ТЕХНИЧЕСКИ УНИВЕРСИТЕТ – ГАБРОВО

Катедра „Компютърни системи и технологии“

КУРСОВА РАБОТА

Тема:.....

Разработил:.....

Фак. №:..... Курс:..... Специалност:.....

Проверил:.....

Подпис:..... Дата:.....

Функцията се състои от две части – заглавна и тяло. В заглавната част се дефинира типа на функцията, нейното име и списък с параметри, докато в тялото се съдържат инициализация на локални променливи в дадена функция, заедно с оператори и конструкции. Вида на параметрите е като при инициализацията на променлива (тип ИмеНаПроменлива), обаче в този случай можем да декларираме голям брой променливи, за това трябва да се отделят със запетаи. В зависимост от типа функция, може да се наложи да се изведе поне един израз, променлива или каквато и да е друга стойност. Тези функции са от типовете *int*, *float*, *double* и в тях трябва задължително да използваме оператора *return*, за да ни върне стойност. Но съществуват и функции, на които не е необходимо да връщат непременно резултат, а именно функцията от тип *void*. В програмен език С функциите се извикват като изпишем името на функцията, следвано от малки скоби, в които се изписва списъкът с параметрите. Теоретически това ще работи, но тъй като сме в програмен език С, ние ще трябва да декларираме функцията, защото в противен случай компилатора ще ни върне грешка и няма да изпълни конзолното приложение. Общия вид на функцията е следния:

```
тип ИмеНаФункцията (тип1 Параметър1, тип2 Параметър2, ...) // заглавна част
{
	// тяло на функцията
	// локални променливи
	// оператори
}
```

Масивите са структура, в която се състоят последователно наредени елементи от един и същи тип от данни. За да имаме достъп до определен елемент от масива, изписва се името на масива следвано от номера на елемента, до който се опитваме да достъпим. Общия вид и декларация на масива е следния:

```
тип ИмеНаМасив[РазмерN] = {елемент1, елемент2, ... , елементN};
```

Размерността на масива е от 0 до n-1. Масивите имат няколко особености. Първата особеност е, че масива е структуриран от еднотипни видове елементи. Т. е. в масива се срещат данни само от един тип. Второ – масива притежава данни от статичен вид. Тук броя на елементите се определя по време на компилация, като броят на елементите в масива винаги е константен. В програмен език С съществуват масиви, които са едномерни и многомерни. Едномерните масиви са редица от последователно наредени елементи от един и същ тип. Многомерните масиви са сбор от едномерни масиви. Те се използват често за обработката на матрици в програмата. Общия вид на многомерен масив ще го представим като двумерен масив, но съществуват и тримерен, и четиримерен и т. н.

```
тип ИмеНаМасив[РазмерY][РазмерX];
```

Използвана литература:

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

struct TrapsSides {
    int a;
    int b;
    int c;
    int d;
    int h;
};
// Деклариране на цял брой трапеци
int brTraps;
// Деклариране на лицето на текущия трапец
double S;
// Декларация на функция за сумиране на лицата на трапеците
double SumTrap(float a, float b, float c, float d, float h, float r);
// Декларация на функции
void EnterBrTraps();
void EnterAndCalSums();
void SortSes();
// Масива с всичките стойности за лицата на трапеците
float Ses[50];

// Самите стойности на текущия трапец в масив
float n[6];

int main()
{

    EnterBrTraps();
    if (brTraps > 50) {
        printf("Ne mojesh da nadvishavash poveche ot 50 trapetsa.\n");
        return 0;
    }
    EnterAndCalSums();
    SortSes();

    return 0;

}

void EnterAndCalSums() {
    // Разглеждаме трапеците, които сме въвели първоначално
    for (int t = 0; t < brTraps; t++)
    {
        // Изписваме страните, височината или радиуса.
        printf("Vuvedi stranite na trapets %d: \n",t);
        for (int i = 0; i < 6; i++)
        {
            // Проверка за текуща променлива, на която задаваме стойност
            switch (i) {
            case 0:printf("a = ");
                scanf_s("%f", &n[0]);
                break;
            case 1:printf("b = ");
                scanf_s("%f", &n[1]);
                break;
            case 2:printf("c = ");
                scanf_s("%f", &n[2]);
                break;
            case 3:printf("d = ");
                scanf_s("%f", &n[3]);
                break;
```

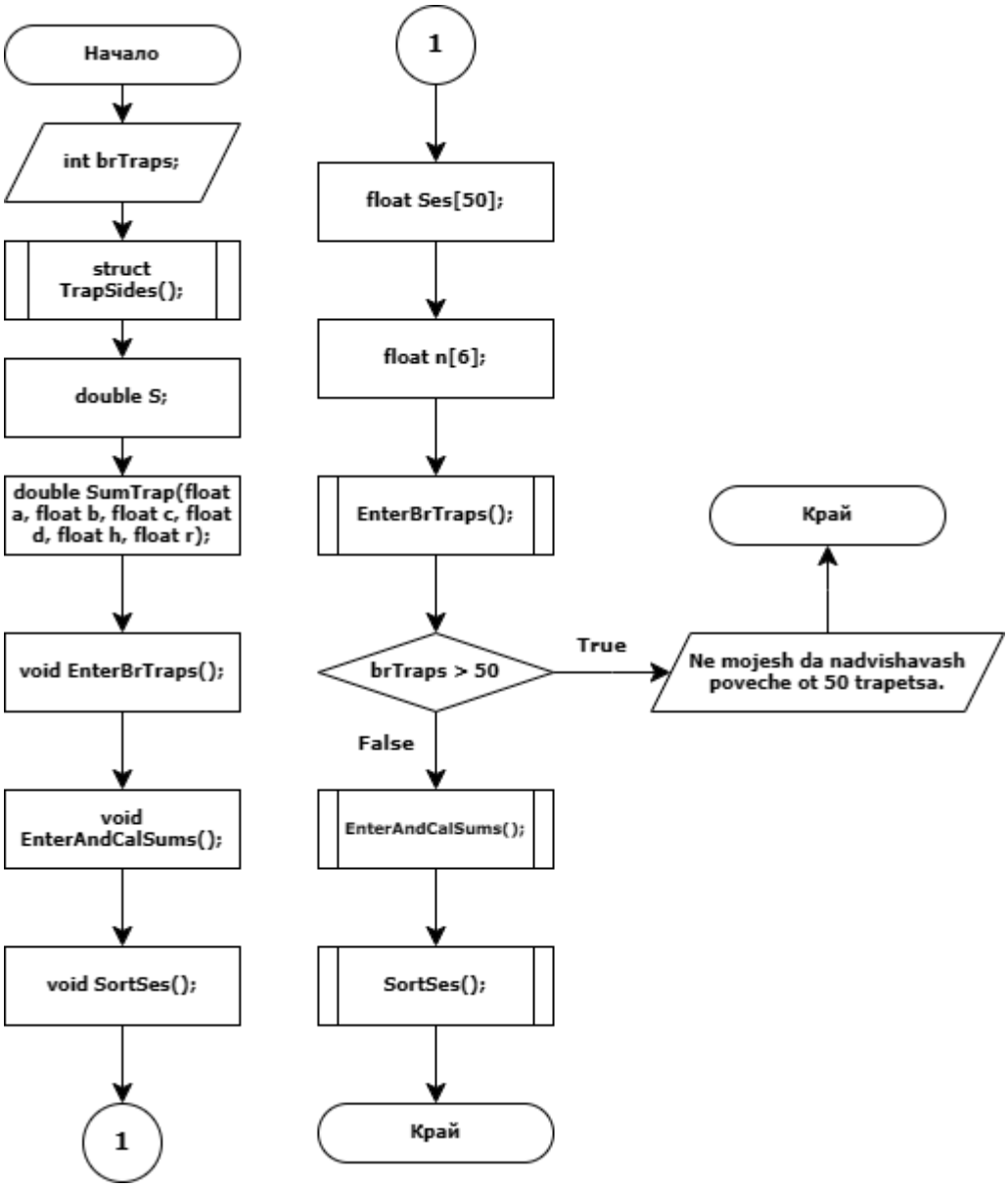
```
            case 4:printf("h = ");
                scanf_s("%f", &n[4]);
                break;
            case 5:printf("r = ");
                scanf_s("%f", &n[5]);
                break;
            }
        }
        // Изпълняваме функция за сумиране на лицето
        SumTrap(n[0], n[1], n[2], n[3], n[4], n[5]);
        // Задаваме текущия елемент от масива да е равен на лицето на текущия
трапец (Записваме лицето)
        Ses[t] = S;
    }
}

void EnterBrTraps() {
    // Въвеждаме броя на трапеците, които ще пресмятаме
    printf("Broi Trapetsi: ");
    scanf_s("%d", &brTraps);
    printf("\n");
}

double SumTrap(float a, float b, float c, float d, float h, float r) {

    // Формула за лице чрез две страни и височина
    if (a!=0 && b!=0 && c==0 && d==0 && h!=0 && r==0) {
        S = 0.5 * (a + b) * h;
        printf("S = %.2lf\n\n", S);
    }
    // Формула за лице чрез четирите страни на трапеца
    else if (a!=0 && b!=0 && c!=0 && d!=0 && h==0 && r==0) {
        S = (a + c) / (4 * (a - c)) * sqrt((a + b - c + d) * (a - b - c + d) * (a
+ b - c - d) * (-a + b + c + d));
        printf("S = %.2lf\n\n", S);
    }
    // Формула за лице чрез височината и радиуса на описаната окръжност
    else if (a != 0 && b != 0 && c != 0 && d != 0 && h == 0 && r == 0) {
        S = 8 * pow(r, 2);
        printf("S = %.2lf\n\n", S);
    }
    // Когато нито едно от горепосочените не е изпълнено
    else {
        printf("Ima problem.");
    }

    return 0;
}

void SortSes() {
    // Сортиране на масива с лицата по големината им във възходящ ред
    for (int u = 0; u < brTraps; u++)
    {
        for (int p = u + 1; p < brTraps; p++)
        {
            if (Ses[u] > Ses[p]) {
                float Mid = Ses[u];
                Ses[u] = Ses[p];
                Ses[p] = Mid;
            }
        }
    }
```

```
    // Изнасяне на резултата накрая
    for (int i = 0; i < brTraps; i++)
    {
        printf("S na trapets %d e: %.2f\n", i, Ses[i]);
    }
}
```

Резултат от програмния код:

```
Broi Trapetsi: 2

Vuvedi stranite na trapets 0:
a = 4
b = 7
c = 0
d = 0
h = 4
r = 0
S = 22.00

Vuvedi stranite na trapets 1:
a = 3
b = 6
c = 0
d = 0
h = 6
r = 0
S = 27.00

S na trapets 0 e: 22.00
S na trapets 1 e: 27.00

D:\VS Applications\LabsApps\KursRab\x64\Debug\KursRab.exe (process 19492) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the conso
le when debugging stops.
Press any key to close this window . . .
```

Блок схема:


Начало
int brTraps;
struct TrapSides();
double S;
double SumTrap(float a, float b, float c, float d, float h, float r);
void EnterBrTraps();
void EnterAndCalSums();
void SortSes();
1
1
float Ses[50];
float n[6];
EnterBrTraps();
brTraps > 50
True
Ne mojesh da nadvishavash poveche ot 50 trapetsa.
Край
False
EnterAndCalSums();
SortSes();
Край