Национальный исследовательский ядерный университет «МИФИ»
Институт интеллектуальных кибернетических систем
Кафедра №12 «Компьютерные системы и технологии»

ОТЧЕТ

О выполнении лабораторной работы №3
«Работа с массивами данных»

Студент: Баранов А.Т.
Группа: Б22-534
Преподаватель: Широких Т.А.

Москва — 2022

1. Формулировка индивидуального задания

Вариант №87

Индивидуальное задание

В исходной последовательности найти группы подряд стоящих одинаковых чисел и создать из них новую последовательность. Удалить найденные группы из исходной последовательности, оставив в ней по одному элементу каждой группы.

Правила изменения размера выделенной под массив области памяти

Размер выделенной под массив области памяти должен изменяться автоматически при выполнении операций, приводящих к изменению количества элементов в массиве.

При заполнении элементами массива всей выделенной под него области памяти её размер должен автоматически увеличиваться на объём, необходимый для размещения N дополнительных элементов массива. При наличии в выделенной под массив области памяти места для $N + 1$ новых элементов, её размер должен сократиться на объём, необходимый для хранения N элементов.

2. Описание использованных типов данных

При выполнении данной лабораторной работы использовался встроенный тип данных `int`, предназначенный для работы с целыми числами, встроенный тип данных double, предназначенный для работы с числами с плавающей запятой двойной точности, и встроенный тип данных char, предназначенный для работы с символами.

3. Описание использованного алгоритма

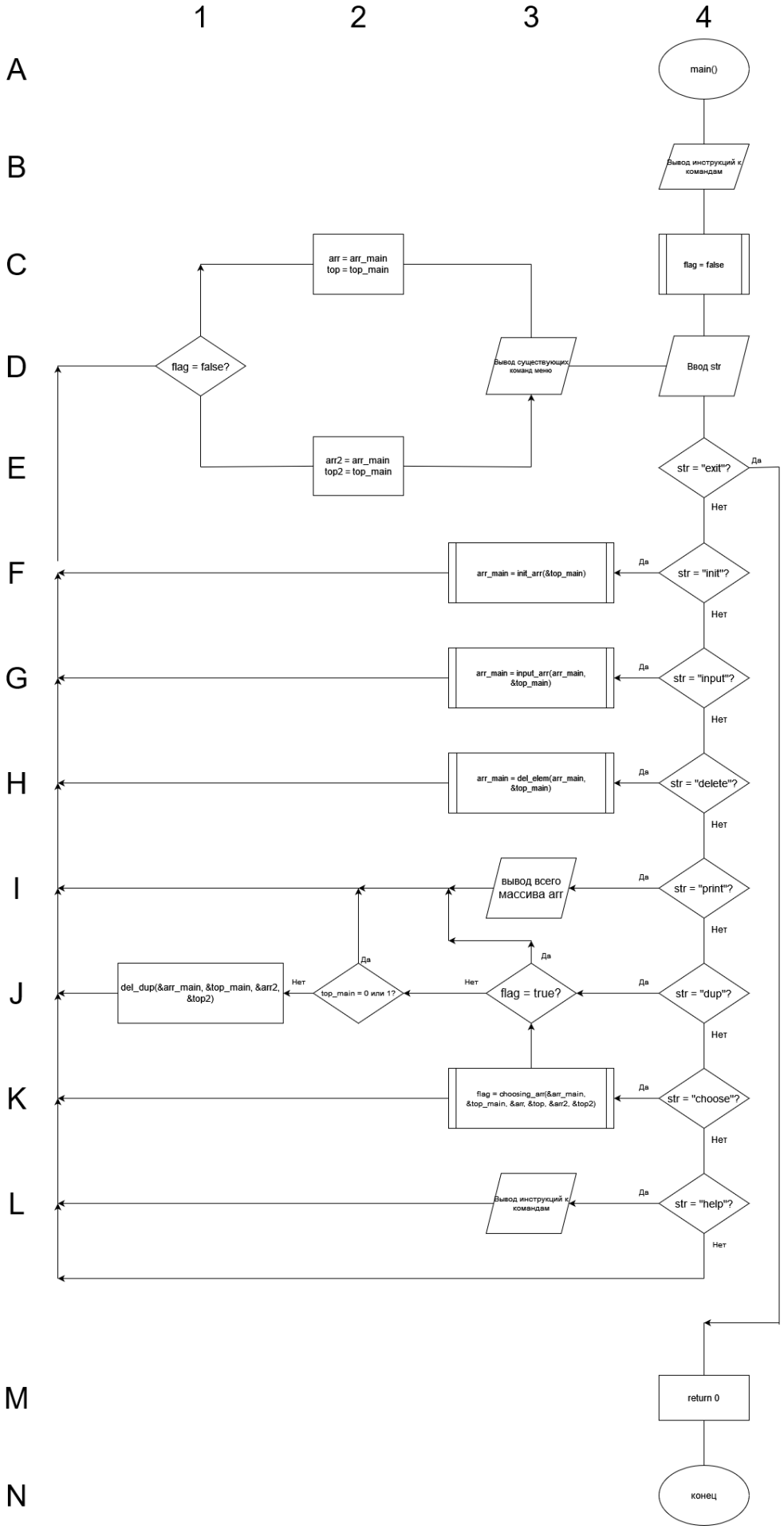


Рис. 1: Блок-схема алгоритма работы функции `main()`

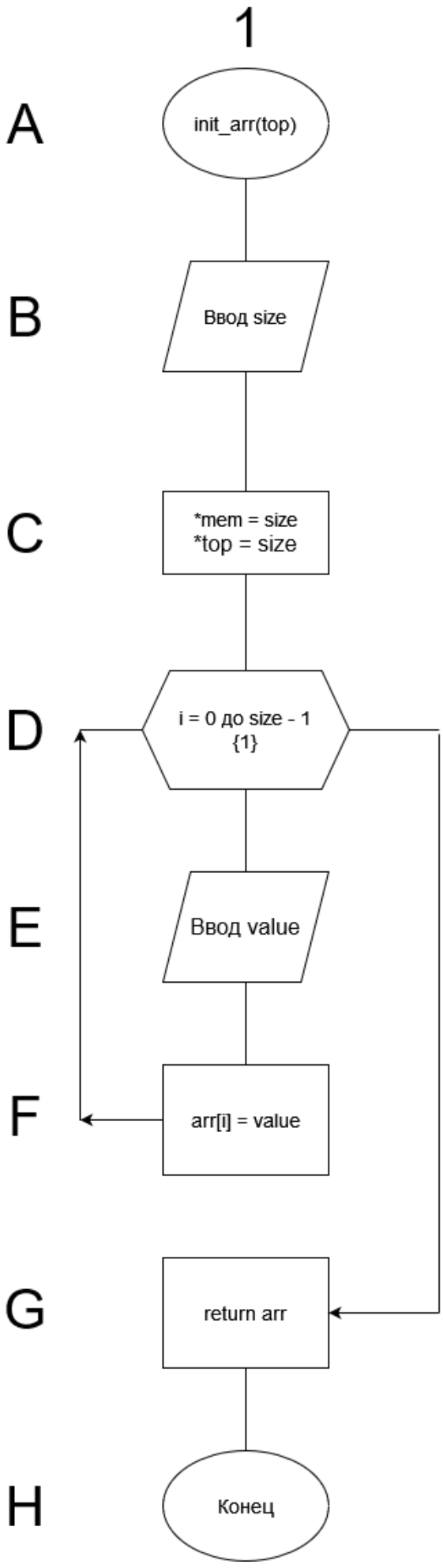


Рис. 2: Блок-схема алгоритма работы функции `init_arr()`

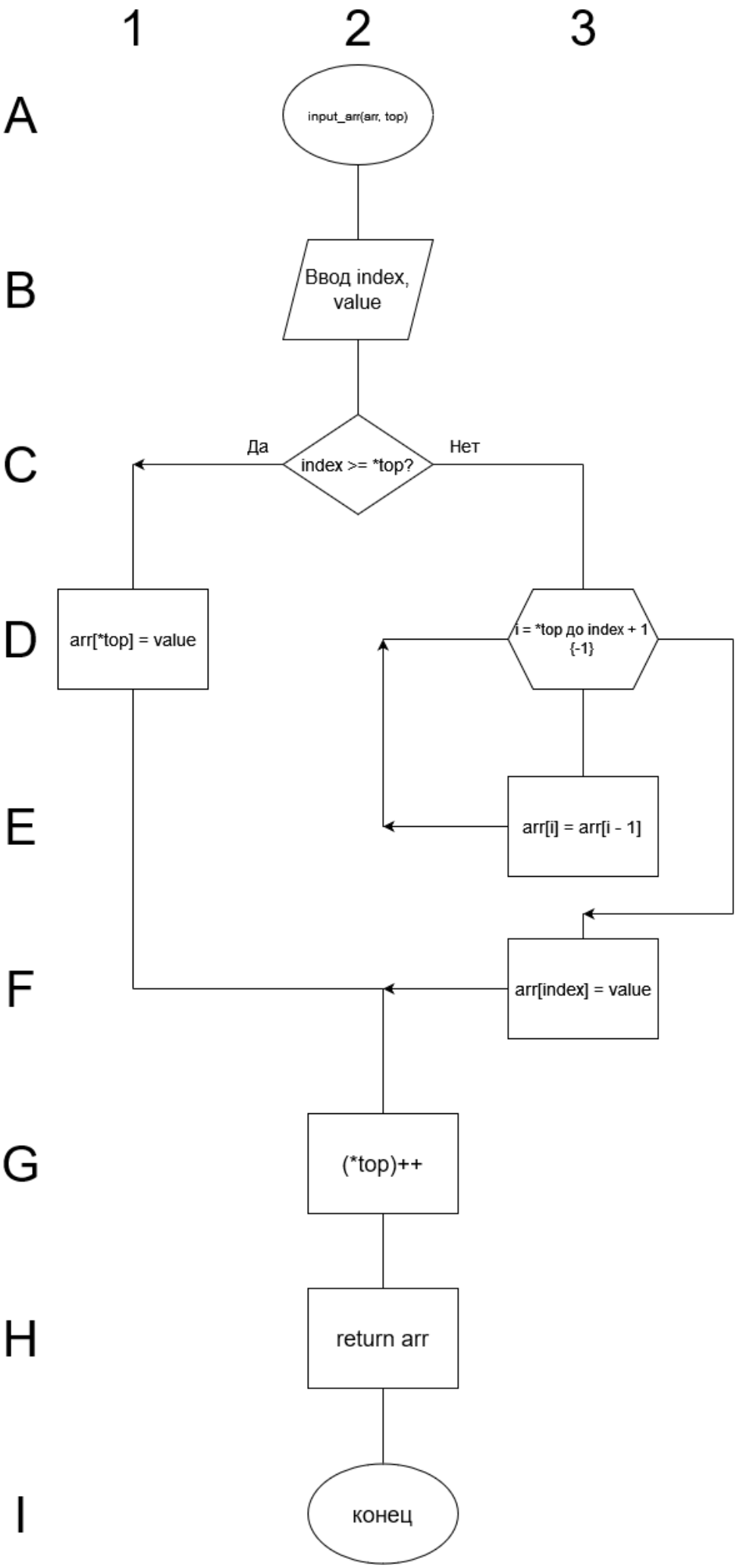


Рис. 3: Блок-схема алгоритма работы функции `input_arr()`

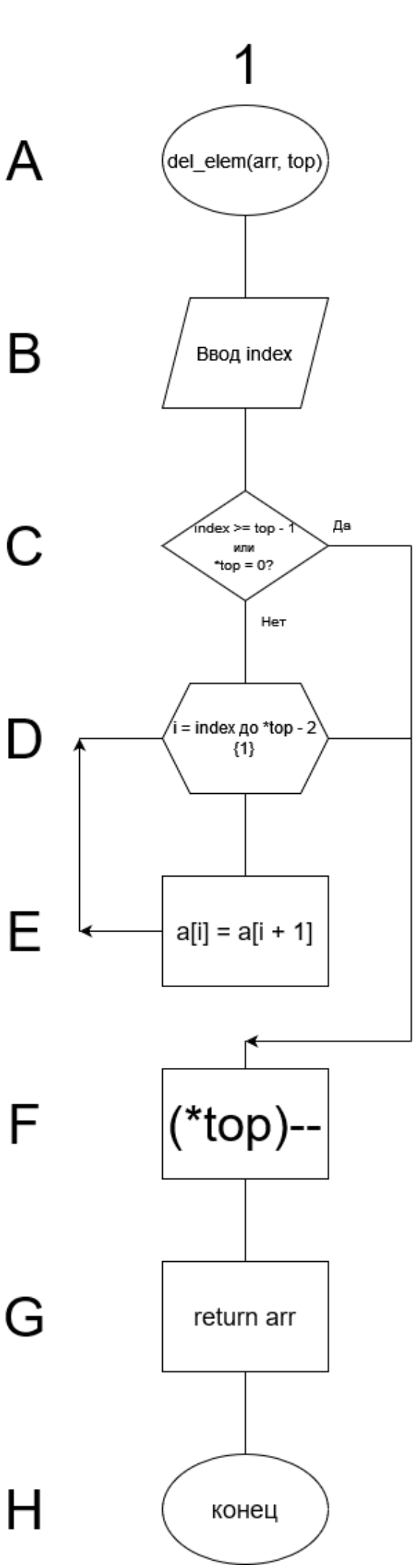


Рис. 4: Блок-схема алгоритма работы функции del_elem()

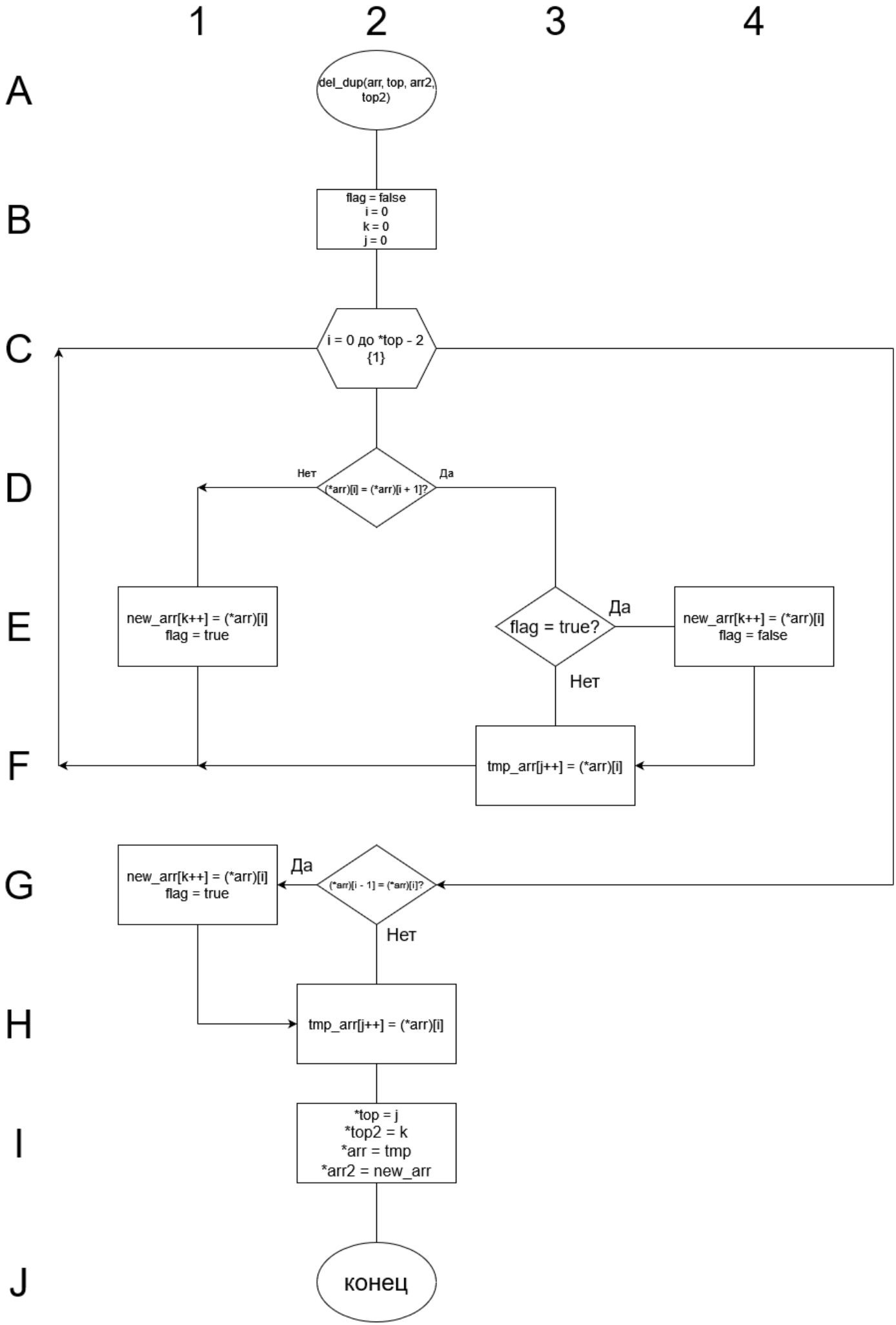


Рис. 5: Блок-схема алгоритма работы функции `del_dup()`

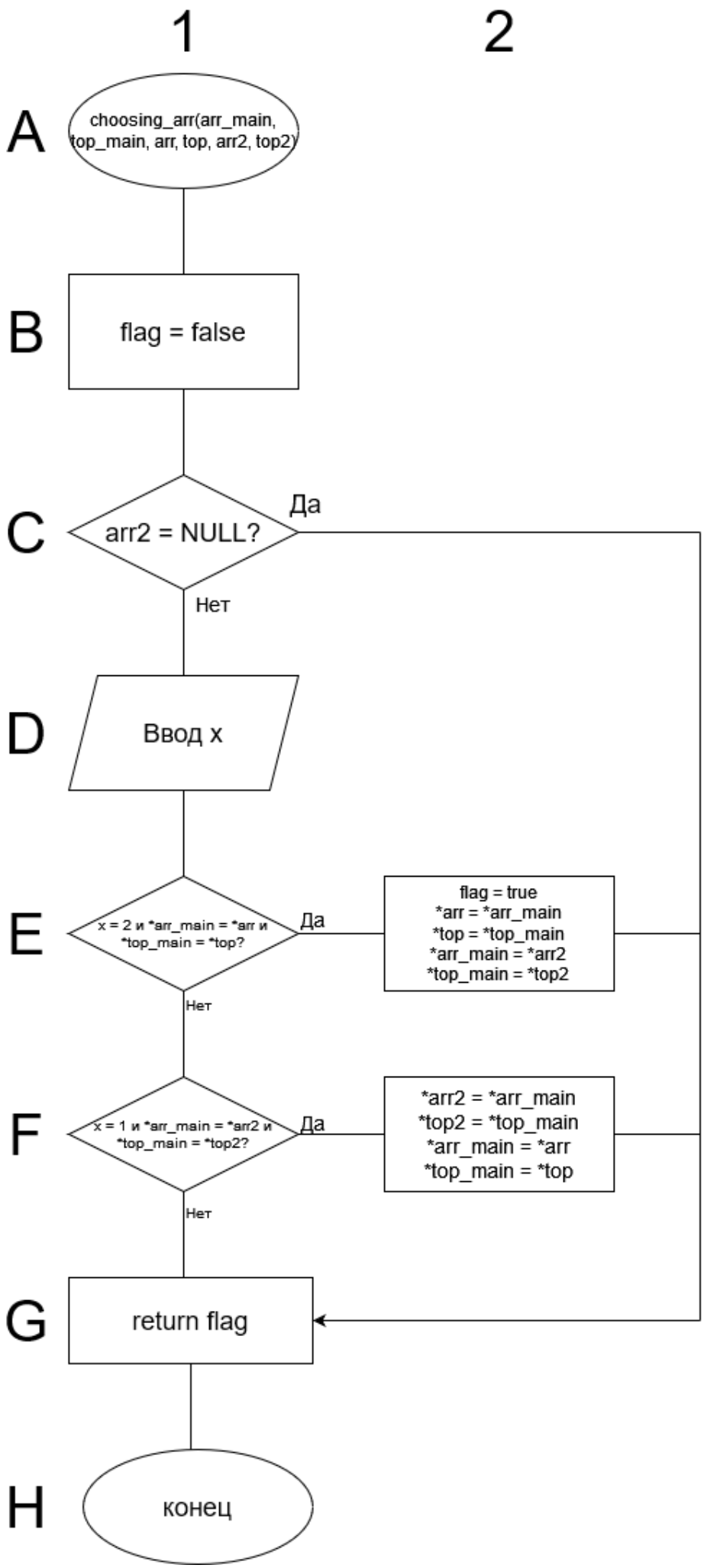


Рис. 6: Блок-схема алгоритма работы функции `choosing_arr()`

4. Исходные коды разработанных программ

Листинг 1: Исходные коды программы main.o (файл: m a i n . c)

```
#include "general.h"

void menu(bool);

int main()
{
    char *str = (char*) calloc(7, sizeof(char)); //создаём строку, куда будут вводиться команды
    unsigned int top = 0, top2 = 0; //длина массивов в любой момент времени. Сначала массивы
//пустые и значение 0.
    unsigned int mem = 0, mem2 = 0; //количество ячеек памяти, выделенных для массива в любой
//момент времени

    long double *arr = NULL, *arr2 = NULL; //указатели на главный и второй массив
    unsigned int mem_main = mem; //количество ячеек памяти,
    unsigned int top_main = top; //указатель и длина массива,
    long double *arr_main = arr; //который выбрал пользователь (функция choose). Всегда
//начинаем с главного массива.

    bool flag = false;   //флажок выбранного массива, для первого и второго немного разные
//правила использования

    menu(flag);          //начинаем диалог с пользователем

    while(1) {
        printf("$");
        scanf("%6[^\n]", str);  //ввод команды, она заведомо не может быть более 6 символов.
        if (str_check(str, flag) == true)//если заранее прошла проверка на неверность команды -
            continue;                    // - следующая итерация(еще раз просим команду)
        if (strcmp(str, "exit") == 0) //штатный способ завершения программы
        {
            printf("\nBye!");
            break;
        }
        //дальше идет сравнение введенной пользователем команды с одной из предусмотренных
//команд, если совпало - выполняем для выбранного массива(строка 12)
        else if (strcmp(str, "init") == 0)
        {
            free(arr_main);
            arr_main = init_arr(&top_main, &mem_main);
        }
        else if (strcmp(str, "input") == 0)
            arr_main = input_arr(arr_main, &top_main, &mem_main);
        else if (strcmp(str, "delete") == 0)
            arr_main = del_elem(arr_main, &top_main, &mem_main);
        else if (strcmp(str, "print") == 0)
            print_arr(arr_main, top_main);
        else if (strcmp(str, "dup") == 0)
            if (flag)       //мы не можем создавать 2ой массив из 2го, 2 массив создается
                printf("\nYou can`t use \"dup\" with array N2!\n"); //только из главного
            else if(top_main == 0 or top_main == 1)  //главный массив может быть слишком
                printf("Array is empty or it has too few elements.\n");//короткий для такой
//команды
            else
            {
                if (!flag)
                    free(arr2);
                del_dup(&arr_main, &top_main, &mem_main, &arr2, &top2, &mem2);
            }
        else if (strcmp(str, "choose") == 0)
            flag = choosing_arr(&arr_main, &top_main, &mem_main, &arr, &top, &mem, &arr2,
&top2, &mem2);
```

```
        else if(strcmp(str, "help") == 0)
            menu(flag);
        else                                //если ничего не подошло - значит неверная команда
            printf("\nYou`ve wrote wrong command. Try again:");
        if (!flag)
        {
            arr = arr_main;  //после каждой итерации запоминаем параметры выбранного массива
            top = top_main;
            mem = mem_main;
            printf("\nYou are using array N1");  //поддерживаемые команды для выбранного
//массива и строка, показывающая текущий массив
            printf("\nMENU: init   input   delete   print   dup   choose   help   exit\n");
        }
        else
//аналогично для второго массива
        {
            arr2 = arr_main;
            top2 = top_main;
            mem2 = mem_main;
            printf("\nYou are using array N2");
            printf("\nMENU: init   input   delete   print   choose   help   exit\n");
        }
    }
    free(arr);          //прибираемся за собой
    free(arr2);
    free(str);

    return 0;
}

void menu(bool flag)     //меню с поддерживаемыми для выбранного массива командами
{
    printf("\nWrite   \"init\"    if you want to initialize array.\n");
    printf("Write   \"input\"   if you want to input element into array by index.\n");
    printf("Write   \"delete\"  if you want to delete the element from array by index.\n");
    printf("Write   \"print\"   if you want to see your array.\n");
    if(!flag)
        printf("Write   \"dup\"     if you want to process equal elements, which are nearby in
array.\n");
    printf("Write   \"choose\"  if you want to choose array.\n");
    printf("Write   \"help\"    if you want to read this message again.\n");
    printf("Write   \"exit\"    if you want to exit the program.\n\n");
}
```

Листинг 2: Исходные коды программы main.o (файл: c h e c k i n g .h)

```
#ifndef LAB3_CHECKING_H
#define LAB3_CHECKING_H

#define N 5                 //Только >= 1 (целые числа)!

#include <stdbool.h>
#include <stdio.h>
#include "iso646.h"
#include <stdlib.h>
#include <string.h>

bool str_check(char*, bool);
bool choosing_arr(long double**, unsigned int*, unsigned int*, long double**,
unsigned int*, unsigned int*, long double**, unsigned int*, unsigned int*);
long double input_elem();
unsigned int input_int();
long double *allocating(long double*, const unsigned int*, unsigned int*);

#endif //LAB3_CHECKING_H
```

Листинг 3: Исходные коды программы main.o (файл: c h e c k i n g . c)

```
#include "checking.h"

bool str_check(char* str, bool arr)          //функция проверки ввода команды
{
    char check;                              //нужная переменная для проверки здорового ввода
    bool flag = false;                       //флажок, который мы возвратим

    check = getchar();                       //берем первый символ из stdin

    if (check == -1)                         //если произошли ошибка ввода или конец ввода
    {
        fprintf(stderr, "\nError or EOF...\nExit the program");
        free(str);
        exit(1);
    }

    else if(check != '\n' or strlen(str) < 3)        //если заведомо больше 6 символов или
меньше 3 - команда автоматически некорректная
    {
        printf("\nYou`ve wrote wrong command. Try again:");
        if (!arr)                       //выводим меню в соответствии с текущим выбором массива
        {
            printf("\nYou are using array N1");
            printf("\nMENU: init   input   delete   print   dup   choose   help   exit\n");
        }
        else
        {
            printf("\nYou are using array N2");
            printf("\nMENU: init   input   delete   print   choose   help   exit\n");
        }

        scanf("%*[^\n]%*c");

        flag = true;                    //команда введена некорректно - поднимаем флажок
    }

    return flag;
}

bool choosing_arr(long double **arr_main, unsigned int *top_main, unsigned int *mem_main,
long double **arr, unsigned int *top, unsigned int *mem, long double **arr2, unsigned int
*top2, unsigned int *mem2)
{
    bool flag = false;                      //флажок выбора массива

    if (*arr2 != NULL)                      //обязательно проверяем, существует ли второй массив
    {
        unsigned int x = 0;                 //стандартные проверки на здоровый ввод числа
        printf("\nChoose array(1 or 2):");
        char check1 = scanf("%1u", &x), check2 = getchar();

        while(check1 != 1 or check2 != '\n' or (x != 1 and x != 2))
        {
            if (check2 == -1)
            {
                fprintf(stderr, "\nError or EOF...\nExit the program");
                exit(1);
            }
            else
                printf("You haven`t wrote 1 or 2. Try again:");

            scanf("%*[^\n]%*c");
            check1 = scanf("%1u", &x);
            check2 = getchar();
        }
```

```
        if(x == 2 and *arr_main == *arr and *top_main == *top and *mem_main == *mem)
        {                                   //если после пользователь выбрал ДРУГОЙ массив, то
            *arr = *arr_main;               //запоминаем параметры старого и по аналогии
            *top = *top_main;               //со строкой 12(main.c) меняем выбранный массив
            *mem = *mem_main;
            *arr_main = *arr2;
            *top_main = *top2;
            *mem_main = *mem2;
            flag = true;                    //выбран массив 2 - поднимаем флажок
            printf("You can`t use \"dup\" with array N2!\n"); //предупреждаем, что мы не
//можем использовать некоторые команды со вторым массивом
        }
        else if(x == 1 and *arr_main == *arr2 and *top_main == *top2 and *mem_main == *mem2)
        {
            *arr2 = *arr_main;
            *top2 = *top_main;
            *mem2 = *mem_main;
            *arr_main = *arr;
            *top_main = *top;
            *mem_main = *mem;
        }
    }
    else
        printf("Array 2 don`t exist yet. Initialize it by \"dup\".\n");

    return flag;
}

unsigned int input_int() //стандартная функция проверки ввода целого беззнакового числа(в
//данном случае индекса)
{
    long long int index;  //создаём long long int, чтобы упростить жизнь, когда пользователь
//вводит отрицательные числа

    char check1 = scanf("%ld", &index), check2 = getchar();

    while(check1 != 1 or check2 != '\n' or index < 0)
    {
        if (check2 == -1)                   //если произошли ошибка ввода или конец ввода
        {
            fprintf(stderr, "\nError or EOF...\nExit the program");
            exit(1);
        }

        else
            printf("You`ve written incorrect integer. Try again:");

        scanf("%*[^\n]%*c");

        check1 = scanf("%ld", &index);
        check2 = getchar();
    }

    return (unsigned int) index;                        //явно приводим к беззнаковому
}



long double input_elem()         //стандартная функция проверки числа с плавающей запятой
{
    long double x;

    printf("Write value(float number) of element:");
```

```
    char check1 = scanf("%Lf", &x), check2 = getchar();

    while(check1 != 1 or check2 != '\n')
    {
        if (check2 == -1)                       //если произошли ошибка ввода или конец ввода
        {
            fprintf(stderr, "\nError or EOF...\nExit the program");
            exit(1);
        }

        else
            printf("You`ve written incorrect value. \nTry again:");

        scanf("%*[^\n]%*c");

        check1 = scanf("%Lf", &x);
        check2 = getchar();
    }

    return x;
}

long double *allocating(long double *arr, const unsigned int *top, unsigned int *mem)
//функция здорового выделения памяти
{
    unsigned int x = *(mem) - *(top); //разность текущих выделения памяти и заполненности
//массива
    long double *tmp = arr;

    if (x == 0)                         //выделение N памяти, если не хватает
    {
        tmp = (long double*)realloc(arr, (*mem + N) * sizeof(long double));
        *mem += N;
    }

    else if (x == N + 1)            //освобождение ячейки памяти, если много свободного места
        tmp = (long double*)realloc(arr, --(*mem) * sizeof(long double));

    if (NULL == tmp)                            //проверка выделения памяти
    {
        fprintf(stderr ,"\nFailed to allocate memory!\n");
        free(arr);
        exit(1);
    }

    return tmp;
}
```

Листинг 4 : Исходные коды программы main.o (файл: g e n e r a l . h)

```
#ifndef LAB3_GENERAL_H
#define LAB3_GENERAL_H
#include "checking.h"
long double* init_arr(unsigned int*, unsigned int*);
long double* input_arr(long double*, unsigned int*, unsigned int*);
long double* del_elem(long double*, unsigned int*, unsigned int*);
void print_arr(long double*, unsigned int);
void del_dup(long double**, unsigned int*, unsigned int *, long double**, unsigned
int*, unsigned int*);
#endif //LAB3_GENERAL_H
```

Листинг 5 : Исходные коды программы main.o (файл: g e n e r a l . c)

```
#include "general.h"

long double* init_arr(unsigned int *top, unsigned int *mem)//функция инициализации массива в
цикле
{
    printf("\nWrite size of array (old array will be deleted):");
```

```
    unsigned int size = input_int();
    *top = size, *mem = size;//заполненность массива будет такой же, сколько ввел пользователь,
//памяти потребуется столько же
    long double *arr = allocating(NULL, top, mem); //выделяем память под новый массив

    printf("\n\n");
    for (unsigned int i = 0; i < size; i++)     //заполняем в цикле
    {
        printf("Index %u:\n", i);
        arr[i] = input_elem();
        printf("\n");
    }

    return arr;
}

long double* input_arr(long double *arr, unsigned int *top, unsigned int *mem)
//функция ввода элемента по индексу
{
    printf("\nWrite integer index:");
    unsigned int index = input_int();
    long double value = input_elem();

    if (index >= *top)          //Вставка в конец массива
    {
        arr = allocating(arr, top, mem); //выделяем память в соответствии с индивидуальным
//заданием
        arr[*top] = value;      //просто вставляем в конец массива
    }

    else if(index < *top)       //Вставка по индексу со сдвигом
    {
        arr = allocating(arr, top, mem);//выделяем память в соответствии с индивидуальным
заданием
        for (unsigned int i = *top; i > index; i--) //сдвигаем некоторые элементы вправо
            arr[i] = arr[i - 1];
        arr[index] = value;     //просто вставляем в освободившуюся ячейку
    }
    (*top)++;                   //заполненность массива увеличилось на 1 элемент

    return arr;
}

long double* del_elem(long double *arr, unsigned int *top, unsigned int *mem)
//функция удаления элемента по индексу
{
    printf("\nWrite integer index:");

    unsigned int index = input_int();

    if (index > *top - 1 or *top == 0) //если указан пытаемся удалить то, что еще не существует
        printf("\nYou haven`t deleted anything!\n");




       else                     //сдвигаем элементы влево и освобождаем одну ячейку памяти
    {
        for (unsigned int i = index; i < *top - 1; i++)
            arr[i] = arr[i + 1];
        arr = allocating(arr, top, mem); //выделяем память в соответствии с индивидуальным
//заданием
        (*top)--;               //заполненность массива уменьшилось
        printf("\nDeleted!\n");
```

```
    }

    return arr;
}

void print_arr(long double *arr, unsigned int top)    //функция вывода массива
{
  if (top == 0)                                       //вывод пустого массива
       printf("Array is empty!");

  printf("\n");
  for (unsigned int i = 0; i < top; i++) //массив непрерывен, поэтому можем выводить просто в
//цикле, перебирая индексы
      printf("a[%u] = %Lf\n", i, arr[i]);
}

void del_dup(long double **arr, unsigned int *top, unsigned int *mem, long double **arr2,
unsigned int *top2, unsigned int *mem2)
{   //функция из для индивидуального задания
    unsigned int top_tmp = *top, mem_tmp = *top;  //создаем новый массив, куда будем копировать
    unsigned int j = 0;                 //последовательность без подряд стоящих равных элементов
    long double *tmp = allocating(NULL, &top_tmp, &mem_tmp);
//выделяем памяти прозапасом столько, насколько заполнен основной массив,
//потому что элементы могут не повторяться

    unsigned int new_top = *top, new_mem = *top;   //так же создаем второй новый массив, куда
//будет копироваться последовательность только из подряд стоящих повторяющихся элементов
    unsigned int k = 0;
    long double *new_arr = allocating(NULL, &new_top, &new_mem);

    bool flag = false;   //флажок нужен, чтобы во второй новый массив копировалось корректное
//количество элементов

    unsigned int i = 0; //i не в цикле, потому что нам нужно будет корректно скопировать
//последний элемент главного массива

    for(; i < *top - 1; i++)          //последовательно в главном массиве проверяем по парам,
равны ли элементы. В зависимости от этого копируем элементы в соотвествующий новый массив
        if((*arr)[i] != (*arr)[i+1])
        {
            if(flag) //если на предыдущей итерации пара элементов совпала,
            {        //а на текущей нет, нужно во второй новый массив добавить еще  один
//элемент для корректного их количества
                new_arr[k++] = (*arr)[i];
                flag = false;
            }
            tmp[j++] = (*arr)[i];
        }
        else
        {
            new_arr[k++] = (*arr)[i];
            flag = true;
        }




       if ((*arr)[i - 1] == (*arr)[i]) //проверяем, корректно ли будет вставить во второй новый
//массив последний элемент главного массива
        new_arr[k++] = (*arr)[i];
    tmp[j++] = (*arr)[i];  //в первый новый массив всегда вставляем последний элемент главного
//массива

    free(*arr);             //освобождаем далее не нужный arr

    top_tmp = j, mem_tmp = j;//присваиваем параметрам первого нового массива его характеристики
```

```
    tmp = allocating(tmp, &top_tmp, &mem_tmp);  //освобождаем ненужную память

    *top = top_tmp, *mem = mem_tmp; //делаем главный массив равным первому новому, старый
//главный нам больше не нужен
    *arr = tmp;

    *top2 = k, *mem2 = k; //присваиваем параметрам второго нового массива его "характеристики"
    new_arr = allocating(new_arr, top2, mem2);  //освобождаем ненужную память
    *arr2 = new_arr; //делаем второй массив равным второму новому в соответствии с
//индивидуальным заданием
}
```

5. Описание тестовых примеров

Таблица 1. Тестирование №1 программы main.o

Шаг	Ввод	Ожидаемое значение	Полученное значение
1	zdravstvuyte	You`ve wrote wrong command. Try again:	You`ve wrote wrong command. Try again:
2	*Любая последовательность символов, которая не совпадает с заданными командами*	You`ve wrote wrong command. Try again:	You`ve wrote wrong command. Try again:
3	init	Write size of array (old array will be deleted):	Write size of array (old array will be deleted):
4	7	Index 0: Write value(float number) of element:	Index 0: Write value(float number) of element:
5	1	Index 1: Write value(float number) of element:	Index 1: Write value(float number) of element:
6	1	Index 2: Write value(float number) of element:	Index 2: Write value(float number) of element:
7	2	Index 3: Write value(float number) of element:	Index 3: Write value(float number) of element:
8	3	Index 4: Write value(float number) of element:	Index 4: Write value(float number) of element:
9	4	Index 5: Write value(float number) of element:	Index 5: Write value(float number) of element:
10	4	Index 6: Write value(float number) of element:	Index 6: Write value(float number) of element:
11	5	You are using array N1 MENU: init input delete print dup choose help exit (Дальше будем называть это MENU + номер используемого массива)	You are using array N1 MENU: init input delete print dup choose help exit
12	input	Write integer index:	Write integer index:
13	*Что нибудь, кроме положительного целого числа и нуля*	You`ve written incorrect integer. Try again:	You`ve written incorrect integer. Try again:
14	0	Write value(float number) of element:	Write value(float number) of element:
15	*Что нибудь, кроме одного любого числа*	You`ve written incorrect value. Try again:	You`ve written incorrect value. Try again:
16	1001.2562782	MENU1	MENU1
17	print	a[0] = 1001.256278 a[1] = 1.000000 a[2] = 1.000000 a[3] = 2.000000 a[4] = 3.000000 a[5] = 4.000000 a[6] = 4.000000 a[7] = 5.000000	a[0] = 1001.256278 a[1] = 1.000000 a[2] = 1.000000 a[3] = 2.000000 a[4] = 3.000000 a[5] = 4.000000 a[6] = 4.000000 a[7] = 5.000000
18	input	Write integer index:	Write integer index:
19	7	Write value(float number) of element:	Write value(float number) of element:
20	3.14	MENU1	MENU1
21	input	Write integer index:	Write integer index:
22	252	Write value(float number) of element:	Write value(float number) of element:
23	2.71	MENU1	MENU1
24	delete	Write integer index:	Write integer index:
25	8	Deleted!	Deleted!
26	print	a[0] = 1001.256278 a[1] = 1.000000 a[2] = 1.000000 a[3] = 2.000000 a[4] = 3.000000 a[5] = 4.000000 a[6] = 4.000000 a[7] = 3.140000 a[8] = 2.710000	a[0] = 1001.256278 a[1] = 1.000000 a[2] = 1.000000 a[3] = 2.000000 a[4] = 3.000000 a[5] = 4.000000 a[6] = 4.000000 a[7] = 3.140000 a[8] = 2.710000
27	choose	Array 2 don`t exist yet. Initialize it by «dup».	Array 2 don`t exist yet. Initialize it by «dup».
28	dup	MENU1	MENU1
29	print	a[0] = 1001.256278 a[1] = 1.000000 a[2] = 2.000000 a[3] = 3.000000 a[4] = 4.000000 a[5] = 3.140000 a[6] = 2.710000	a[0] = 1001.256278 a[1] = 1.000000 a[2] = 2.000000 a[3] = 3.000000 a[4] = 4.000000 a[5] = 3.140000 a[6] = 2.710000
30	input	Write integer index:	Write integer index:
31	0	Write value(float number) of element:	Write value(float number) of element:
32	300	MENU1	MENU1
33	delete	Write integer index:	Write integer index:
34	256	You haven`t deleted anything!	You haven`t deleted anything!
35	delete	Write integer index:	Write integer index:
36	4	Deleted!	Deleted!
37	print	a[0] = 300.000000 a[1] = 1001.256278 a[2] = 1.000000 a[3] = 2.000000 a[4] = 4.000000 a[5] = 3.140000 a[6] = 2.710000	a[0] = 300.000000 a[1] = 1001.256278 a[2] = 1.000000 a[3] = 2.000000 a[4] = 4.000000 a[5] = 3.140000 a[6] = 2.710000
38	choose	Choose array(1 or 2):	Choose array(1 or 2):
39	*Любую последовательность символов кроме 1 и 2*	You haven`t wrote 1 or 2. Try again:	You haven`t wrote 1 or 2. Try again:
40	2	You can`t use "dup" with array N2! MENU2	You can`t use "dup" with array N2! MENU2

41	print	a[0] = 1.000000 a[1] = 1.000000 a[2] = 4.000000 a[3] = 4.000000	a[0] = 1.000000 a[1] = 1.000000 a[2] = 4.000000 a[3] = 4.000000
42	input	Write integer index:	Write integer index:
43	0	Write value(float number) of element:	Write value(float number) of element:
44	1000	MENU2	MENU2
45	input	Write integer index:	Write integer index:
46	252	Write value(float number) of element:	Write value(float number) of element:
47	0	MENU2	MENU2
48	delete	Write integer index:	Write integer index:
49	1	Deleted!	Deleted!
50	delete	Write integer index:	Write integer index:
51	500	You haven`t deleted anything!	You haven`t deleted anything!
52	print	a[0] = 1000.000000 a[1] = 1.000000 a[2] = 4.000000 a[3] = 4.000000 a[4] = 0.000000	a[0] = 1000.000000 a[1] = 1.000000 a[2] = 4.000000 a[3] = 4.000000 a[4] = 0.000000
53	help	Write "init" if you want to initialize array. Write "input" if you want to input element into array by index. Write "delete" if you want to delete the element from array by index. Write "print" if you want to see your array. Write "choose" if you want to choose array. Write "help" if you want to read this message again. Write "exit" if you want to exit the program.	Write "init" if you want to initialize array. Write "input" if you want to input element into array by index. Write "delete" if you want to delete the element from array by index. Write "print" if you want to see your array. Write "choose" if you want to choose array. Write "help" if you want to read this message again. Write "exit" if you want to exit the program.
54	dup	You can`t use "dup" with array N2!	You can`t use "dup" with array N2!
55	init	Write size of array (old array will be deleted):	Write size of array (old array will be deleted):
56	3	Index 1: Write value(float number) of element:	Index 1: Write value(float number) of element:
57	1	Index 2: Write value(float number) of element:	Index 2: Write value(float number) of element:
58	2	Index 3: Write value(float number) of element:	Index 3: Write value(float number) of element:
59	3	MENU2	MENU2
60	print	a[0] = 1.000000 a[1] = 2.000000 a[2] = 3.000000	a[0] = 1.000000 a[1] = 2.000000 a[2] = 3.000000
61	choose	Choose array(1 or 2):	Choose array(1 or 2):
62	1	MENU1	MENU1
63	init	Write size of array (old array will be deleted):	Write size of array (old array will be deleted):
64	5	Index 1: Write value(float number) of element:	Index 1: Write value(float number) of element:
65	10	Index 2: Write value(float number) of element:	Index 2: Write value(float number) of element:
66	10	Index 3: Write value(float number) of element:	Index 3: Write value(float number) of element:
67	10	Index 4: Write value(float number) of element:	Index 4: Write value(float number) of element:
68	20	Index 5: Write value(float number) of element:	Index 5: Write value(float number) of element:
69	20	MENU1	MENU1
70	dup	MENU1	MENU1
71	print	a[0] = 10.000000 a[1] = 20.000000	a[0] = 10.000000 a[1] = 20.000000
72	choose	Choose array(1 or 2):	Choose array(1 or 2):
73	2	You can`t use "dup" with array N2! MENU2	You can`t use "dup" with array N2! MENU2
74	print	a[0] = 10.000000 a[1] = 10.000000 a[2] = 10.000000 a[3] = 20.000000 a[4] = 20.000000	a[0] = 10.000000 a[1] = 10.000000 a[2] = 10.000000 a[3] = 20.000000 a[4] = 20.000000
75	exit	Bye!	Bye!

Таблица 2. Тестирование №2 программы main.o

Шаг	**Ввод**	**Ожидаемое значение**	**Полученное значение**
1	Сочетание клавиш Ctrl + d	Error or EOF Exit the program	Error or EOF Exit the program

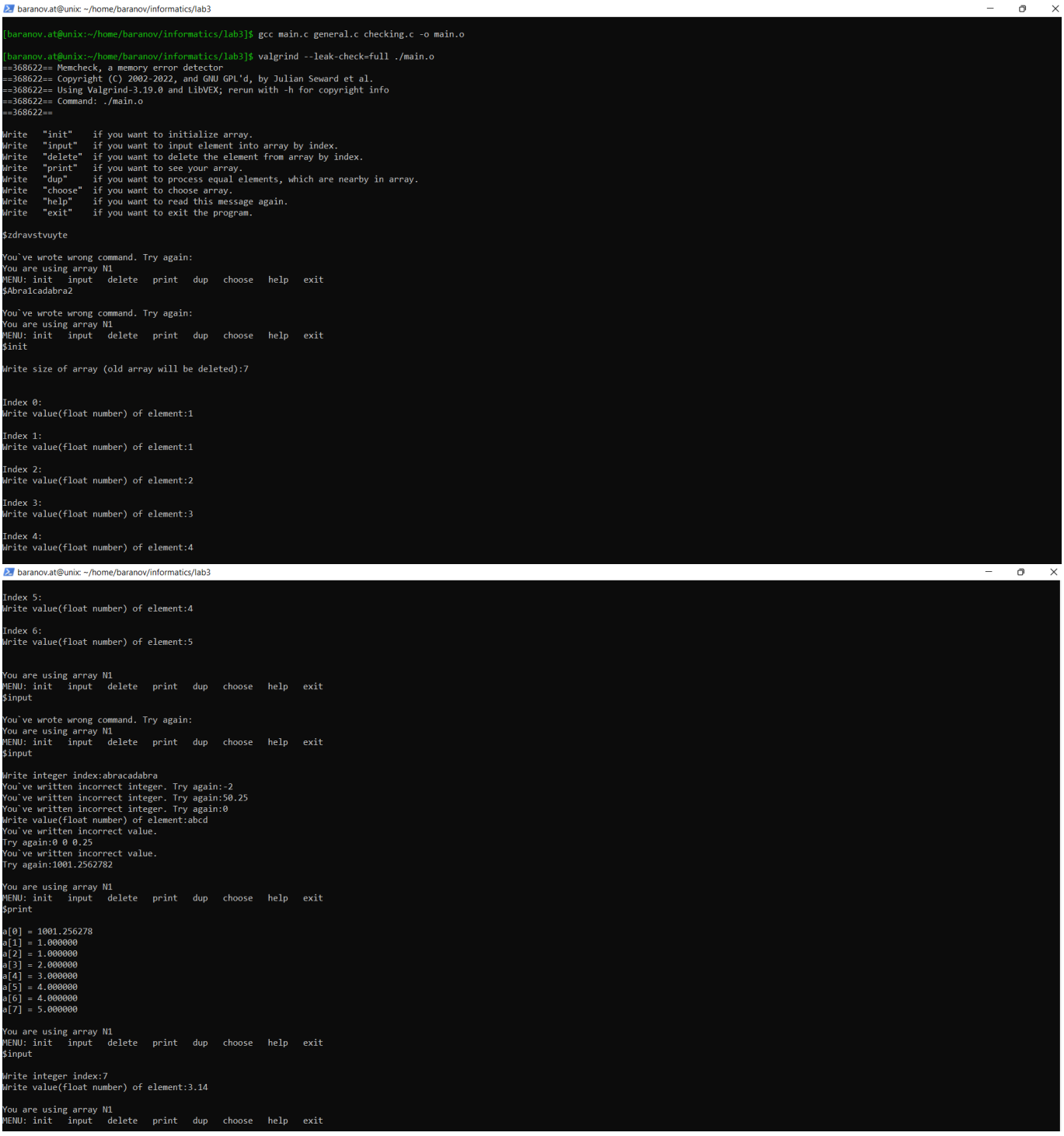
Таблица 3. Тестирование №3 программы main.o

Шаг	**Ввод**	**Ожидаемое значение**	**Полученное значение**
1	init	Write integer index:	Write integer index:
2	Сочетание клавиш Ctrl + d	Error or EOF Exit the program	Error or EOF Exit the program

Таблица 4. Тестирование №4 программы main.o

Шаг	**Ввод**	**Ожидаемое значение**	**Полученное значение**
1	init	Write integer index:	Write integer index:
2	2	Write value(float number) of element:	Write value(float number) of element:
3	Сочетание клавиш Ctrl + d	Error or EOF Exit the program	Error or EOF Exit the program

6. Скриншоты

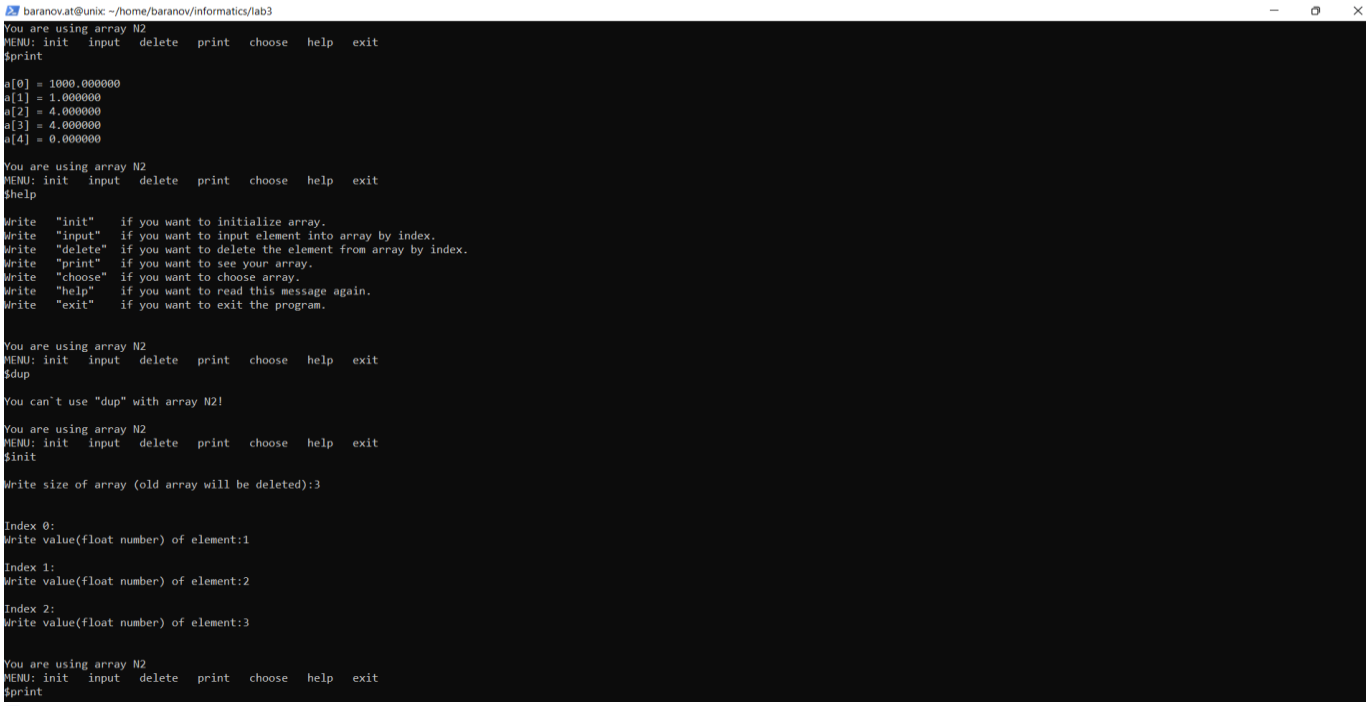

baranov.at@unix: ~/home/baranov/informatics/lab3
[baranov.at@unix:~/home/baranov/informatics/lab3]$ gcc main.c general.c checking.c -o main.o
[baranov.at@unix:~/home/baranov/informatics/lab3]$ valgrind --leak-check=full ./main.o
==368622== Memcheck, a memory error detector
==368622== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
==368622== Using Valgrind-3.19.0 and LibVEX; rerun with -h for copyright info
==368622== Command: ./main.o
==368622==
Write "init" if you want to initialize array.
Write "input" if you want to input element into array by index.
Write "delete" if you want to delete the element from array by index.
Write "print" if you want to see your array.
Write "dup" if you want to process equal elements, which are nearby in array.
Write "choose" if you want to choose array.
Write "help" if you want to read this message again.
Write "exit" if you want to exit the program.
$zdravstvuyte
You`ve wrote wrong command. Try again:
You are using array N1
MENU: init input delete print dup choose help exit
$Abra1cadabra2
You`ve wrote wrong command. Try again:
You are using array N1
MENU: init input delete print dup choose help exit
$init
Write size of array (old array will be deleted):7
Index 0:
Write value(float number) of element:1
Index 1:
Write value(float number) of element:1
Index 2:
Write value(float number) of element:2
Index 3:
Write value(float number) of element:3
Index 4:
Write value(float number) of element:4

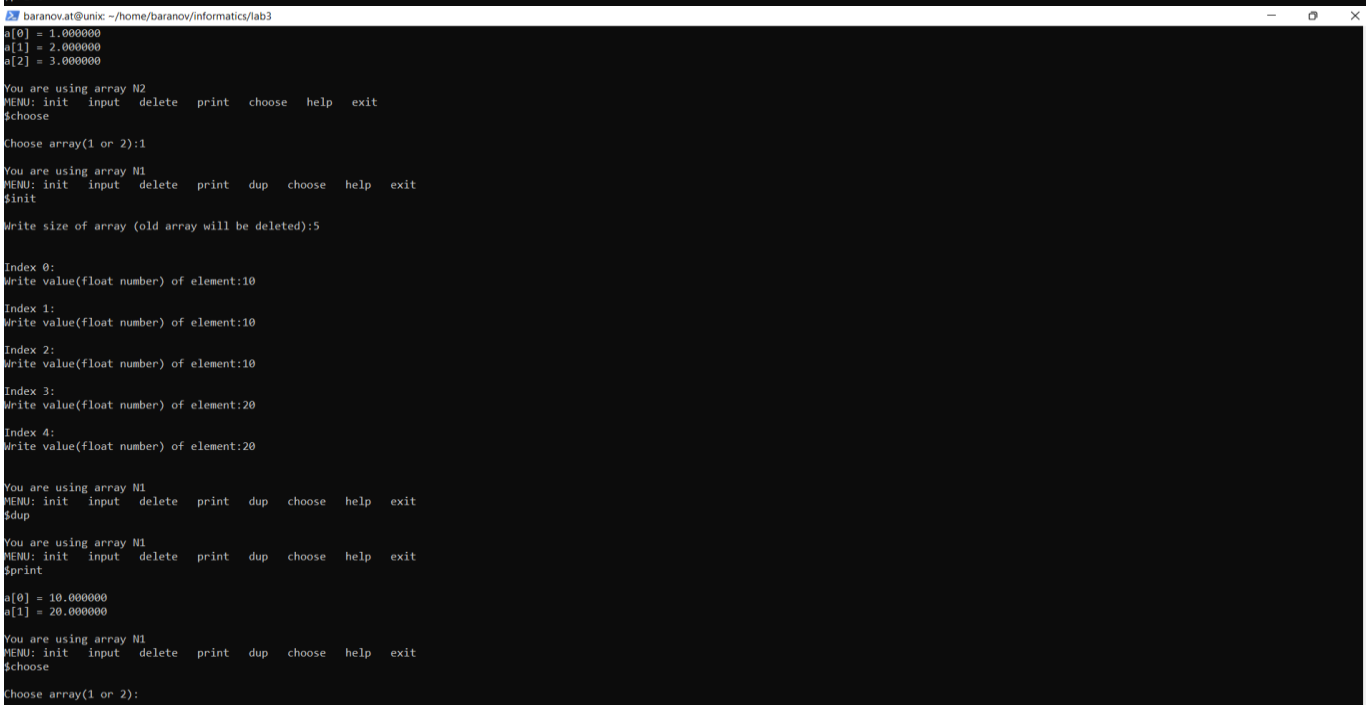
baranov.at@unix: ~/home/baranov/informatics/lab3
Index 5:
Write value(float number) of element:4
Index 6:
Write value(float number) of element:5
You are using array N1
MENU: init input delete print dup choose help exit
$input
You`ve wrote wrong command. Try again:
You are using array N1
MENU: init input delete print dup choose help exit
$input
Write integer index:abracadabra
You`ve written incorrect integer. Try again:-2
You`ve written incorrect integer. Try again:50.25
You`ve written incorrect integer. Try again:0
Write value(float number) of element:abcd
You`ve written incorrect value.
Try again:0 0 0.25
You`ve written incorrect value.
Try again:1001.2562782
You are using array N1
MENU: init input delete print dup choose help exit
$print
a[0] = 1001.256278
a[1] = 1.000000
a[2] = 1.000000
a[3] = 2.000000
a[4] = 3.000000
a[5] = 4.000000
a[6] = 4.000000
a[7] = 5.000000
You are using array N1
MENU: init input delete print dup choose help exit
$input
Write integer index:7
Write value(float number) of element:3.14
You are using array N1
MENU: init input delete print dup choose help exit

```
baranov.at@unix: ~/home/baranov/informatics/lab3
Write integer index:252
Write value(float number) of element:2.71

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$delete

Write integer index:8

Deleted!

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$print

a[0] = 1001.256278
a[1] = 1.000000
a[2] = 1.000000
a[3] = 2.000000
a[4] = 3.000000
a[5] = 4.000000
a[6] = 4.000000
a[7] = 3.140000
a[8] = 2.710000

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$choose
Array 2 don`t exist yet. Initialize it by "dup".

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$dup

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$print

a[0] = 1001.256278
a[1] = 1.000000
a[2] = 2.000000
a[3] = 3.000000
a[4] = 4.000000
a[5] = 3.140000
a[6] = 2.710000

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$
```

```
baranov.at@unix: ~/home/baranov/informatics/lab3
a[6] = 2.710000

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$input

Write integer index:0
Write value(float number) of element:300

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$delete

Write integer index:256

You haven`t deleted anything!

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$delete

Write integer index:4

Deleted!

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$print

a[0] = 300.000000
a[1] = 1001.256278
a[2] = 1.000000
a[3] = 2.000000
a[4] = 4.000000
a[5] = 3.140000
a[6] = 2.710000

You are using array N1
MENU: init   input   delete   print   dup   choose   help   exit
$choose

Choose array(1 or 2):do svidanya
You haven`t wrote 1 or 2. Try again:poka poka
You haven`t wrote 1 or 2. Try again:2
You can`t use "dup" with array N2!

You are using array N2
MENU: init   input   delete   print   choose   help   exit
$
```

```
baranov.at@unix: ~/home/baranov/informatics/lab3
You are using array N2
MENU: init   input   delete   print   choose   help   exit
$print

a[0] = 1.000000
a[1] = 1.000000
a[2] = 4.000000
a[3] = 4.000000

You are using array N2
MENU: init   input   delete   print   choose   help   exit
$input

You`ve wrote wrong command. Try again:
You are using array N2
MENU: init   input   delete   print   choose   help   exit
$0

You`ve wrote wrong command. Try again:
You are using array N2
MENU: init   input   delete   print   choose   help   exit
1000
$input

Write integer index:0
Write value(float number) of element:1000

You are using array N2
MENU: init   input   delete   print   choose   help   exit
$input

Write integer index:252
Write value(float number) of element:0

You are using array N2
MENU: init   input   delete   print   choose   help   exit
$delete

Write integer index:1

Deleted!

You are using array N2
MENU: init   input   delete   print   choose   help   exit
$delete

Write integer index:500

You haven`t deleted anything!
```





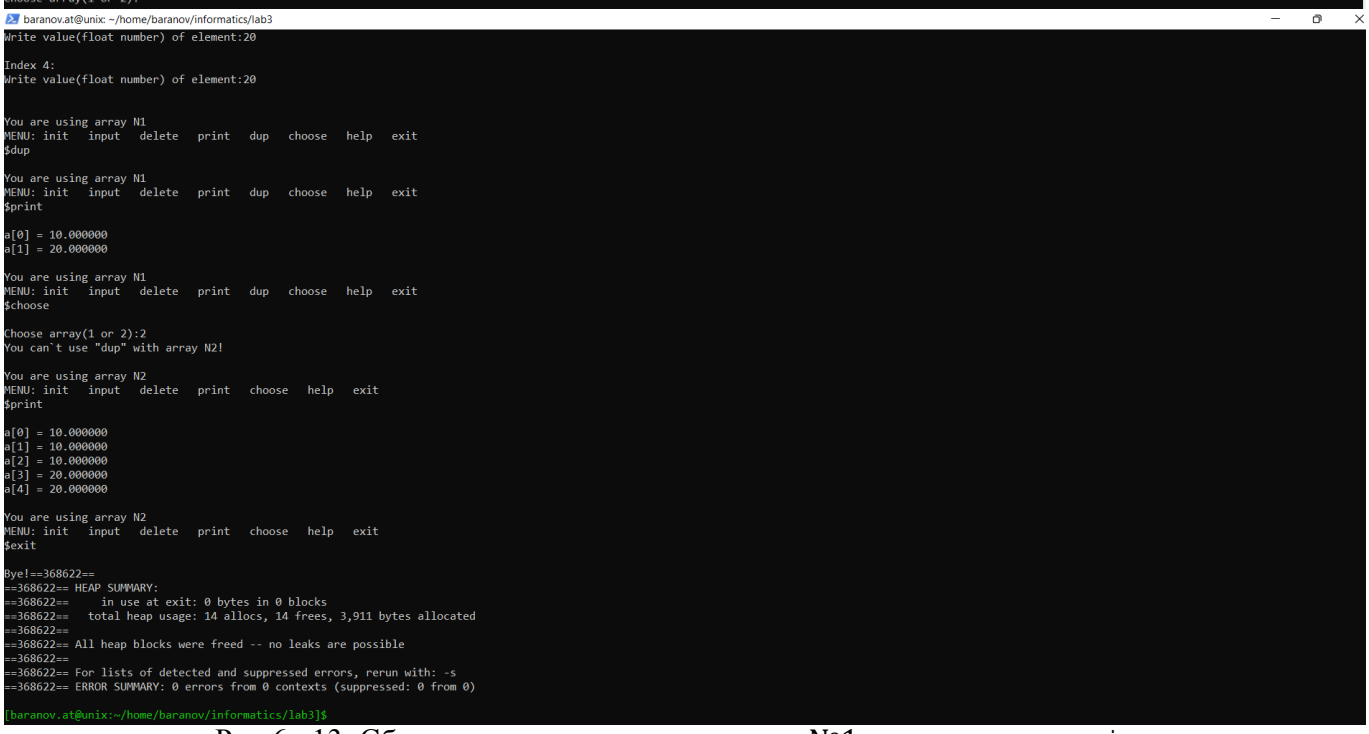


Рис.6- 13: Сборка,запуск и тестирование №1 программы main.o

Рис.14: Запуск и тестирование №2 и №3 программы main.o

Рис.15: Запуск и тестирование №4 программы main.

7. Выводы

В ходе выполнения данной работы на примере программы, выполняющей обработку массива, были рассмотрены принципы построения использующих массивы программ на языке C .

1. Организация ввода/вывода.
2. Разработка функций.
3. Объявление и использование переменных.
4. Инициализация массивов в цикле, добавление и удаление его элементов.
5. Работа с динамическим выделением и освобождением памяти.
6. Обработка массива, удаление рядом стоящих одинаковых элементов.