Algoritmos e Programação B

Ponteiros

- parte 2 –

Ponteiros e Funções

Sumário

- Funções: Passagem de parâmetros por Valor
- Funções: Passagem de parâmetros por Referência (ponteiro como argumento)
- Funções e Matrizes
- Ponteiros e Funções: Retornando Ponteiros

Funções:
Passagem de parâmetros por Valor
(revisão)

- Na linguagem C, todas as passagens de parâmetros são feitas por valor.
  - Isto significa que os parâmetros das funções, que recebem os valores, são **cópias** dos argumentos passados.
- Assim, as alterações feitas nos valores dos parâmetros dentro de uma função não alteram os valores dos argumentos que foram colocados na chamada da função.

Funções:
Passagem de parâmetros por Valor

(revisão)

- Exemplo de passagem de parâmetros por valor

```
#include<stdio.h>
float pot(float, float);
int main(void){
   float x, y;
   x = 2;
   y = 3;
   printf("%.2f elevado a %.2f = %.2f", x, y, pot(x, y));
}
float pot(float a, float b){
      int i;
      float e=1;
      for(i=0;i<b;i++) e = e*a;
      return e;
}
```

Ponteiros e Funções
Passagem de parâmetros por Referência

- Em C, pode-se passar valores por referência a uma função, passando um **ponteiro como argumento**:
  - Isso faz com que o endereço do argumento seja passado para a função e, assim, pode-se alterar o valor do argumento fora da função.
  - Ponteiros são passados para funções como qualquer outra variável. Para isto, é necessário declarar os parâmetros como variáveis do tipo ponteiro.

Ponteiros e Funções
Passagem de parâmetros por Referência

- Exemplo de passagem de parâmetros por referência

```
#include<stdio.h>
float pot(float *, float);
int main(void){
   float x, y;
   x = 2;
   y = 3;
   printf("%.2f elevado a %.2f = %.2f", x, y, pot(&x, y));
}
float pot(float *a, float b){
      int i;
      float e=1;
      for(i=0;i<b;i++)  e = e*(*a);
      return e;
}
```

Ponteiros e Funções
Passagem de parâmetros por Referência

```
#include<stdio.h>

void f1(float *a, int b);

int main(void){
    float x=10;
    int b=5;
    printf("Na main, antes de f1, x = %f e b = %d\n", x, b);
    f1(&x, b);
    printf("Na main, depois de f1, x = %f e b = %d\n\n", x, b);
    system("pause");
    return 1;
}

void f1(float *a, int b){
     b = 30;
     *a = 20;
     printf("Em f1, *a = %f e b = %d\n", *a, b);
}
```

Ponteiros e Funções
Passagem de parâmetros por Referência

- Os ponteiros que são parâmetros de funções são variáveis que contém uma cópia do endereço contido no ponteiro que é o argumento na chamada da função.

- LEMBRE-SE:
  - Se há um ponteiro apontando para uma variável (p = &x), ao ser alterado o conteúdo da memória apontada pelo ponteiro (*p = 5), automaticamente o valor da variável x é alterado (x passa a valer 5 também).
  - ...

Ponteiros e Funções
Passagem de parâmetros por Referência

- **... e ainda lembre-se:**
  - Se há dois ponteiros contendo um mesmo endereço, como no caso de um ser a cópia do outro (p1 = p2), ao ser alterado o conteúdo apontado por um deles (*p1 = 5), automaticamente é alterado o conteúdo do outro (*p2 passa a ser 5 também), pois ambos apontam para o mesmo endereço.

  - Isto porque o ponteiro local da função é uma cópia do ponteiro que foi passado como argumento, de forma que ambos apontam para o mesmo local da memória. Então, se dentro da função é alterado o conteúdo apontado pelo ponteiro local, também é alterado o conteúdo apontado pelo ponteiro da função chamadora.

Ponteiros e Funções
Passagem de parâmetros por Referência

- Exemplo de passagem de parâmetros por referência
- **Qual o problema no código abaixo?**

```
#include<stdio.h>

float pot(float *, float);

int main(void){
   float x, y;
   x = 2; y = 3;
   printf("%.2f elevado a %.2f = %.2f", x, y, pot(&x, y));
}

float pot(float *a, float b){
      int i;
      float e=1;
      for(i=0;i<b;i++)  e = e*(*a);
      *a = e;
      return e;
}
```

Funções e matrizes

Funções e Matrizes

- Quando é necessário passar para uma função matrizes (vetor, matriz ou string), estas são sempre passadas por **referência**.

- Exemplo (string):

```
#include<stdio.h>

void imprime_letraMaiuscula(char *string);

void main(void){
     char s[80];
     gets(s);
     imprime_letraMaiuscula(s);
}

void imprime_letraMaiuscula(char *string){
     int i;
     for(i=0;string[i];i++){
        string[i]=toupper(string[i]);
        putchar(string[i]);
     }
}
```

Funções e Matrizes

- Em C, a referência à matriz (matriz, vetor ou string) é feita apenas pela posição do primeiro elemento (é um ponteiro).

- Assim, quando passamos esta posição, estamos nos referindo ainda ao dado original. Por passar apenas a posição inicial, não tem importância também o tamanho declarado para o vetor na declaração de parâmetros.

- **Importante:** a passagem de **um** elemento da matriz ou vetor (vetor[i], por exemplo) é por valor!

Exemplo 1: Funções e Vetores

```
#include<stdio.h>

void f1(float vet[]);
void f2(float *v);
int main(void){
    float x[2]={3, 4};
    int i;
    printf("Na main, antes das funcoes:\n x[0] = %f, x[1] = %f\n", x[0], x[1]);
    f1(x);
    printf("Na main, apos f1:\n x[0] = %f, x[1] = %f\n", x[0], x[1]);
    f2(x);
    printf("Na main, apos f2:\n x[0] = %f, x[1] = %f\n", x[0], x[1]);
    system("pause");
    return 1;
}

void f1(float vet[]){
     vet[0]=10;
     vet[1]=20;
     printf("Em f1:\n vet[0] = %f, vet[1] = %f\n", vet[0], vet[1]);
}

void f2(float *v){
     v[0]=1;
     v[1]=2;
     printf("Em f2:\n v[0] = %f, v[1] = %f\n", v[0], v[1]);
}
```

Exemplo 2: Funções e Matrizes

```
#include<stdio.h>

void f1(float mat[][]);
void mostraMatriz(float mat[][]);

int main(void){
    float x[2][2];
    int i, j;
    printf("Informe os elementos da
matriz 2x2:\n");
    for(i=0;i<2;i++)
       for(j=0;j<2;j++)
          scanf("%f", &x[i][j]);
    printf("Matriz digitada:\n");
    mostraMatriz(x);
    f1(x);
    printf("Na main, a matriz apos a
execucao de f1...\n");
    mostraMatriz(x);
    system("pause");
    return 1;
}
```

```
void mostraMatriz(float mat[][2]){
     int i,j;
    for(i=0;i<2;i++)
       for(j=0;j<2;j++){
         printf("%f\n", mat[i][j]);
       }
}
void f1(float mat[][2]){
     int i, j;
    printf("Matriz alterada em
f1:\n");
    for(i=0;i<2;i++)
       for(j=0;j<2;j++){
         mat[i][j] = i+j+10;
         printf("f1: %f\n",
mat[i][j]);
       }
}
```

Exercícios

1. Faça um programa que lê uma string e um caractere e chama uma função para contar quantas vezes o caractere ocorre na string. Use ponteiros para receber e manipular a string na função.

1. Faça um programa que lê dois vetores de inteiros com 10 elementos e chama funções para executar as seguintes operações:
   a) Soma de vetores
   b) Subtração de vetores
   c) Produto escalar dos vetores

   Use ponteiros para manipular os vetores nas funções.

3. Faça um programa que lê duas matrizes 3x3 e efetua a soma das matrizes usando funções. Use ponteiros para manipular as matrizes nas funções.

Ponteiros e Funções: Retornando Ponteiros

- Uma função pode retornar um ponteiro da mesma forma como retorna outros tipos de dado.
- Para isto, precisa ser declarada de forma a retornar um ponteiro.
- Para declarar um ponteiro para um certo tipo de dado:

```
tipo *nome_da_variável;
```

Ponteiros e Funções: Retornando Ponteiros

- De forma análoga, para declarar uma função que retorna ponteiros, deve-se usar:

```
tipo *nome_da_função(parametros)
```

Ponteiros e Funções: Retornando Ponteiros

- A função a seguir retorna um ponteiro para char

```
char *f(char c, char *s){
    while (c!=*s && *s) s++;
    return(s);
}
```

Ponteiros e Funções: Retornando Ponteiros

```
#include<stdio.h>
char *f(char c, char *s);
main(void)
{
  char *p, palavra[80], letra;
  printf("Digite uma letra: ");
  scanf("%c", &letra);
  printf("Digite uma palavra: ");
  scanf("%s", palavra);
  printf("%s  %c\n", palavra, letra);
  p = f(letra, palavra);
  if (*p)   printf("%s  %c", palavra, letra);
  else printf("A letra %c não existe em %s\n", letra, palavra);
}

char *f(char c, char *s){
    while (c!=*s && *s) s++;
    return(s);
}
```

Exercícios B

1. Faça um programa que lê dois valores inteiros e chama uma função que retorna um ponteiro para o maior deles.