Linguagem de Programação

2ª Lista de Exercícios – Funções

1. Escreva um protótipo das funções descritas abaixo:
   a) `numzeros()` espera um `int` como argumento e imprime o número de zeros de seu argumento;
   b) `max()` espera dois argumentos do tipo `int` e devolve um `int`;
   c) `n_to_char()` espera um argumento do tipo `int` e devolve um `char`;
   d) `digitos()` espera um argumento do tipo `int` e outro do tipo `double` e devolve um `int`;
   e) `random()` não espera nenhum argumento e devolve um `int`.

2. Encontre o erro em cada uma das seguintes funções e explique por que está errado:

a)
```
int soma(int x, y) {
      int resultado;
      resultado = x + y;
      return resultado;
  }
```

b)
```
void foo(float a) {
      float a;
      printf("%f\n", a);
  }
```

c)
```
int quadrado(int x);
  {
     return x * x;
  }
```

d)
```
void salame(num) {
     int num, i;
     for (i = 1; i <= num; i++)
         printf("Meu salame!\n");
  }
```

3. Criar um programa que dados 3 números inteiros, utilize uma função do tipo `void` que imprima o maior desses números.

4. Escreva uma função de protótipo `void retangulo(int a, int c);` que desenha no vídeo um retângulo formado por asteriscos (*) com *a* linhas de altura e *c* colunas de comprimento. Por exemplo, se for feita a seguinte chamada a função: `retangulo(5, 10);`

   A função deve desenhar no vídeo o seguinte retângulo:

```
**********
**********
**********
**********
**********
```

5. Faça uma função que recebe por parâmetro o raio de uma esfera e calcula e devolve o seu volume ($4\pi R^3/3$).

6. Escreva uma função que recebe como parâmetros dois inteiros positivos, *a* e *b*, e devolve o MDC (Máximo Divisor Comum) de *a* e *b*, calculado por meio do algoritmo de Euclides. Exemplo:

	1	1	1	2	
24	15	9	6	**3**	= *mdc(24,15)*
9	6	3	0		

7. Faça uma função que verifique se um valor é perfeito ou não. Um valor é dito perfeito quando ele é igual a soma dos seus divisores excetuando ele próprio. (Ex: 6 é perfeito, 6 = 1 + 2 + 3, que são seus divisores). A função deve retornar um valor booleano.

8. Faça uma função que recebe, por parâmetro, a altura (*alt*) e o sexo de uma pessoa e retorna o seu peso ideal. Para homens, calcular o peso ideal usando a fórmula peso ideal = 72.7 x *alt* - 58 e ,para mulheres, peso ideal = 62.1 x *alt* - 44.7.

9. Escreva uma função para calcular o fatorial de um número natural.

10. Considere a função para calcular o fatorial de um número natural implementado na questão anterior e escreva um programa que recebe dois números naturais (*n* e *k*) como parâmetros da função *main()* e calcula e imprime:

    a) O número de permutação P_n: $P_n = n!$

    b) O número de arranjos $A_{n,k}$: $A_{n,k} = \frac{n!}{(n-k)!}$

    c) O número de combinações $C_{n,k}$: $C_{n,k} = \frac{n!}{k! * (n-k)!}$

11. Criar uma função para calcular x^y, dados como parâmetros *x* (um número real) e *y* (um número natural). Restrição: não é permitido utilizar a função `pow`.

12. Escreva uma função de protótipo `double hipotenusa(double x, double y);` que calcula e devolve o comprimento da hipotenusa de um triângulo retângulo cujos catetos são dados pelos parâmetros *x* e *y*. Lembre-se que $hipotenusa = \sqrt{x^2 + y^2}$. Dica: utilize a função da questão anterior para obter os quadrados dos catetos e a função `sqrt` para obter a raiz quadrada.

13. A função `floor`, definida no arquivo *math.h*, arredonda seu argumento (um número do tipo `double`) para o maior inteiro que não seja maior que esse argumento, na prática, isso significa devolver a parte inteira do argumento. Entretanto, o valor de retorno da função `floor` é um `double`. Crie uma função de protótipo `int arredondarParaInt(double n);` que arredonda seu parâmetro *n* para o inteiro mais próximo. Dica: some 0.5 a *n* e utilize a função `floor`. Escreva um programa que leia vários números e use a função `arredondarParaInt` para arredondar cada um desses números para o inteiro mais próximo.

14. Escreva uma função de protótipo `double arredondar(double n, int c);` que arredonda o valor de *n* para um número com precisão de *c* casas decimais. Por exemplo, `arredondar(5.78351,1)` devolve `5.8`, `arredondar(5.78351,2)` devolve `5.78`, `arredondar(5.78351,3)` devolve `5.784`. Dica: utilize a função *arredondarParaInt* passando seu argumento multiplicado por 10^c, e depois divida o valor de retorno da função por 10^c.

15. Escreva uma função de protótipo `void init_vetor(int a[], int n, int val);` que inicialize o vetor *a* com *n* elementos com o valor de *val*.

16. Escreva uma função que recebe uma string e um caractere como parâmetros e devolve a

posição da 1ª ocorrência do caractere na string. Caso o caractere não esteja contido na string, a função deve devolver -1.

17. Escreva uma função que recebe um vetor de strings com até 20 caracteres cada e o número de strings do vetor como parâmetros, e devolve verdadeiro se o vetor está em ordem alfabética (crescente), ou falso, caso contrário. A função deve ter o seguinte protótipo:

```
bool estaOrdenado(char vetor[][21], int n);
```

18. Escreva uma função que receba como parâmetro uma matriz quadrada de ordem *n* de inteiros e devolve verdadeiro se ela é uma matriz triangular superior, ou falso, caso contrário. *Matriz triangular superior é uma matriz onde todos os elementos de posições acima da diagonal principal são diferentes de 0 e todos os elementos demais elementos são iguais a 0.*

19. Escreva uma função que receba uma matriz *A* bidimensional de valores reais e um valor real *x*, e multiplique todos os elementos de *A* por *x*.

20. Escreva uma função que recebe um inteiro *m* e devolve `true` (verdadeiro) se *m* é primo ou `false` (falso), caso contrário.

21. Escreva um programa que receba um número inteiro não-negativo *n* e imprima os *n* primeiros números primos. Utilize os parâmetros da *main()* para receber o valor de *n* e a função da questão anterior.

22. Considere os grupos de uma ou mais diretivas seguidas pelos códigos que as utilizam abaixo. Qual é o código resultante em cada caso? É um código válido? Assuma que as variáveis foram declaradas.

a)
```
#define KPH 95  /* Km por hora */
dist = KPH * tempo;
```

b)
```
#define METROS 4
#define POD METROS + METROS
plort = METROS * POD;
```

c)
```
#define SEIS = 6;
num = SEIS;
```

d)
```
#define NEW(x) x + 5
y = NEW(y);
berg = NEW(berg) * lob;
est = NEW(berg) / NEW(y);
nilp = lob * NEW(-berg);
```

23. Corrija a definição no item d) da questão anterior para torná-la mais confiável.

24. Escreva um programa que use uma macro `MINIMUM2` para determinar o menor entre dois valores numéricos.

25. Escreva um programa que use uma macro `MINIMUM3` para determinar o menor de três valores numéricos recebidos via argumentos da função *main()*. A macro `MINIMUM3` deve usar a macro `MINIMUM2` definida um exercício anterior.