Fundamentos de Programação 1

Linguagem C
“Atribuição, Aritmética e Comparação de Ponteiros - Vetores e Ponteiros”.

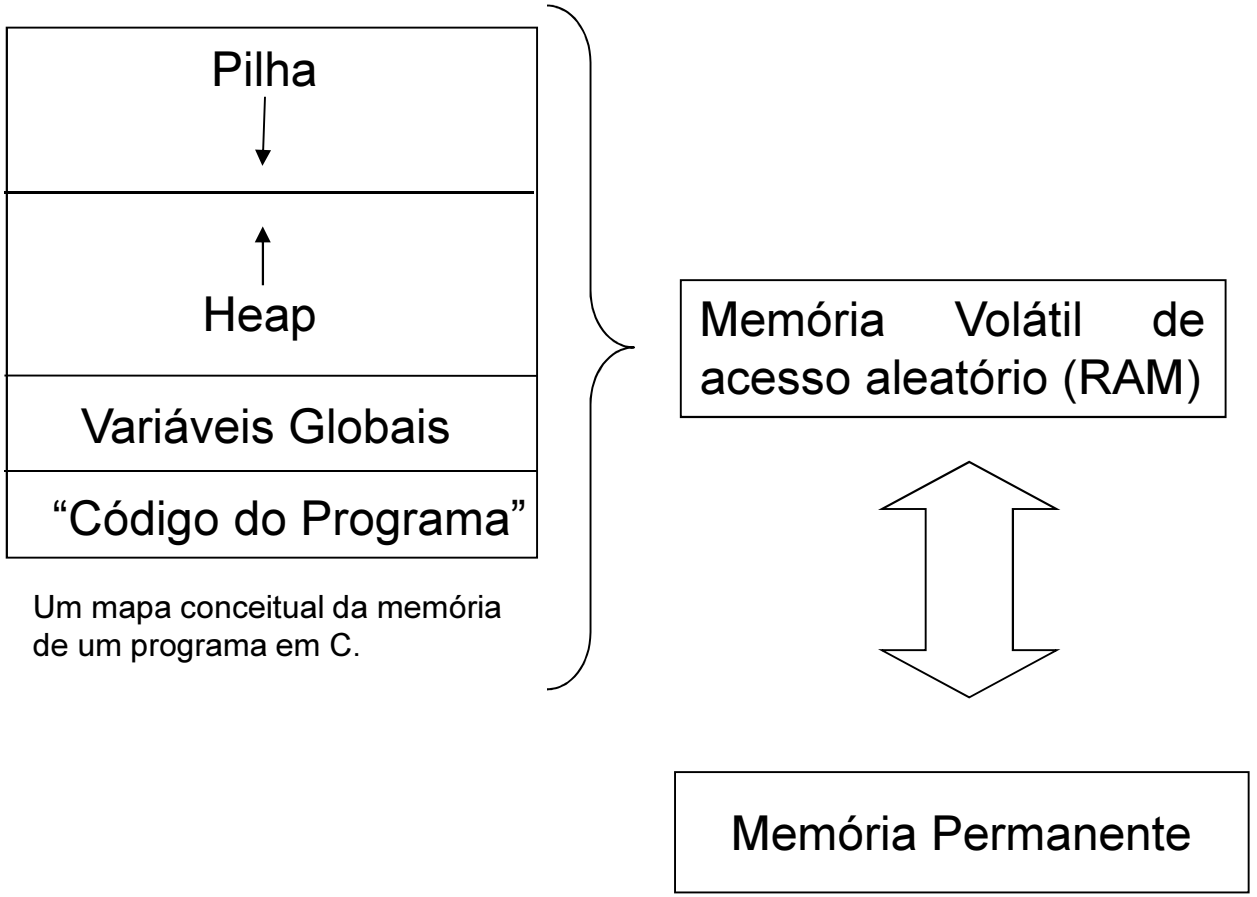
Slides 15

Prof. SIMÃO

Atribuição de Ponteiros.

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

void main ()
{

   int x = 3;

   int *p1 = 0, *p2 = 0;

   p1 = &x;

   p2 = p1;

   printf (" O endereço da variável x é %p. \n",  p2 );

   system ("Pause");

   return;

}
```

Memória



Um mapa conceitual da memória de um programa em C.

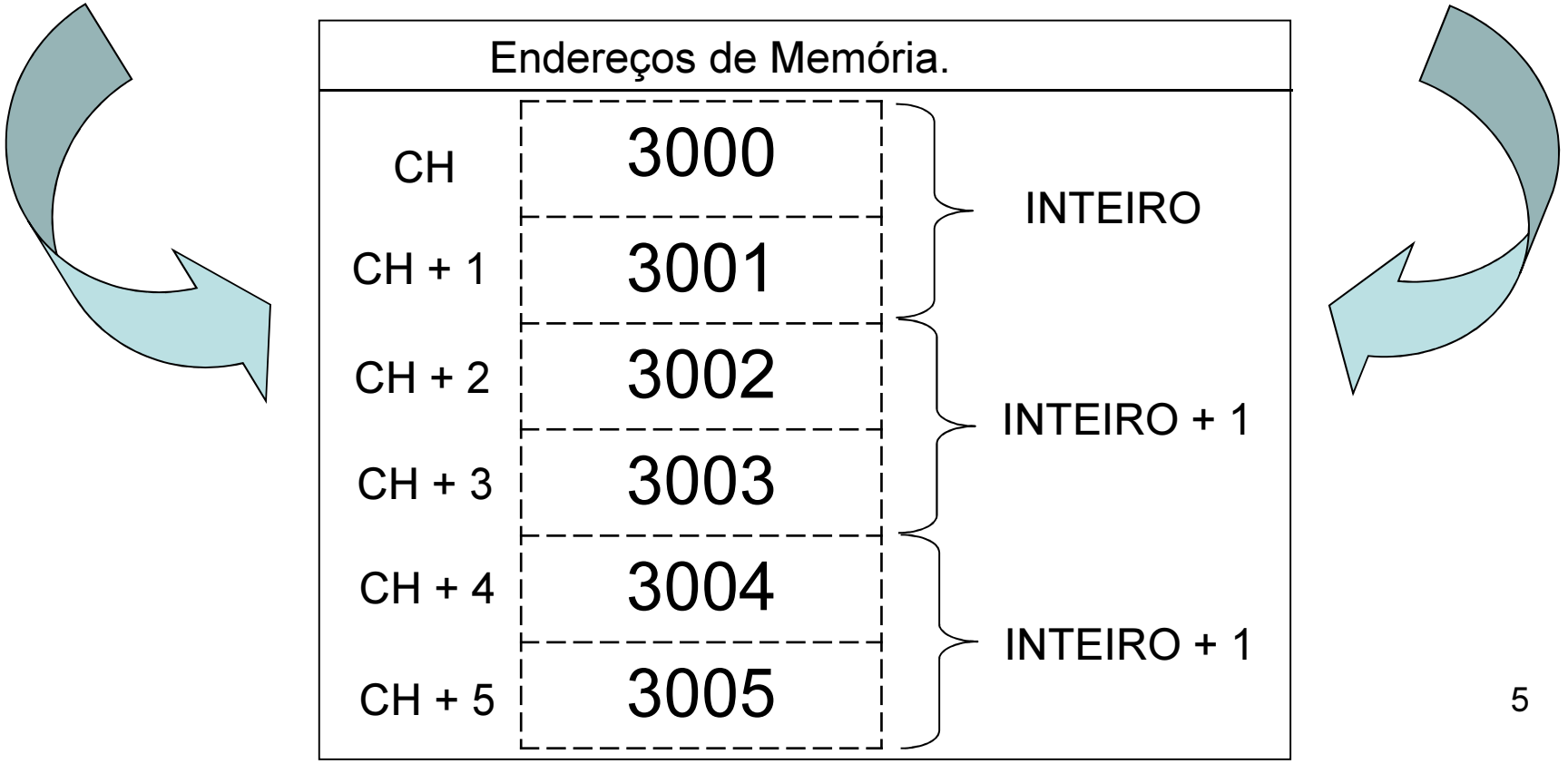
Variáveis em C

Tipo de dados	Variação	Total de Bytes Utilizados
char	0 a 255	1
int	-32.768 a 32.767	2
short int	-128 a 127	1
unsigned int	0 a 65.535	2
long int	-4.294.967.296 a 4.294.967.296	4
float	Aproximadamente 6 dígitos de precisão	4
double	Aproximadamente 10 dígitos de precisão	8
long double	Aproximadamente 10 dígitos de precisão	10
void	-	0

Aritmética de Ponteiro

```
void main ()
{
   char *CH;
   CH = 3000;
   printf (" O endereço apontado por ch é %p. \n",  ch );
   CH = CH + 1;
   printf (" O endereço apontado por ch é %p. \n",  ch );
   CH = CH + 1;
   printf (" O endereço apontado por ch é %p. \n",  ch );
}
```

```
void main ()
{
   int *INTEIRO;
   INTEIRO = 3000;
   printf (" O endereço apontado por INTEIRO é %p. \n",  INTEIRO );
   INTEIRO = INTEIRO + 1;
   printf (" O endereço apontado por INTEIRO é %p. \n",  INTEIRO );
   INTEIRO = INTEIRO + 1;
   printf (" O endereço apontado por INTEIRO é %p. \n",  INTEIRO );
}
```



Vetores e Ponteiros

```
#include <stdio.h>
#include <stdlib.h>
int main()
{   // Define e inicializa uma string.
    char str [80] = "Universidades";
    // Um ponteiro chamando Pont para caractere.
    char *Pont;
    // O ponteiro Pont recebe o endereço da primeira posição da str.
    // O nome de um string sozinho sempre é o endereço da primeira posição.
    // Obs.: Neste caso, não é necessário &.
    Pont = str;

    // Enquanto o conteúdo do ponteiro não for \0.
    while ( *Pont != '\0')
    {
        putchar ( *Pont );
        // Aritmética de ponteiros
        Pont = Pont + 1;
    }
    printf("\n");

    Pont = str;

    // O ponteiro que aponta para o primeiro elemento de uma string pode ainda se
    // comportar como um vetor!!!!
    int idx = 0;
    while( Pont [ idx ] != '\0‘ )
    {
       putchar( Pont[idx] );
       idx = idx + 1;
    }
    printf("\n");
    system("Pause");
    return 0;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
int main()
{   // Define e inicializa uma string.
    char str [80] = "Universidades";
    // Um ponteiro chamando Pont para caractere.
    char *Pont;
    // O ponteiro Pont recebe o endereço da primeira posição da str.
    // O nome de um string sozinho sempre é o endereço da primeira posição.
    // Obs.: Neste caso, não é necessário &.
    Pont = str;

    // Enquanto o conteúdo do ponteiro não for \0.
    while ( *Pont )
    {
        putchar (*Pont);
        // Aritmética de ponteiros
        Pont = Pont + 1;
    }
    printf("\n");

    Pont = str;

    // O ponteiro que aponta para o primeiro elemento de uma string pode ainda se
    // comportar como um vetor!!!!
    int idx = 0;
    while( Pont[idx] )
    {
       putchar( Pont[idx] );
       idx = idx + 1;
    }
    printf ("\n");
    system("Pause");
    return 0;
}
```

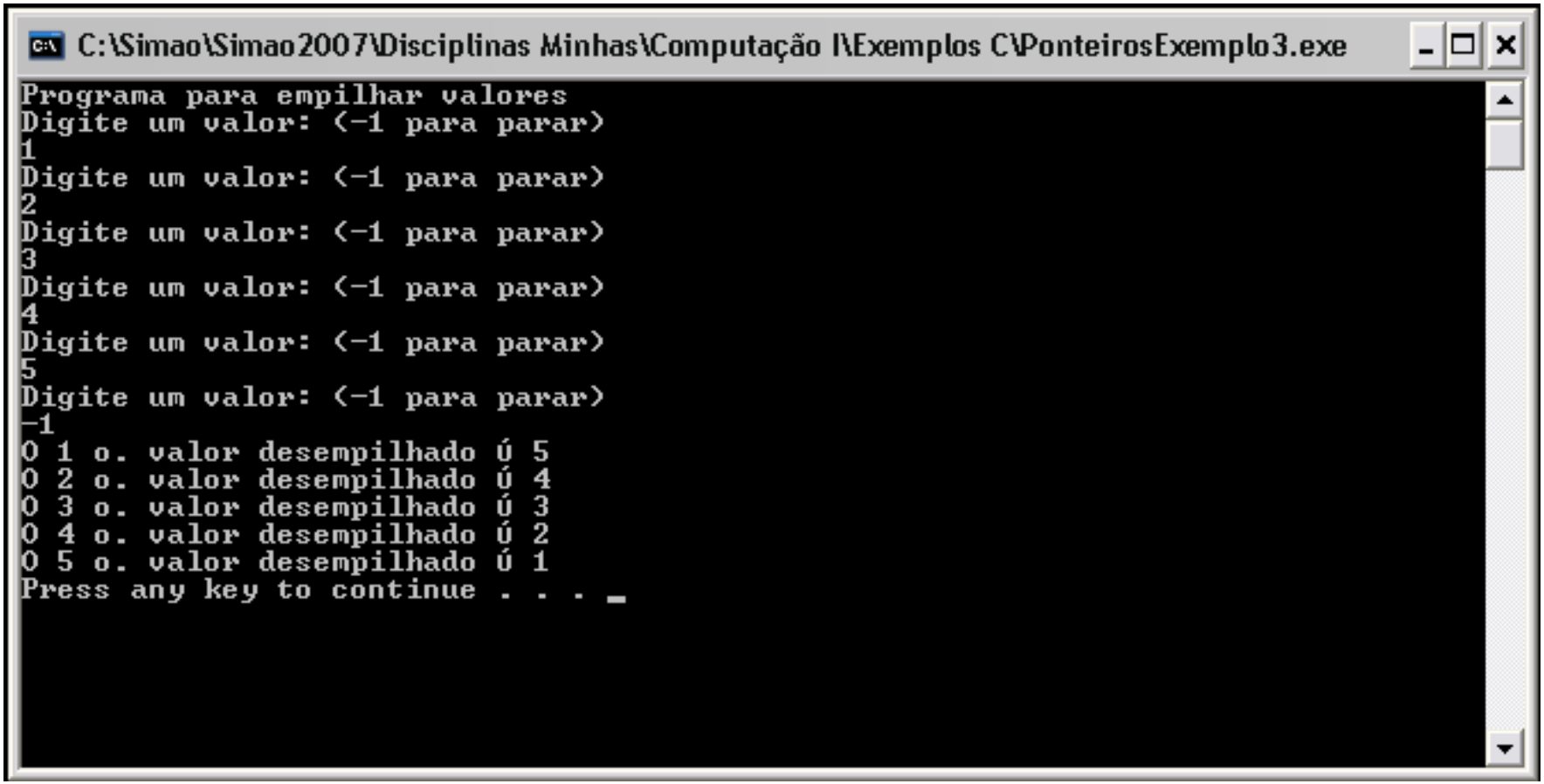
Exemplo - Pilha

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

#define TAMANHO 5

// cabeçalho de função para empilhar número em um vetor
void empilha ( int i );

// cabeçalho de função para desempilhar número em um vetor
int  desempilha ();

// ponteiro para topo (fim) de vetor
int *Topo;

// ponteiro para base (início) de vetor
int *Base;

// ponteiro para navegar sobre o vetor pilha via aritmética de ponteiros
int *Pont;

// vetor para pilha (primeiro número a entrar é ultimo a sair)
int Pilha[TAMANHO];
```

```
// função principal
int main()
{
   int valor, cont = 0;
   // Isto significa que o ponteiro Base recebe o endereço de início do vetor Pilha.
   // (Note que não há & comercial para receber o endereço quando se trata de um vetor!)
   // (Isto porque o próprio (nome do) vetor é internamente implementado como ponteiro...)
   // (Isto é, o nome do vetor é um ponteiro)
   Base = Pilha;
   // O ponteiro Pont recebe o ponteiro Base
   Pont = Base;
   // O ponteiro Topo recebe o endereço de fim do vetor Pilha
   Topo = Base + TAMANHO;

   // O programa em si.
   printf ("Programa para empilhar valores \n");
   do
   {
       printf ( "Digite um valor: (-1 para parar) \n“ );
       scanf ( "%d", &valor );
       if ( valor != -1 )
       {
          empilha (valor);
          cont = cont + 1;
       }
   } while ( valor != -1 );

   int i;
   for ( i = 0; i < cont; i = i + 1 )
   {
        valor = desempilha();
        printf("O %d o. valor desempilhado é %d \n", i+1, valor);
   }
   system("Pause");
   return 0;
}
```

```
void empilha ( int i )
{
    // Se o ponteiro Pont estiver apontando já para o topo
    // então não se pode mais empilhar o número i

    if ( Pont  == Topo )
    {

       printf ("Pilha cheia! \n");
       // sai do programa
       exit (1);

    }
    else
    {

      // A "variável" apontada por Pont recebe o valor de i , i.e. a posição Pilha[0] recebe o valor de i.
      *Pont = i;

      // ponteiro Pont avança da sua posição inicial em 2 bytes (tamanho de um inteiro)
      // (i.e. Pont avança para seu endereço imediatamente superior)
      Pont = Pont + 1;

   }
}
```

```
int desempilha ( )
{
   // variável intermediária
   int valor;

   // Se o ponteiro Pont estiver num endereço maior ou igual que o do Base, isto é,
   // Se ele tiver endereço maior ou igual que início do Vetor Pilha...
   // então se extrai a informação em uma variável intermediária.
   if ( Pont >= Base )
   {
      // Ponteiro Pont aponta para seu endereço imediatamente inferior
      Pont = Pont - 1;

      //  Valor recebe o valor da 'variável' apontado por Pont.
      valor = *Pont;
   }
   else
   {
      printf ("Pilha vazia! \n");
      exit (1);
   }
   return valor;
}
```


C:\Simao\Simao2007\Disciplinas Minhas\Computação I\Exemplos C\PonteirosExemplo3.exe
Programa para empilhar valores
Digite um valor: (-1 para parar)
1
Digite um valor: (-1 para parar)
2
Digite um valor: (-1 para parar)
3
Digite um valor: (-1 para parar)
4
Digite um valor: (-1 para parar)
5
Digite um valor: (-1 para parar)
-1
O 1 o. valor desempilhado ú 5
O 2 o. valor desempilhado ú 4
O 3 o. valor desempilhado ú 3
O 4 o. valor desempilhado ú 2
O 5 o. valor desempilhado ú 1
Press any key to continue . . .