

Linguagem C

Funções

Funções em Linguagem “C”

- **O que são “Funções”?** (ou subprogramas ou subrotinas)
 - São trechos de código fonte agrupados sob um nome, que podem ser chamados sempre que for necessário executar uma determinada ação programada neste trecho;
- **Como usar funções?**
 - Atribui-se um nome à uma seqüência de comandos, e faz-se referência a este nome nos vários lugares do programa onde a seqüência em questão deveria ser repetida.

Funções em Linguagem “C”

- **Por que usar funções?**
 - Evita escrita repetida de código (uma certa seqüência de comandos deve ser repetida em vários lugares de um programa).
 - Economiza o tempo gasto com o trabalho de copiar estas seqüências;
 - Evitar a necessidade de mudar em múltiplos lugares caso deseje alterar o seu funcionamento;
 - Dividir grandes tarefas de computação em tarefas menores:
 - Facilita o gerenciamento de grandes sistemas e
 - Aumenta a confiabilidade dos mesmos.
- **Resumo: Principais motivações para uso das funções!**
 - Evitar repetição de código
 - Modularização

Funções em Linguagem “C”

- **Funções em “C”**

- Em “C”, todo programa é composto por funções;
- Já utilizamos muitas funções, mesmo sem saber que eram funções (printf, scanf, sqrt, e até a famosa função “main”...)

- **Formato de declaração de funções :**

tipo_de_retorno nome_da_função (tipo1 param1, tipo2 param2,..., tipoN paramN)

{

/ corpo da função */*

return valor_de_retorno;

} / fim da função */*

- **tipo_de_retorno** especifica o tipo do valor que será retornado para quem chamou a função (int, float, double, char, **void**).
- Se o tipo_de_retorno for **void** significa que se trata de uma função que se comporta como uma subrotina; ou seja, a função não necessita retornar nenhum valor (exemplo: printf)

Funções em Linguagem “C”

- **Exemplo:** Calcular o fatorial de um número n(sem função):

```
#include <stdio.h>
#include <stdlib.h>
int i,n,fatorial;
int main() {
    printf("Digite o número N:");
    scanf("%d", &n);
    fatorial = 1;
    for(i=2;i<=n;i++) {
        fatorial = fatorial * i;
    }
    if(n>=0) {
        printf("O fatorial de N = %d vale %d.\n", n, fatorial);
    } else {
        printf("Não existe fatorial de numero negativo\n");
    }
    system("PAUSE");
    return 0;
}
```

Funções em Linguagem “C”

- **Exemplo:** Calcular o fatorial de um número n (com função):

```
#include <stdio.h>
#include <stdlib.h>
int i,n,fatorial;
void fat() {
    fatorial = 1;
    for(i=2;i<=n;i++) {
        fatorial = fatorial * i;
    }
}
int main() {
    printf("Digite o numero N:");
    scanf("%d", &n);
    fat();
    if(n>=0) {
        printf("O fatorial de N = %d vale %d.\n", n, fatorial);
    } else {
        printf("Não existe fatorial de numero negativo\n");
    }
    system("PAUSE");
    return 0;
}
```

Funções em Linguagem “C”

- **Exemplo:** Calcular o fatorial de um número n (com função):

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

void fat() {
    fatorial = 1;
    for(i=2;i<=n;i++) {
        fatorial = fatorial * i;
    }
}

int main() {
    int i,n,fatorial;
    printf("Digite o numero N:");
    scanf("%d", &n);
    fat();
    if(n>=0) {
        printf("O fatorial de N = %d vale %d.\n", n, fatorial);
    } else {
        printf("Não existe fatorial de numero negativo\n");
    }
    system("PAUSE");
    return 0;
}
```

Funções em Linguagem “C”

- **Exemplo:** Calcular o fatorial de um número n (com função):

```
#include <stdio.h>
#include <stdlib.h>
void fat() {
    int i,n,fatorial;
    fatorial = 1;
    for(i=2;i<=n;i++) {
        fatorial = fatorial * i;
    }
}
int main() {
    int i,n,fatorial;
    printf("Digite o numero N:");
    scanf("%d", &n);
    fat();
    if(n>=0) {
        printf("O fatorial de N = %d vale %d.\n", n, fatorial);
    } else {
        printf("Não existe fatorial de numero negativo\n");
    }
    system("PAUSE");
    return 0;
}
```


Funções em Linguagem “C”

- Até agora, nenhuma vantagem.....mas e se a função “retornar o valor do fatorial”????

Funções em Linguagem “C”

- Com o retorno da função fat(), começa a ficar interessante....

```
#include <stdio.h>
#include <stdlib.h>
int i,n,fatorial;
int fat() {
    fatorial = 1;
    for(i=2;i<=n;i++) {
        fatorial = fatorial * i;
    }
    return(fatorial);
}
int main() {
    printf("Digite o numero N:");
    scanf("%d", &n);
    if(n>=0){printf("O fatorial de N = %d vale %d.\n", n, fat());}
    else    {printf("Não existe fatorial de numero negativo\n");}
    system("PAUSE");
    return 0;
}
```

Funções em Linguagem “C”

- E se, ao invés de utilizar sempre a variável “n” como “parâmetro”, se pudéssemos variar....

Funções em Linguagem “C”

- Função fat recebendo um número como parâmetro(por valor!)

```
#include <stdio.h>
#include <stdlib.h>
int i,n,fatorial;
int fat(int numero) {
    fatorial = 1;
    for(i=2;i<=numero;i++) {
        fatorial = fatorial * i;
    }
    return(fatorial);
}
int main() {
    printf("Digite o numero N:");
    scanf("%d", &n);
    if(n>=0){printf("O fatorial de N = %d vale %d.\n", n, fat(n));}
    else    {printf("Não existe fatorial de numero negativo\n");}
    system("PAUSE");
    return 0;
}
```

Funções em Linguagem “C”

- Agora o programa principal-função main- não precisa mais “conhecer” as variáveis i e fatorial. Portanto estas podem ser variáveis “locais” à função “fat”.

Funções em Linguagem “C”

- Variáveis “locais” às funções “fat” e “main”.

```
#include <stdio.h>
#include <stdlib.h>
int fat(int numero) {
    int i,fatorial;
    fatorial = 1;
    for(i=2;i<=numero;i++) {
        fatorial = fatorial * i;
    }
    return(fatorial);
}
int main() {
    int n;
    printf("Digite o numero N:");
    scanf("%d", &n);
    if(n>=0){printf("O fatorial de N = %d vale %d.\n", n, fat(n));}
    else    {printf("Não existe fatorial de numero negativo\n");}
    system("PAUSE");
    return 0;
}
```

Funções em Linguagem “C”

- Parece que ficou mais complicado????Então vamos fazer o seguinte programa:
 - Como calcular a combinação de N elementos P a P????
 - Simples: Basta calcular pela fórmula:

$$C_P^N = \frac{N!}{P!(N - P)!}$$

- Então, vamos lá!!!!
- Primeiro, sem funções!!!

Combinação sem Funções

```
#include <stdio.h>
#include <stdlib.h>
int N,P,i;
int Fat1, Fat2, Fat3;
int main()
{
    printf("Digite o numero N:");
    scanf("%d", &N);
    printf("Digite o numero P:");
    scanf("%d", &P);
    if(N<0 || P<0)
    {
        printf("Não existe combinação negativa.\n");
        system("PAUSE");
        return 0;
    }
    else
    { // Iniciou o else
        Fat1 = 1;
        for(i=2;i<=N;i++)
        { Fat1 = Fat1 * i; }

        Fat2 = 1;
        for(i=2;i<=P;i++)
        { Fat2 = Fat2 * i; }

        Fat3 = 1;
        for(i=2;i<=(N-P);i++)
        { Fat3 = Fat3 * i; }

        printf("A combinação de %d numeros %d a %d vale %d.\n", N,P,P, Fat1/(Fat2*Fat3));
        system("PAUSE");
        return 0;
    } // Acabou o else
} // Acabou a função main
```


Combinação com Funções

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

int fat(int numero)
{
    int i, fatorial;
    fatorial = 1;
    for(i=2; i<=numero; i++)
        { fatorial = fatorial * i; }
    return(fatorial);
}
```

```
int main()
{
    int N, P;
    printf("Digite o numero N:");
    scanf("%d", &N);
    printf("Digite o numero P:");
    scanf("%d", &P);
    if(N<0 || P<0)
    {
        printf("Não existe combinação negativa\n");
    }
    else
    {
        printf("A combinação de %d numeros %d a %d
        %d.\n", N, P, P, fat(N)/(fat(P)*fat(N-P)));
    }
    system("PAUSE");
    return 0;
}
```

vale

Aspectos Importantes sobre Funções em Linguagem “C”

- Funções permitem que cada “parte”(função) do programa possua suas próprias variáveis locais, evitando “confusões” em programas grandes (será que alguém já usou a variável “i“???)
- Usualmente existe um arquivo (.h) onde são definidos os “protótipos” das funções(os cabeçalhos). A partir daí é possível saber como utilizar as funções (quantos parâmetros as funções possuem, qual a sua ordem e quais os seus tipos).

Funções “Recursivas” em Linguagem “C”

```
#include<stdio.h>
#include<stdlib.h>
long int fatorial(int); /* prototipo da funcao fatorial */

int main(void) {
    int N; /* declaracao da variavel local N*/
    long int F; /* declaracao da variavel local F */
    printf("Entre com um valor inteiro: ");
    scanf("%d",&N);
    F = fatorial( N);
    if (F != -1) {
        printf("O fatorial de %d e´ %ld \n",N,F);
    } else {
        printf("Não existe fatorial de numero negativo \n");
    }
    system("pause");
    return(0);
}
```

```
long int fatorial(int N) {
    long int Fat;

    if(N==0 ) {
        return(1); /* definição de fatorial de 0*/
    } else {
        if(N>0) {
            Fat = N*fatorial(N-1);
            return(Fat);
        } else {
            return(-1); /* não há fatorial */
        }
    }
}
```

Funções e Vetores

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

```
#define NUM_ALUNOS 3
```

```
float media ( float notas[], int n ) {
    int i = 0;
    float m = 0.0;

    for ( i = 0; i < n; i++ ) {
        m = m + ( notas[i] / n );
    }
    return m;
}
```

```
int main( ) {
    float notas [NUM_ALUNOS];
    float media_turma;
    int i;

    for ( i = 0; i < NUM_ALUNOS; i++ ) {
        do {
            printf ("Digite a nota do %d o. aluno: ", i+1);
            scanf ("%f", &notas[i]);
        } while ( ( notas[i] < 0.0 ) || ( notas[i] > 10.0 ) );
    }
    media_turma = media ( notas, NUM_ALUNOS );

    printf ("A media da turma eh %.2f \n", media_turma);
    system ("Pause");
    return 0;
}
```

Funções em Linguagem “C”

- **Exercício 1:**

- Fazer um programa capaz de calcular a área de um quadrado, um retângulo ou um triângulo retângulo.
- O programa deve perguntar qual a figura geométrica, e então pedir para o usuário digitar os tamanhos dos lados.
- Um quadrado só tem um tamanho de lado, o retângulo tem dois, e o triângulo retângulo também tem dois lados(mais a hipotenusa, mas neste caso não é necessário digitar este valor).
- Após a digitação, o programa deve calcular a área e apresentar ao usuário.
- Utilizar uma função para a leitura dos lados (verificando se não é digitado um valor negativo para o lado), e uma função para o cálculo de cada área. As fórmulas são
 - $\text{Área_quadrado} = \text{lado} * \text{lado}$
 - $\text{Área_retângulo} = \text{lado1} * \text{lado2}$
 - $\text{Área_triângulo} = (\text{lado 1} * \text{lado 2})/2$

Funções em Linguagem “C”

- **Exercício 2:**
 - Criar um programa capaz de calcular o tempo entre dois horários quaisquer de um determinado dia.
 - O programa deve ler dois horários, compostos por três números inteiros, representando horas, minutos e segundos. O programa deve verificar se o horário é válido(horas entre 0 e 23, minutos entre 0 e 59, e segundos entre 00 e 59).
 - O programa deve ter uma função para calcular a quantidade de segundos em um horário, e outra função para calcular e imprimir a quantidade de horas, minutos e segundos em uma quantidade de segundos;

Funções em Linguagem “C”

- **Exercício 3:**
 - Criar um programa capaz de ler duas notas de cada um dos 10 alunos de uma turma, calculando a média geral da primeira e da segunda prova. Em seguida, informe quantos alunos tiraram notas acima da média, tanto na primeira como na segunda prova.

Funções em Linguagem “C”

- Exercício Desafio: Fazer um programa para ler 2 matrizes 2x2 e imprimir na tela as matrizes, e sua soma.

Digite os 4 valores da matriz:

Valor [0][0] = 1

Valor [0][1] = 2

Valor [1][0] = 3

Valor [1][1] = 4

Digite os 4 valores da matriz:

Valor [0][0] = 1

Valor [0][1] = 2

Valor [1][0] = 3

Valor [1][1] = 4

A primeira matriz eh:

1 2

3 4

A segunda matriz eh:

1 2

3 4

A soma das duas matrizes eh:

2 4

6 8