

# Todo list

# Программирование

А. В. Власова

24 декабря 2015 г.

# Глава 1

## Основные конструкции языка

### 1.1 Задание 1

#### 1.1.1 Задание

Пользователь задает длину отрезка в саженьях, аршинах и вершках (например, 8 сажень 2 аршина 11.4 вершка). Определить длину того же отрезка в метрах (в данном случае 19). 1 сажень = 3 аршина = 48 вершков, 1 вершок = 4.445 см.

#### 1.1.2 Теоритические сведения

В процессе написания кода программы были использованы функции ввода/вывода `printf()`, `scanf()` и `puts()`, содержащиеся в стандартном заголовочном файле `stdio.h`.

#### 1.1.3 Проектирование

Среда разработки: QT Creator 3.5.1 (opensource)

Компилятор: GCC

В программе разделены взаимодействие с пользователем и бизнес-логика.

`trancfer_to_meters.c`

считывает из консоли введенное пользователем количество сажень, аршинов и вершков, после чего передает их функции

`double trancfer_to_meters()`

.

`trancfer_to_meter.c`

переводит полученные данные в метры. Окончательный результат выводится в консоль.

Для контроля исправности программы были созданы модульные тесты, расположенные в файле

`tst_testtest.cpp`

.

### 1.1.4 Тестовый план и результаты тестирования

При проведении модульного тестирования программа вызывает функцию

`void trancfer_test()`

Код тестируется с помощью макроса `QCOMPARE()`, который сравнивает значение, полученное в функции

`double trancfer_to_meters()`,

и ожидаемый результат. При несовпадении данных, выполнение программы завершается сообщением о непройденном тесте.

### 1.1.5 Выводы

Выполнение задания позволило освоить процесс разделения программы на взаимодействие с пользователем и бизнес-логику, а также приобрести некоторые навыки отладки кода и исправления ошибок, выводимых компилятором.

## Листинги

```
1 #include <stdio.h>
2 #include "file_clear.h"
3 #include "search_sine.h"
4 #include "shahmati_ugroza.h"
5 #include "trancfer_to_meters.h"
6 #include "palindromes_read.h"
7 #include "main.h"
8
9 int main()
10 {
```

```

11 puts ("1. transfer to meters");
12 puts ("2. search chess threat");
13 puts ("3. search sine");
14 puts ("4. search clear in array");
15 puts ("5. search palindromes");
16
17 int choice;
18 scanf ("%d", &choice);
19 switch (choice) {
20 case 1:
21     puts ("transfer to meters");
22
23     trancfer();
24     break;
25
26
27
28
29 case 2:
30     puts ("search chess threat");
31
32     search_threat_reading();
33     break;
34
35
36
37
38 case 3:
39     puts ("search sine");
40
41     search_sine();
42     break;
43
44
45
46
47 case 4:
48     puts ("search clear in array");
49
50     file_clear();
51     break;
52
53
54
55
56 case 5:
57     puts ("search palindromes");
58
59     file_open();

```

```

60         break;
61
62
63
64     }
65     return (0);
66
67 }

```

```

1  #include <stdio.h>
2  #include "trancfer_to_meters.h"
3  #include "trancfer_to_meter.h"
4
5  void trancfer()
6  {
7      double sazhen, arshin, vershok;
8      double result1 = 0;
9      puts("Введите через пробел количество сажень, аршинов и
          вершков");
10     scanf("%lf %lf %lf", &sazhen, &arshin, &vershok);
11     result1 = trancfer_to_meters(sazhen, arshin, vershok);
12     printf("%lf м", result1);
13
14 }

```

```

1  #ifndef TRANSFER_TO_METERS_H
2  #define TRANSFER_TO_METERS_H
3
4  void trancfer();
5
6  #endif // TRANSFER_TO_METERS_H

```

```

1  #include <stdio.h>
2  #include "trancfer_to_meter.h"
3
4  double trancfer_to_meters (double sazhen, double arshin,
          double vershok)
5  {
6      double m;
7      double sm;
8      arshin = sazhen * 3 + arshin;
9      vershok = arshin * 16 + vershok;
10     sm = vershok * 4.445;
11     m = sm/100;
12
13     return(m);
14
15 }

```

```

1 #ifndef TRANCFER_TO_METER_H
2 #define TRANCFER_TO_METER_H
3
4 #ifdef __cplusplus
5 extern "C"{
6 #endif
7 double trancfer_to_meters (double, double, double);
8 #ifdef __cplusplus
9 }
10 #endif
11 #endif // TRANCFER_TO_METER_H

```

```

1 #include <QString>
2 #include <QtTest>
3 #include "search_clear.h"
4 #include "search_threat.h"
5 #include "sinx.h"
6 #include "trancfer_to_meter.h"
7
8
9 class TestTest : public QObject
10 {
11     Q_OBJECT
12
13
14 private Q_SLOTS:
15     void testCase1();
16     void search_sine_test();
17     void search_threat_test();
18     void trancfer_test();
19 };
20
21
22 void TestTestCase1()
23 {
24     QVERIFY2(true, "Failure");
25 }
26 void search_sine_test()
27 {
28     QCOMPARE(sinx(0.5, 0.001), 0.5);
29 }
30
31 void search_threat_test()
32 {
33     QCOMPARE(search_threat(2, 4, 1, 2, 3, 3), "threat from
34         elephant");
35 }

```

```

36 void tranfer_test()
37 {
38 QCOMPARE(tranfer_to_meters(8, 2, 11.4), 19);
39 }
40
41 QTEST_APPLESS_MAIN(TestTest)
42
43 #include "tst_testtest.moc"

```

## 1.2 Задание 2

### 1.2.1 Задание

На шахматной доске стоят черный король и белые ладья и слон (ладья бьет по горизонтали и вертикали, слон — по диагоналям). Проверить, есть ли угроза королю и если есть, то от кого именно. Координаты короля, ладьи и слона вводить целыми числами.

### 1.2.2 Теоритические сведения

В процессе написания кода программы в файле

`search_threat.c`

был использован оператор ветвления `if`. Также были задействованы функции ввода/вывода `puts()`, `scanf()` и математическая функция `abs()`.

### 1.2.3 Проектирование

Среда разработки: QT Creator 3.5.1 (opensource)

Компилятор: GCC

В программе разделены взаимодействие с пользователем и бизнес-логика.

`shahmati_ugroza.c`

считывает из консоли введенные пользователем координаты короля, ладьи и слона. Полученные данные передаются функции

```
int search_threat()
```

, где происходит поиск угроз королю со стороны слона и ладьи. Найденные угрозы выводятся в консоль.

Для контроля исправности программы были созданы модульные тесты, расположенные в файле



tst\_testtest.cpp

.

### 1.2.4 Тестовый план и результаты тестирования

При проведении модульного тестирования программа вызывает функцию

```
void search_threat_test()
```

. Код тестируется с помощью макроса QCOMPARE(), который сравнивает значение, полученное в функции

```
int search_threat()
```

, и ожидаемый результат. При несовпадении данных, выполнение программы завершается сообщением о непройденном тесте.

### 1.2.5 Выводы

Выполнение задания позволило освоить процесс разделения программы на взаимодействие с пользователем и бизнес-логику, а также приобрести некоторые навыки отладки кода и исправления ошибок, выводимых компилятором.

## Листинги

```
1 #include <stdio.h>
2 #include "file_clear.h"
3 #include "search_sine.h"
4 #include "shahmati_ugroza.h"
5 #include "trancfer_to_meters.h"
6 #include "palindromes_read.h"
7 #include "main.h"
8
9 int main()
10 {
11     puts ("1. transfer to meters");
12     puts ("2. search chess threat");
13     puts ("3. search sine");
14     puts ("4. search clear in array");
15     puts ("5. search palindromes");
16
17     int choice;
```

```

18     scanf ("%d", &choice);
19     switch (choice) {
20     case 1:
21         puts ("transfer to meters");
22
23         trancfer();
24         break;
25
26
27
28
29     case 2:
30         puts ("search chess threat");
31
32         search_threat_reading();
33         break;
34
35
36
37
38     case 3:
39         puts ("search sine");
40
41         search_sine();
42         break;
43
44
45
46
47     case 4:
48         puts ("search clear in array");
49
50         file_clear();
51         break;
52
53
54
55
56     case 5:
57         puts ("search palindromes");
58
59         file_open();
60         break;
61
62
63
64     }
65     return (0);
66

```

67 }

```
1 #include <stdio.h>
2 #include "shahmati_ugroza.h"
3 #include "search_threat.h"
4
5 void search_threat_reading()
6 {
7     int xking, yking, xladya, yladya, xslon, yslon;
8     int result;
9
10
11     puts("Введите через пробел координаты короля, ладьи и сло
        на");
12     scanf("%d %d %d %d %d %d", &xking, &yking, &xladya, &
        yladya, &xslon, &yslon);
13     result = search_threat(xking, yking, xladya, yladya,
        xslon, yslon);
14     if (result == 1) printf ("threat from elephant");
15     if (result == 2) printf ("threat from rook");
16     if (result == 3) printf ("threat from elephant");
17 }
```

```
1 #ifndef SHAHMATI_UGROZA_H
2 #define SHAHMATI_UGROZA
3
4 void search_threat_reading();
5
6 #endif // SHAHMATI_UGROZA_H
```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include "search_threat.h"
4
5 int search_threat(int xking, int yking, int xladya, int
    yladya, int xslon, int yslon)
6 {
7     int threat;
8
9     if (xking==xladya) threat = 1;
10    if (yking==yladya) threat = 2;
11    if ((abs(xking-xslon))==abs(yking-yslon)) threat = 3;
12
13    return (threat);
14
15 }
```

```
1 #ifndef SEARCH_THREAT_H
```

```

2 #define SEARCH_THREAT_H
3
4 #ifdef __cplusplus
5 extern "C"{
6 #endif
7
8 int search_threat(int, int, int, int, int, int);
9 #ifdef __cplusplus
10 }
11 #endif
12 #endif // SEARCH_THREAT_H

```

```

1 #include <QString>
2 #include <QtTest>
3 #include "search_clear.h"
4 #include "search_threat.h"
5 #include "sinx.h"
6 #include "trancfer_to_meter.h"
7
8
9 class TestTest : public QObject
10 {
11     Q_OBJECT
12
13
14 private Q_SLOTS:
15     void testCase1();
16     void search_sine_test();
17     void search_threat_test();
18     void trancfer_test();
19 };
20
21
22 void TestTestCase1()
23 {
24     QVERIFY2(true, "Failure");
25 }
26 void search_sine_test()
27 {
28     QCOMPARE(sinx(0.5, 0.001), 0.5);
29 }
30
31 void search_threat_test()
32 {
33     QCOMPARE(search_threat(2, 4, 1, 2, 3, 3), "threat from
34         elephant");
35 }
36 void trancfer_test()

```

```
37 {  
38 QCOMPARE(trancfer_to_meters(8, 2, 11.4), 19);  
39 }  
40  
41 QTEST_APPLESS_MAIN(TestTest)  
42  
43 #include "tst_testtest.moc"
```

# Глава 2

## ЦИКЛЫ

### 2.1 Задание 3

#### 2.1.1 Задание

Используя ряд  $\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots$ , рассчитать значение синуса  $x$  с заданной точностью (например,  $10^{-3}$ ). Заданную точность считать достигнутой, если очередной элемент ряда меньше заданной точности.

#### 2.1.2 Теоритические сведения

В процессе написания кода программы был использован цикл `do while` для поочередного определения слагаемых. Цикл прекращается при достижении указанной точности. Кроме того, были применены функции ввода/вывода `puts()`, `scanf()` и `printf()`.

#### 2.1.3 Проектирование

Среда разработки: QT Creator 3.5.1 (opensource)

Компилятор: GCC

В программе разделены взаимодействие с пользователем и бизнес-логика.

`search_sine.c`

считывает из консоли введенные пользователем значение  $x$  и точность. Полученные данные передаются функции

`double sinx()`

, где высчитывается значение  $\sin(x)$ .

Для контроля исправности программы были созданы модульные тесты, расположенные в файле

tst\_testtest.cpp

.

### 2.1.4 Тестовый план и результаты тестирования

При проведении модульного тестирования программа вызывает функцию

```
void search_sine_test()
```

. Код тестируется с помощью макроса QCOMPARE(), который сравнивает значение, полученное в функции

```
double sinx()
```

, и ожидаемый результат. При несовпадении данных, выполнение программы завершается сообщением о непройденном тесте.

### 2.1.5 Выводы

Выполнение задания позволило освоить процесс создания цикла do while, а также выяснить пути исправления некоторых ошибок, выводимых компилятором.

## Листинги

```
1 #include <stdio.h>
2 #include "file_clear.h"
3 #include "search_sine.h"
4 #include "shahmati_ugroza.h"
5 #include "trancfer_to_meters.h"
6 #include "palindromes_read.h"
7 #include "main.h"
8
9 int main()
10 {
11     puts ("1. transfer to meters");
12     puts ("2. search chess threat");
13     puts ("3. search sine");
14     puts ("4. search clear in array");
```

```

15     puts ("5. search palindromes");
16
17     int choice;
18     scanf ("%d", &choice);
19     switch (choice) {
20     case 1:
21         puts ("transfer to meters");
22
23         trancfer();
24         break;
25
26
27
28
29     case 2:
30         puts ("search chess threat");
31
32         search_threat_reading();
33         break;
34
35
36
37
38     case 3:
39         puts ("search sine");
40
41         search_sine();
42         break;
43
44
45
46
47     case 4:
48         puts ("search clear in array");
49
50         file_clear();
51         break;
52
53
54
55
56     case 5:
57         puts ("search palindromes");
58
59         file_open();
60         break;
61
62
63

```



```

64     }
65     return (0);
66
67 }

```

```

1  #include <stdio.h>
2  #include "search_sine.h"
3  #include "sinx.h"
4
5
6
7  void search_sine ()
8  {
9      double x, exact = 0;
10     double result;
11     puts("Введите через пробел значения x и точности");
12     scanf("%lf %lf", &x, &exact);
13     result = sinx(x, exact);
14     printf("sinx = %G", result);
15
16 }

```

```

1  #ifndef SEARCH_SINE_H
2  #define SEARCH_SINE_H
3
4  void search_sine();
5
6  #endif // SEARCH_SINE_H

```

```

1  #include <stdio.h>
2  #include "sinx.h"
3
4  double sinx (double x, double exact)
5  {
6      double summand, sin, i;
7      summand=x;
8      sin=summand;
9      i=0;
10     do {
11         i++;
12         summand*=((-1)*x*x)/(2*i*(2*i+1));
13         sin+=summand;
14     }
15     while (summand>exact);
16     return(sin);
17
18 }

```

```

1 #ifndef SINX_H
2 #define SINX_H
3
4 #ifdef __cplusplus
5 extern "C"{
6 #endif
7
8 double sinx(double, double);
9
10
11 #ifdef __cplusplus
12 }
13 #endif
14 #endif // SINX_H

```

```

1 #include <QString>
2 #include <QtTest>
3 #include "search_clear.h"
4 #include "search_threat.h"
5 #include "sinx.h"
6 #include "trancfer_to_meter.h"
7
8
9 class TestTest : public QObject
10 {
11     Q_OBJECT
12
13
14 private Q_SLOTS:
15     void testCase1();
16     void search_sine_test();
17     void search_threat_test();
18     void trancfer_test();
19 };
20
21
22 void TestTestCase1()
23 {
24     QVERIFY2(true, "Failure");
25 }
26 void search_sine_test()
27 {
28     QCOMPARE(sinx(0.5, 0.001), 0.5);
29 }
30
31 void search_threat_test()
32 {

```

```
33 QCOMPARE(search_threat(2, 4, 1, 2, 3, 3), "threat from  
    elephant");  
34 }  
35  
36 void tranfer_test()  
37 {  
38     QCOMPARE(tranfer_to_meters(8, 2, 11.4), 19);  
39 }  
40  
41 QTEST_APPLESS_MAIN(TestTest)  
42  
43 #include "tst_testtest.moc"
```

# Глава 3

## Массивы

### 3.1 Задание 4

#### 3.1.1 Задание

Квадрат  $n \times n$  состоит из прозрачных и непрозрачных маленьких квадратов. Имеется ли хотя бы один просвет по каждому из двух измерений? Вывести координаты каждого просвета.

#### 3.1.2 Теоритические сведения

В процессе написания кода программы для работы с файлом и для динамического выделения памяти были использованы указатели. Кроме того, были применены циклы `while` и `for`, оператор ветвления `if`. Также использовались функции для работы с файлом `fopen()` и `fclose()`, функции выделения и освобождения памяти `malloc()` и `free()`, функция чтения данных из файла `fscanf()`, функция вывода `printf()`.

#### 3.1.3 Проектирование

Среда разработки: QT Creator 3.5.1 (opensource)

Компилятор: GCC

В программе разделены взаимодействие с пользователем и бизнес-логика. Функция

```
void file_clear()
```

считывает из файла данные и записывает их в двумерный массив. Полученные данные передаются функции

```
void clear()
```

, где происходит поиск пустых строк и столбцов массива.

### 3.1.4 Выводы

Выполнение задания позволило освоить процесс динамического выделения памяти и научиться работать с массивами.

### Листинги

```
1 #include <stdio.h>
2 #include "file_clear.h"
3 #include "search_sine.h"
4 #include "shahmati_ugroza.h"
5 #include "trancfer_to_meters.h"
6 #include "palindromes_read.h"
7 #include "main.h"
8
9 int main()
10 {
11     puts ("1. transfer to meters");
12     puts ("2. search chess threat");
13     puts ("3. search sine");
14     puts ("4. search clear in array");
15     puts ("5. search palindromes");
16
17     int choice;
18     scanf ("%d", &choice);
19     switch (choice) {
20     case 1:
21         puts ("transfer to meters");
22
23         trancfer();
24         break;
25
26
27
28
29     case 2:
30         puts ("search chess threat");
31
32         search_threat_reading();
33         break;
34
35
36
```

```

37
38     case 3:
39         puts ("search sine");
40
41         search_sine();
42         break;
43
44
45
46
47     case 4:
48         puts ("search clear in array");
49
50         file_clear();
51         break;
52
53
54
55
56     case 5:
57         puts ("search palindromes");
58
59         file_open();
60         break;
61
62
63
64     }
65     return (0);
66
67 }

```

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include "file_clear.h"
4  void file_clear()
5  {
6      FILE *fp;
7      fp=fopen("array.txt","a");
8      int i,j,n = 0;
9      fscanf(fp, "%d \n", &n);
10     int *matrix;
11     matrix=(int*) malloc(n*n*sizeof(int));
12     for (i=0; i<n; i++)
13         for (j=0; j<n; j++)
14             {
15                 fscanf(fp, "%d", (matrix+i*n+j));
16             }
17

```

```

18     fclose (fp);
19     void clear(int n, int* matrix);
20
21 }

```

```

1 #ifndef FILE_CLEAR_H
2 #define FILE_CLEAR_H
3
4 void file_clear();
5
6 #endif // FILE_CLEAR_H

```

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include "search_clear.h"
4
5 void clear(int n, int* matrix)
6 {
7     int i,j;
8     int sum = *matrix;
9
10    for (i=0; i<n; i++)
11    {
12        sum = *(matrix+i*n);
13        j = 1;
14        while (j<n)
15        do
16        {sum = sum + *(matrix+i*n+j);}
17        if (sum == 0) printf("Просвет в %d строке", i);
18        sum = 0;
19    }
20
21    for (j=0; j<n; j++)
22    {
23        sum = *(matrix+j);
24        i = 1;
25        while (i<n)
26        do
27        {sum = sum + *(matrix+i*n+j);}
28        if (sum == 0) printf("Просвет в %d столбце", j);
29        sum = 0;
30    }
31    free (matrix);
32 }

```

```

1 #ifndef SEARCH_CLEAR_H
2 #define SEARCH_CLEAR_H
3

```

```
4|
5| #ifdef __cplusplus
6| extern "C"{
7| #endif
8|
9| void clear(int, int*);
10| #ifdef __cplusplus
11| }
12| #endif
13| #endif // SEARCH_CLEAR_H
```



# Глава 4

## Строки

### 4.1 Задание 5

#### 4.1.1 Задание

Имеется большой словарь русских слов. Найти в нем слова-палиндромы, одинаково читающиеся как слева направо, так и справа налево, например, АННА, ШАЛАШ.

#### 4.1.2 Теоритические сведения

В процессе написания кода программы была использована функция поиска длины строки `strlen()`, функции для работы с файлами `fopen()` и `fclose()`. Также были применены циклы `while` и `for`, оператор ветвления `if`.

#### 4.1.3 Проектирование

Среда разработки: QT Creator 3.5.1 (opensource)

Компилятор: GCC

В программе разделены взаимодействие с пользователем и бизнес-логика.

```
palindromes_read.c
```

считывает из файла слова и передает их функции

```
char search_palindromes()
```

, где определяется, является слово палиндромом или нет.

#### 4.1.4 Выводы

Выполнение задания позволило освоить некоторые функции заголовочного файла string.h.

#### Листинги

```
1 #include <stdio.h>
2 #include "file_clear.h"
3 #include "search_sine.h"
4 #include "shahmati_ugroza.h"
5 #include "trancfer_to_meters.h"
6 #include "palindromes_read.h"
7 #include "main.h"
8
9 int main()
10 {
11     puts ("1. transfer to meters");
12     puts ("2. search chess threat");
13     puts ("3. search sine");
14     puts ("4. search clear in array");
15     puts ("5. search palindromes");
16
17     int choice;
18     scanf ("%d", &choice);
19     switch (choice) {
20     case 1:
21         puts ("transfer to meters");
22
23         trancfer();
24         break;
25
26
27
28
29     case 2:
30         puts ("search chess threat");
31
32         search_threat_reading();
33         break;
34
35
36
37
38     case 3:
39         puts ("search sine");
40
41         search_sine();
```

```

42         break;
43
44
45
46
47     case 4:
48         puts ("search clear in array");
49
50         file_clear();
51         break;
52
53
54
55
56     case 5:
57         puts ("search palindromes");
58
59         file_open();
60         break;
61
62
63
64     }
65     return (0);
66
67 }

```

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include "palindromes_read.h"
5  #include "palindromes_search.h"
6
7  int file_open()
8  {
9      FILE *file;
10     file = fopen("palindromes.txt", "r");
11     char *string;
12     char palindrom[30];
13     int result;
14     if (file==0)
15     {
16         printf("Ошибка \n"); return -1;}
17     else printf ("Найденные палиндромы:\n");
18
19
20     while (1)
21     do
22     { string = fgets(palindrom, sizeof(palindrom), file);

```

```

23     if (string == NULL) {
24         if (feof(file) != 0)
25             {printf ("Конец файла \n");
26              break;}
27         else
28             {printf("Ошибка чтения \n");
29              break;}
30     }
31
32
33     else result = search_palindromes(string);
34     if (result == 0) printf ("%s \n", string);
35
36 }
37 fclose(file);
38 return 0;
39
40 }

```

```

1 #ifndef PALINDROMES_READ_H
2 #define PALINDROMES_READ_H
3
4 int file_open();
5
6 #endif // PALINDROMES_READ_H

```

```

1 #include <stdio.h>
2 #include <string.h>
3 #include <stdlib.h>
4 #include <palindromes_search.h>
5
6 char search_palindromes(char* string)
7 {
8     int i;
9     int lenght;
10    int count = 0;
11    lenght = strlen(string);
12    for (i=0; i<=lenght/2; ++i)
13    {
14        if (string[i] != string[lenght-i])
15            ++count;
16
17    }
18
19    return (count);
20
21 }

```

```
1 #ifndef PALINDROMES_SEARCH_H
2 #define PALINDROMES_SEARCH_H
3
4 char search_palindromes(char*);
5
6 #endif // PALINDROMES_SEARCH_H
```