Os *valores* associados aos parâmetros durante a vida de uma função chamada são determinados com base em como esses argumentos são *passados* para a função. A linguagem de programação C permite que os parâmetros sejam passados para as funções de duas maneiras: ***por valor*** e ***por referência***.

1) Passagem de Parâmetros por Valor

Passar um argumento *por valor* significa que o valor do argumento é calculado no tempo da chamada e copiado para o parâmetro correspondente.

Exemplo:

```
int soma (int p1, int p2)
{
  p1+= p2;
  return p1;
}
```

```
int main()
{
  int v1=3, v2=5;
  printf("%d",soma(v1,v2));
}
```

Neste exemplo a passagem de parâmetros é ilustrada na chamada *soma(v1,v2)* de *main*. Ñesta chamada, os argumentos *v1* e *v2* são passados *por valor* e os parâmetros formais (*p1* e *p2)* da função *soma* usam cópias dos valores de *v1* e *v2* respectivamente. Ainda, na função *soma* o valor de *p1* é alterado, mas a variável *v1* de *main* permanece inalterada.

2) Passagem de Parâmetros por Referência

Passar um argumento *por referência* (ou *por endereço*) significa que o *endereço de memória* do argumento é copiado para o parâmetro correspondente, de modo que o parâmetro se torna uma *referência* (ponteiro) indireta ao argumento real.

Exemplo:

```
void troca(int *x, int *y)
{
 int aux;
 aux=*x;
 *x=*y;
 *y=aux;
}
```

```
int main()
int main()
{
 int a=0,b=5;
 troca(&a,&b);
 printf("%d %d",a,b);
}
```

Neste exemplo a passagem de parâmetro é ilustrada na chamada *troca(&a,&b)* de *main*. Nesta chamada, os endereços dos argumentos *a* e *b*, escritos da forma *&a* e *&b* são passados para as variáveis ponteiros *x* e *y*. Assim, os valores de *a* e *b* são atualizados de acordo com as modificações em *x* e *y*.

Os argumentos são passados por referência nos casos em que o valor real do argumento deva mudar durante a vida da chamada e também quando o argumento passado ocupa muita memória, como no caso de uma matriz grande. Nesses casos, a sobrecarga de espaço e tempo necessária para criar uma cópia completa do objeto, se ele fosse passado por valor, é evitada pela cópia de um único endereço para o parâmetro.

Exercícios:

1) Escreva uma função com protótipo

```
void somabit (int b1, int b2, int *vaium, int *soma);
```

que recebe três bits (inteiros 0 ou 1): *b1* , *b2* e **vaium* e devolve um bit *soma* que armazena o resultado da soma dos três primeiros, e o novo bit "vai-um" em **vaium*.

2) Seja o código abaixo usando passagem por referência. Analise o código e explique o resultado mostrando passo a passo as alterações ocorridas no vetor *a*. Verifique, por fim, se a execução desse código produz algum efeito prejudicial à legibilidade.

```
void incrementa(int *x, int *y){
     *x = *x + (*y);
     (*y)++;
     }

int main(){
    int a[] = {1,2,3};
    for (int i=0; i<3; i++){
        incrementa(&a[i],&a[1]);
        printf("\n %d",a[i]);
        }
    }
```

3) Escreva um programa que receba um número inteiro representando a quantidade total de segundos e, usando passagem de parâmetros por referência, converta a quantidade informada de segundos em Horas, Minutos e Segundos. Imprima o resultado da conversão no formato HH:MM:SS. Utilize a seguinte função protótipo:

```
void converteHora (int total_segundos, int *hora, int *min, int *seg);
```

4) Dado o código abaixo utilizando os argumentos *argc* e *argv*. Experimente executar este programa com diferentes argumentos e explique o resultado.

```
#include <stdio.h>
int main(int argc, char *argv[] ) {
int i;
  printf("\n %d", argc);
  for(i=0; i < argc; i++)
     printf("\n %s ",argv[i]);
}
```

5) Escreva um programa que faça uso dos parâmetros *argv* e *argc*. O programa *main* deverá receber o dia, mês e ano, e verificar se as entradas formam uma data válida. Em caso positivo, imprimir a data seguindo o exemplo:

Entrada:		Saída:
01/11/2011	==>	01 de novembro de 2011

6) Escreva um programa que faça o uso dos parâmetros *argv* e *argc*. O programa *main* deverá receber a quantidade de elementos para alocar dinamicamente um vetor de números inteiros. A quantidade máxima permitida é 30 elementos, se o usuário informar um número maior do que esse, o programa deverá exibir uma mensagem de alerta e encerrar. Preencha o vetor com números randômicos e imprima na tela.

Bibliografia:

Tucker A. B., Noonan R. B. *Linguagens de Programação: Princípios e Paradigmas*, McGraw-Hill, 2009.