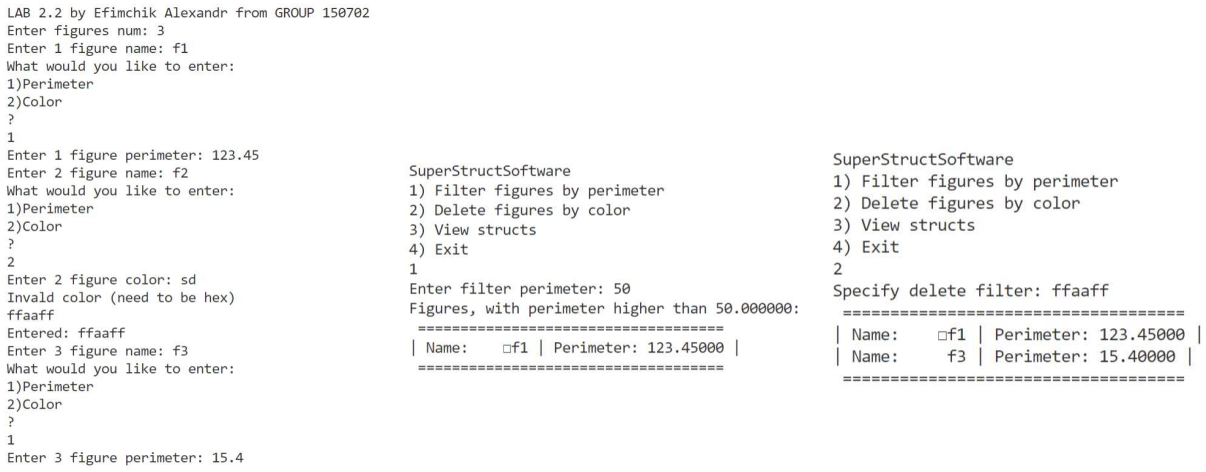
```
                {
                    figures[j - 1] = figures[j];
                }
                num--;
                figures = realloc(figures, sizeof(struct figure) * num);
            }
        }
        viewData(figures, num);
        break;
    case 3:
        viewData(figures, num);
        break;
    case 4:
        return 0;
    default:
        printf("Invalid menu index\n");
        break;
    }
  }
  return 0;
}

char *inputString()
{
  char *current_word;
  current_word = calloc(1, sizeof(char));
  int word_size = 0;
  char current_char = 0;
  current_char = getc(stdin);
  if (current_char != '\n') {
    current_word = realloc(current_word, sizeof(char) * (word_size + 1));
    current_word[word_size] = current_char;
    word_size++;
  }
  while (1)
  {
    current_char = getc(stdin);
    if (current_char == '\n')
    {
      break;
    }
    current_word = realloc(current_word, sizeof(char) * (word_size + 1));
    current_word[word_size] = current_char;
    word_size++;
```

```
    };
    current_word = realloc(current_word, sizeof(char) * (word_size + 1));
    current_word[word_size] = '\0';

    return current_word;
}
void viewData(struct figure* f, int num)
{
    printf(" ======================================== \n");
    for (int i = 0; i < num; i++)
    {
        if (f[i].params.flag != 0)
        {
            printf("| Name: %6s | Color: %6s       |\n", f[i].Name, f[i].params.Color);
        } else {
            printf("| Name: %6s | Perimeter: %5.5f |\n", f[i].Name,
f[i].params.Perimeter);
        }
    }
    printf(" ======================================== \n");
}
```

2.2.3 Результаты выполнения компьютерной программы по заданию 1 в виде «скрин-шота» изображения на мониторе представлены на рисунке 2:



а) б) в)

Рисунок 2 - Скрин-шот выполнения операций: а) заполнения массива структур пользовательскими данными; б) фильтрация фигур с заполненным полем «периметр» по заданному значению; в) удаление элемента массива с фигурой заданного названия

2.2 Выводы по результатам выполнения ЛР

В результате выполнения ЛР изучены методы работы с объединениями (принцип работы, причины использования, плюсы и минусы этого подхода), получены практические навыки по написанию функций работы с объединениями на языке С.

3 КОНТРОЛЬНЫЕ ВОПРОСЫ И ПРЕДЛОЖЕНИЯ

3.1 Пояснить основные положения, термины и определения в материалах лекции (лекций) и литературе по теме ЛР.

3.2 Объяснить алгоритмы выполнения заданий, указанных в данном варианте ЛР.

3.3 Прокомментировать листинги (фрагменты листингов) компьютерных программ в данном варианте ЛР.

3.4 Прокомментировать результаты выполнения заданий, указанных в варианте ЛР.

4 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Луцик, Ю. А. Основы алгоритмизации и программирования : язык Си : учебно-метод. пособие / Ю. А. Луцик, А. М. Ковальчук, Е. А. Сасин. – Минск : БГУИР, 2015. – 170с. : ил.