Программирование

Корсков А. В.

7 ноября 2015 г.

Глава 1

Основные конструкции языка

1.1 Задание 1

1.1.1 Задание

В морской миле 2000 ярдов или 6000 футов. Задано некоторое расстояние в футах, например, 9139 футов. Вывести то же расстояние в милях, ярдах и футах, например, 9139 = 1 миля 1046 ярдов и 1 фут.

1.1.2 Теоретические сведения

Для реализации данного алгоритмы были использованы функции библеотеки stdio.h для ввода и вывода информации. Также для передачи значений из функции мы использовали указатели. Морская миля — единица измерения расстояния, применяемая в мореплавании и авиации. Ярд (англ. yard) — британская и американская единица измерения расстояния. Фут — единица измерения длины в английской системе мер. Мы знаем, что в 1 миле содержится 2000 ярдов или 6000 футов. А в 1 ярде содержится 3 фута.

1.1.3 Проектирование

В функции main вызывается функция convert_UI. Взаимодействие с пользователем реализовано в фунции convert_UI. Она считывает ввёдные пользователем значения из консоли, вызывает функцию convert, а после выполнения функции convert, выводит результат в консоль. Для реализации алгоритма мы выделили функцию convert. Функция принимает три указателя на тип int. Первый соответствует колличеству футов,

второй соответствует колличеству ярдов, третий соответствует колличеству миль. Сначала в функции вычисляется колличество миль. Далее находится колличество футов без миль. На следующем шаге вычисляется колличество ярдов. И на последнем шаге находится колличество футов. Функция ничего не возвращает, так как для передачи значения мы пользуемся указателями.

1.1.4 Описание тестового стенда и методики тестирования

Для решения задачи использовалась вертуальная машина с операционной системой Debian, в которой установлен QtCreator. Пользователь может сам ввести его данные или воспользоваться автоматическими тестами, запустив их отдельно.

1.1.5 Тестовый план и результаты тестирования

В программе предусмотрены тесты, которые проверяют правильность исполнения программы. В тесте вызывается функция, ей передаётся колличество футов, равное 9139. Далее каждая из трёх переменных проверяется на соответствие с ожидаемым правильным результатом. При вызове теста ошибок не обнаружено.

1.1.6 Выводы

В ходе работы мы научились структурировать проект, разбивая задачи на мелкие подзадачи. Получили опыт создания тестов. Также мы научились работать с системой контроля версий и приобрели навыки работы с указателями.

Листинги

```
#include <stdio.h>
#include "convert_UI.h"
#include "check_UI.h"
#include "removal_UI.h"
#include "square_UI.h"



/// Какая ужасно длинная функция main
/// нужно обязательно разделить на функции поменьше
```

```c
int main(void)
{
    puts("1. Перевод футов в мили, ярды и футы.");
    puts("2. Проверка допустимости треугольника.");
    puts("3. Удаление из числа нечётных чисел.");
    puts("4. Проверить является ли двумерный массив латинским
        квадратом.");
    int choice;
    scanf("%d", &choice);
    switch(choice){
        case 1:
            convert_UI();
        break;
        case 2:
            check_UI();
        break;
        case 3:
            removal_UI();
        break;
        case 4:
            square_UI();
    }
    return 0;
}
```

```c
#include <stdio.h>
#include "convert.h"

void convert_UI(){
    puts("Введите количество футов.");
    int ft,yd,m;
    scanf("%d", &ft);
    convert(&ft, &yd, &m);
    printf("Колличество миль: %d, колличество ярдов: %d колли
       чество футов: %d \n", m, yd, ft);
}
```

```c
#include <stdio.h>
#include"convert.h"

/// Давайте в качестве усложнения задания заведем еще структу
    ру для миль, ярдов, футов
/// то есть будет
/// us_measurment_system convert(foot){
/// ...
/// }
void convert(int* ft, int* yd, int* m){
    *m = *ft / 6000;
    *ft = *ft - *m * 6000;
```

```
12      *yd = *ft / 3;
13      *ft = *ft - *yd * 3;
14  }
```

1.2 Задание 2

1.2.1 Задание

Заданы три целых числа: a, b, c. Определить, могут ли они быть длинами сторон треугольника, и если да, определить, является ли он равнобедренным либо равносторонним.

1.2.2 Теоритические сведения

Треугольник существует только тогда, когда сумма любых двух его сторон больше третьей. У равнобедренного треугольника две стороны равны. У равностороннего все стороны равны. Для реализации данного алгоритмы были использованы функции библеотеки stdio.h для ввода и вывода информации. Также использовалась конструкция if...else.

1.2.3 Проектирование

В функции main вызывается функция check_UI. Взаимодействие с пользователем реализовано в фунции check_UI. Она считывает ввёдные пользователем значения из консоли, вызывает функцию check, а после выполнения функции check, выводит результат в консоль. Для реализации алгоритма мы выделили функцию check. Функция принимает три переменные типа int. Каждая из этих трёх переменных задаёт одну из сторон треугольника. Сначала входные данные проверяются на корректность. Далее проверяем возможен ли данный треугольник или нет. Затем смотрим является ли данный треугольник равнобедренним. Если это условие выполняется, проверяем будет ли данный треугольник равносторонним.

1.2.4 Описание тестового стенда и методики тестирования

Для решения задачи использовалась вертуальная машина с операционной системой Debian, в которой установлен QtCreator. Пользователь может сам ввести его данные или воспользоваться автоматическими тестами, запустив их отдельно.

1.2.5 Тестовый план и результаты тестирования

В программе предусмотрены тесты, которые проверяют правильность исполнения программы. В тесте вызывается функция, ей передаётся три стороны треугольника, которые все равны 3. Далее полученное число сверяется с ожидаемым правильным ответом. При вызове теста ошибок не обнаружено.

1.2.6 Выводы

В ходе работы мы научились работать с конструкцией if...else. Получили опыт создания вложенных операторов if...else.

```c
#include <stdio.h>
#include "check.h"

void check_UI(){
    puts("Введите три стороны треугольника.");
    int a,b,c,result_check;
    scanf("%d%d%d", &a, &b, &c);
    result_check = check(a,b,c);
    switch(result_check){
        case 0:
            printf("Данные некорректны.\n");
        break;
        case 1:
            printf("Данный треугольник возможен.\n");
        break;
        case 2:
            printf("Данный треугольник является равнобедренны
               м.\n");
        break;
        case 3:
            printf("Данный треугольник является равносторонни
               м.\n");
        break;
        case 4:
            printf("Данный треугольник не возможен.\n");
    }
}
```

```c
#include <stdio.h>
#include "check.h"

/// Заведите енит для результатов, чтобы не просто числовые к
    оды
int check(int a, int b, int c){
    int resultat;
```

```
7       if (a <= 0 || b <= 0 || c <= 0)
8           /// Может быть сразу сделать return(0) ?
9           resultat = 0;
10      else{
11          if (a + b > c && a + c > b && c + b > a){
12              resultat = 1;
13              /// если a равно b, то и b равно a, здесть избыто
                    чные условия
14              if ((a == b && b == a) || (a == c && c == a) || (
                    b == c && c == b)){
15                  resultat = 2;
16                  if (a == b && b == c)
17                      resultat = 3;
18              }
19          }
20          else
21              resultat = 4;
22      }
23
24      /// Есть разные подходы, но  я бы  в этом случае не завод
            ила переменную для результата,
25      /// а сразу бы выходила из функции, как только решила, чт
            о с этим треугольником
26      return resultat;
27  }
```

Глава 2

Циклы

2.1 Задание 1

2.1.1 Задание

2.1.2 Теоритические сведения

2.1.3 Проектирование

2.1.4 Описание тестового стенда и методики тестирования

2.1.5 Тестовый план и результаты тестирования

2.1.6 Выводы