Laboratório 6

Funções / Modularidade.

Oficina de Programação em C (ICP037)

Prof. Ronald Souza
IC/UFRJ

Objetivo
Praticar os conceitos de programação vistos na Aula 6.

Relembrando funções

Um dos exemplos de uso de funções visto em aula foi como segue:

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

//Assinatura da função 'dist':
float dist(float x1, float y1, float x2, float y2);

int main () {
     float x, y, a, b;
     scanf("%f %f %f %f", &x, &y, &a, &b);

     //Chamada da função 'dist':
     printf("Distancia = %0.2f\n", dist(x,y,a,b));
     return 0;
}

//Implementação da função 'dist':
float dist(float x1, float y1, float x2, float y2) {
     float dx, dy, distancia;
     dx = x2 - x1;
     dy = y2 - y1;
     distancia = sqrt(dx * dx + dy * dy);
     return distancia;
}
```

Atividade 1 - Modularidade de implementação

Crie 3 arquivos **dentro de um mesmo diretório**, com os seguintes nomes:

- main.c
- teste.c
- teste.h

Agora, execute os seguintes passos:

1) No arquivo **main.c**, escreva ou copie o código abaixo:

```
#include <stdio.h>
#include "teste.h"

int main() {
    int a = 3, b = 2, res;
    res = multiplica(a, b);
    printf("res = %d", res);
    return 0;
}
```

2) No arquivo **teste.h**, escreva ou copie a linha abaixo:

```
int multiplica(int a, int b);
```

3) No arquivo **teste.c**, escreva ou copie o código abaixo:

```
int multiplica(int a, int b){
    return a * b;
}
```

4) **Salve as modificações em todos os arquivos;**

5) Agora vamos fazer o programa rodar. Para compilar, abra um terminal e o aponte para o mesmo diretório onde se encontram os arquivos acima. Então, digite o comando abaixo:

```
gcc main.c teste.c -o main.out -Wall
```

6) Repare: **não** foi preciso ordenar a compilação do arquivo **teste.h;** Por que?

7) Rode o programa digitando “./main.out” (sem as aspas). **Se tudo deu certo, você rodou seu primeiro programa modularizado!**

Atividade 2 - Painel (prof. Adriano Cruz) - Zé quer montar um painel de LEDs contendo diversos dígitos, mas não sabe quantos LEDs exatamente ele deve comprar. A figura abaixo mostra quantos LEDs são utilizados para montar cada um dos algarismos:

Cada retângulo vermelho corresponde a um led. Por exemplo, para montar o algarismo 3 são necessários 5 LEDs. Considerando esta configuração, escreva um programa que ajude o Zé a descobrir a quantidade de LEDs necessária para montar um painel com o valor desejado.
Seu programa deverá conter 3 funções:
1) `main()`, onde será feita a coleta do número e a exibição do total de LEDs necessários;

2) `int totalLEDS(int n)`, que recebe o valor n coletado na main() e retorna o total de LEDS necessários para reproduzi-lo no painel.

3) `int ledsAlgarismo(int a)`, que recebe da função 'totalLEDS' **um único dígito** (isto é, um algarismo entre 0 e 9), e retorna o total de LEDs necessários para aquele único dígito.

→ *A modularização por arquivos é **opcional** nesta atividade.*

São exemplos de entrada e saída:

ENTRADA	**SAÍDA**
11	4
4	4
76	9
115380	27
2819311	29

Atividade 3 - Tabuada - Escreva um programa que lê do teclado um número entre 1 e 10 e o utilize como parâmetro de entrada da função "**tabuada**".
Sua função "tabuada" deverá imprimir em ordem crescente todas as multiplicações do número de entrada por valores entre 1 e 10 e **não retornar nenhum valor** para a main().

→ *A modularização por arquivos é **opcional** nesta atividade.*

Por exemplo, a *chamada* **tabuada(2);** deverá produzir a seguinte saída:

2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
2 x 10 = 20