Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра вычислительных систем

ОТЧЕТ
по практической работе 1

по дисциплине «**Программирование**»

Выполнил:
студент гр. ИВ-222
«9» февраля 2023 г. _____ Кочетков Р.Н.

Проверил:
Старший преподаватель кафедры
вычислительных систем
«__» февраля 2023 г. _____ Фульман В.О.

Оценка «_____»

Новосибирск 2023

ОГЛАВЛЕНИЕ

ЗАДАНИЕ

Получение навыков отладки программ на примере использования отладчика GDB.

Задание 1

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

void init(int* arr, int n)
{
    arr = malloc(n * sizeof(int));
    int i;
    for (i = 0; i < n; ++i)
    {
        arr[i] = i;
    }
}

int main()
{
    int* arr = NULL;
    int n = 10;
    init(arr, n);

    int i;
    for (i = 0; i < n; ++i)
    {
        printf("%d\n", arr[i]);
    }
    return 0;
}
```

Задание 2

```
#include <stdio.h>

typedef struct
{
    char str[3];
    int num;
} NumberRepr;

void format(NumberRepr* number)
{
    sprintf(number->str, "%3d", number->num);
}

int main()
{
    NumberRepr number = { .num = 1025 };

    format(&number);

    printf("str: %s\n", number.str);
    printf("num: %d\n", number.num);

    return 0;
}
```

Задание 3

```
#include <stdio.h>

#define SQR(x) x * x

int main()
{
    int y = 5;
    int z = SQR(y + 1);
    printf("z = %d\n", z);
    return 0;
}
```

Задание 4

```
#include <stdio.h>

void swap(int* a, int* b)
{
    int tmp = *a;
    *a = *b;
    *b = tmp;
}

void bubble_sort(int* array, int size)
{
    int i, j;
    for (i = 0; i < size - 1; ++i) {
        for (j = 0; j < size - i; ++j) {
            if (array[j] > array[j + 1]) {
                swap(&array[j], &array[j + 1]);
            }
        }
    }
}


int main()
{
    int array[100] = {10, 15, 5, 4, 21, 7};

    bubble_sort(array, 6);

    int i;
    for (i = 0; i < 6 ; ++i) {
        printf("%d ", array[i]);
    }
    printf("\n");

    return 0;
}
```

ВЫПОЛНЕНИЕ РАБОТЫ

1. В задании 1 на экран должен выводиться массив, заполненный значениями от 0 до 9.

```
roma@Roma:~/lab$ gcc -Wall -g -O0 -o 1 1.c
roma@Roma:~/lab$ ./1
Ошибка сегментирования
```

При компиляции ошибок нет, при запуске исполняемого файла происходит ошибка сегментирования.

Запускаем gdb с точкой останова на функции init

```
(gdb) b init
Breakpoint 1 at 0x117c: file 1.c, line 6.
```

Видим, что в init массив заполняется правильно

```
(gdb)
8               for (i = 0; i < n; ++i)
1: *arr@10 = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

После отработки init в функции main arr всё ещё инициализирован нулевым указателем

```
1: *arr@10 = <error: Cannot access memory at address 0x0>
(gdb) n
18              init(arr, n);
1: *arr@10 = <error: Cannot access memory at address 0x0>
(gdb)
21              for (i = 0; i < n; ++i)
1: *arr@10 = <error: Cannot access memory at address 0x0>
```

Область действия участка памяти, выделяемого malloc, не распространяется на main

Для исправления ошибки выделяем память в функции main и уже потом передаём массив в init

```
int main()
{
        int* arr = NULL;
        int n = 10;
        arr = malloc(n * sizeof(int));
        init(arr, n);
```

2. Во 2 задании программа должна инициализировать поля структуры одним числом в символьном и целочисленном типах.

```
roma@Roma:~/lab$ gcc -Wall -g -O0 -o 2 2.c
2.c: In function ‘format’:
2.c:9:30: warning: ‘sprintf’ writing a terminating nul past the end of the destination [-Wformat-overflow=]
    9 |     sprintf(number->str, "%3d", number->num);
      |                              ^
2.c:9:5: note: ‘sprintf’ output between 4 and 12 bytes into a destination of size 3
    9 |     sprintf(number->str, "%3d", number->num);
      |     ^~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
roma@Roma:~/lab$ ./2
str: 1025
num: 1024
```

Предупреждения компилятора указывают на функцию format, вывод полей неправильный
Сравним значения полей структуры до и после вызова функции format

```
Breakpoint 1, main () at 2.c:14
14          format(&number);
(gdb) display number.num
1: number.num = 1025
(gdb) display number.str
2: number.str = "\000\000"
(gdb) n
15          printf("str: %s\n", number.str);
1: number.num = 1024
2: number.str = "102"
```

Оба поля принимают неправильные значения
Посмотрим на расположение полей экземпляра структуры в памяти

```
Breakpoint 1, main () at 2.c:14
14          format(&number);
(gdb) x/3cb number.str
0x7fffffffde40: 0 '\000'        0 '\000'        0 '\000'
(gdb) p &number.num
$1 = (int *) 0x7fffffffde44
```

Они располагаются один за другим без пробелов, то есть при отработке format символы, не вместившиеся в str записываются в num.
Для исправления ошибки переставим поля структуры местами

```
typedef struct
{
    int num;
    char str[3];
} NumberRepr;
```

3. В 3 задании число, увеличенное на единицу, должно возводиться в квадрат с помощью функции, заданной макросом, и выводиться на экран.

```
roma@Roma:~/lab$ gcc -Wall -gdwarf-2 -g3 -O0 -o 3 3.c
roma@Roma:~/lab$ ./3
z = 11
```

Программа компилируется без ошибок, но не возводит число в квадрат (z должно быть равным 36)

Развернём макрос с помощью команды macro expand

```
Breakpoint 1, main () at 3.c:7
7               int z = SQR(y + 1);
(gdb) macro expand SQR(y + 1)
expands to: y + 1 * y + 1
```

Видим, что ошибка возникает из-за неправильного порядка действий

Для исправления ошибки поставим скобки при определении макроса

```
2   #define SQR(x) (x) * (x)
```

4. В 4 задании программа должна сортировать массив пузырьковым методом и выводить его на экран.
При компиляции ошибок нет, при запуске исполняемого файла на экран выводится неотсортированный массив

```
roma@Roma:~/lab$ gcc -Wall -g -O0 -o 4 4.c
roma@Roma:~/lab$ ./4
4 0 5 7 10 15
```

Поставим точку останова в функции bubble_sort и сверим значения соседних элементов до и после отработки функции swap

```
Breakpoint 1, bubble_sort (array=0x7fffffffdcb0, size=6) at 4.c:15
15                                      swap(&array[j], &array[j + 1]);
(gdb) p array[j]
$1 = 15
(gdb) p array[j+1]
$2 = 5
(gdb) n
13                      for (j = 0; j < size - i; ++j) {
(gdb) p array[j]
$3 = 5
(gdb) p array[j+1]
$4 = 15
```

swap работает правильно
В функции bubble_sort отследим значение переменной j, отвечающей за индексы массива

```
Breakpoint 1, bubble_sort (array=0x7fffffffdcb0, size=6) at 4.c:12
12              for (i = 0; i < size - 1; ++i) {
(gdb) display j
12              for (i = 0; i < size - 1; ++i) {
1: j = 6
```

Видим, что j принимает недопустимое значение 6, при том что максимальный индекс массива равен 5.
Для исправления ошибки меняем условие вложенного цикла for с size - i на size - 1

```
for (i = 0; i < size - 1; ++i) {
        for (j = 0; j < size - 1; ++j) {
                if (array[j] > array[j + 1]) {
                        swap(&array[j], &array[j + 1]);
                }
```

ПРИЛОЖЕНИЕ

Исправленная программа из задания 1:

11.c

```c
#include <stdio.h>
#include <stdlib.h>

void init(int* arr, int n)
{
        int i;
        for (i = 0; i < n; ++i)
        {
                arr[i] = i;
        }
}

int main()
{
        int* arr = NULL;
        int n = 10;
        arr = malloc(n * sizeof(int));
        init(arr, n);
        int i;
        for (i = 0; i < n; ++i)
        {
                printf("%d\n", arr[i]);
        }
        return 0;
}
```

Исправленная программа из задания 2:

22.c

```c
#include <stdio.h>

typedef struct
{
    int num;
    char str[3];
} NumberRepr;

void format(NumberRepr *number)
{
    sprintf(number->str, "%3d", number->num);
}

int main()
{
    NumberRepr number = {.num = 1025};
    format(&number);
    printf("str: %s\n", number.str);
    printf("num: %d\n", number.num);
    return 0;
}
```

Исправленная программа из задания 3:

33.c

```c
#include <stdio.h>
#define SQR(x) (x) * (x)

int main()
{
        int y = 5;
        int z = SQR(y + 1);
        printf("z = %d\n", z);
        return 0;
}
```

Исправленная программа из задания 4:

44.c

```c
#include <stdio.h>

void swap(int* a, int* b)
{
        int tmp = *a;
        *a = *b;
        *b = tmp;
}

void bubble_sort(int* array, int size)
{
        int i, j;
        for (i = 0; i < size - 1; ++i) {
                for (j = 0; j < size - 1; ++j) {
                        if (array[j] > array[j + 1]) {
                                swap(&array[j], &array[j + 1]);
                        }
                }
        }
}

int main()
{
        int array[100] = {10, 15, 5, 4, 21, 7};
        bubble_sort(array, 6);
        int i;
        for (i = 0; i < 6 ; ++i) {
                printf("%d ", array[i]);
        }
        printf("\n");
        return 0;
}
```