Ponteiros

Estruturas de Dados 1

Prof. John L. Gardenghi

22 de agosto de 2019

Faculdade UnB Gama
Universidade de Brasília

Operador sizeof

```c
#include <stdio.h>

int main() {
  int array[20];
  char string[20];

  printf( "TAMANHOS:\n"
          "\t char.........: %ld\n"
          "\t short........: %ld\n"
          "\t int..........: %ld\n"
          "\t long.........: %ld\n"
          "\t float........: %ld\n"
          "\t double.......: %ld\n"
          "\t long double..: %ld\n"
          "\t vetor int....: %ld\n"
          "\t string.......: %ld\n",
          sizeof(char), sizeof(short), sizeof(int), sizeof(long),
          sizeof(float), sizeof(double), sizeof(long double),
          sizeof(array), sizeof(string) );

  return 0;
}
```

Endereço de uma variável

```c
#include <stdio.h>
int main() {
  int i = 5;
  printf( "&i (dec.): %ld\n", (long int) &i );
  printf( "&i (hexa.): %p\n", (void *) &i );
  return 0;
}
```

Declação e uso de ponteiros

```c
#include <stdio.h>
int main () {
  int i; int *p;
  i = 1234; p = &i;
  printf ("*p = %d\n", *p);
  printf (" p = %ld\n", (long int) p);
  printf (" p = %p\n", (void *) p);
  printf ("&p = %p\n", (void *) &p);
  return 0;
}
```

Exemplo de aplicação

Passagem por referência (vs. passagem por valor).

```
void troca (int *p, int *q) {
  int temp;
  temp = *p;
  *p = *q;
  *q = temp;
}

int main() {
  int a = 2, b = 3;
  troca(&a, &b);
  printf( "a = %d, b = %d\n, a, b );
  return 0;
}
```

Indireção múltipla

Como podemos arrumar o código abaixo para imprimir 10 usando q?

```c
#include <stdio.h>
int main() {
  int x;
  int *p = &x;
  int **q = &p;
  x = 10;
  printf("%d\n", &q);
  return 0;
}
```

Ponteiros e vetores

Observe a saída deste código.

```
int main() {
  int i, offset;
  int b[] = { 10, 20, 30, 40 };
  int *bPtr;
  bPtr = b;
  printf( "VETOR\n" );
  printf( "\n\tNotacao usando indices:\n" );
  for ( i = 0; i < 4; i++ )
    printf( "\t\tb[%d] = %d\n", i, b[i] );
  printf( "\n\tNotacao usando deslocamento\n" );
  for ( offset = 0; offset < 4; offset++ )
    printf( "\t\t*(b + %d) = %d\n", offset, *(b + offset) );
  printf( "\nPONTEIRO\n" );
  printf( "\n\tNotacao usando indices:\n" );
  for ( i = 0; i < 4; i++ )
    printf( "\t\tbPtr[%d] = %d\n", i, bPtr[i] );
  printf( "\n\tNotacao usando deslocamento\n" );
  for ( offset = 0; offset < 4; offset++ )
    printf( "\t\t*(bPtr + %d) = %d\n", offset, *(bPtr + offset) );
  return 0;
}
```

Ponteiros e vetores

Como corrigir este código?

```
int *primos (void) {
  int v[3];
  v[0] = 1009; v[1] = 1013; v[2] = 1019;
  return v;
}
```

Ponteiros e vetores

Qual a saída deste código?

```
int main() {
  int arr[] = {  9,   8,  98,  88,
                87,   1,   2,   4,
               101, 102, 103, 105 };
  int *x =  arr+4;
  int *ptr = &arr[7];

  arr[*ptr]++;
  printf( "Valor 1: %d\n", *ptr );
  printf( "Valor 2: %d\n", *x );
  *x = 7;
  printf( "Valor 3: %d\n\n", arr[4] + *ptr );

  return 0;
}
```

Ponteiros e vetores

Qual a saída deste código?

```
int main() {
  int b[5] = { 1, 2, 3, 4, 5 };
  int *bPtr;
  int i;
  bPtr = b;
  *(bPtr+2) += 10;
  bPtr = bPtr+2;
  for ( i = 0; i < 5; i++ )
    printf( "b[%d] = %d\n", i, b[i] );
  printf("\n");
  for ( i = 0; i < 5; i++ )
    printf( "bPtr[%d] = %d\n", i, bPtr[i] );
  return 0;
}
```

Ponteiros e vetores

Qual a saída deste código?

```
int main() {
  int var;
  char *ptr;
  ptr = &var;
  ptr[0] = 's';
  ptr[1] = 'o';
  ptr[2] = 'l';
  ptr[3] = '\0';
  printf( "%s ... var = %d\n\n", (char *) ptr, var);
  var = var - 3584;
  printf( "%s ... var = %d\n\n", (char *) ptr, var);
  return 0;
}
```

Ponteiros e vetores - Exemplo de Alocação Dinâmica 1

```c
#include <stdio.h>
#include <stdlib.h>
int main() {
  int i, n;
  double *v;
  double soma;
  scanf( "%d", &n );
  v = malloc( n * sizeof(double) );
  if ( v == NULL ) {
    printf( "Erro na alocacao de memoria.\n" );
    return 1;
  }
  for ( i = 0; i < n; i++ )
    scanf( "%lf", &v[i] );
  soma = 0;
  for ( i = 0; i < n; i++ )
    soma += v[i];
  printf( "Media = %lf\n", soma / n );
  free( v );
  return 0;
}
```

Ponteiros e vetores - Exemplo de Alocação Dinâmica 2

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main() {
  int **matriz;
  int i, j, m, n;
  scanf( "%d %d", &m, &n );
  matriz = malloc( m * sizeof( int * ) );  /* Aloca espaço */
  if ( matriz == NULL ) {
    printf( "Erro de alocacao\n" );
    return 1;
  }
  for ( i = 0; i < m; i++ ) {
    matriz[i] = malloc( n * sizeof( int ) );
    if ( matriz[i] == NULL ) {
      printf( "Erro de alocação\n" );
      return 1;
    }
  }
  srand( time( NULL ) );                  /* Sorteia números para a matriz entre 0 e 100 */
  for( i = 0; i < m; i++ )
    for ( j = 0; j < n; j++ )
      matriz[i][j] = rand()%101;
  for( i = 0; i < m; i++ ) {              /* Imprime elementos da matriz */
    for ( j = 0; j < n; j++ )
      printf( "%3d ", matriz[i][j] );
    printf( "\n" );
  }
  for ( i = 0; i < m; i++ )               /* Libera memória */
    free( matriz[i] );
  free( matriz );
  return 0;
}
```

Ponteiros e vetores: retorno de função

```
char *maiusculo( char str[] ) {
  int d, i, n;
  char *ptr;
  /* Calcula o tamanho de str */
  n = -1;
  do {
    n++;
  } while( str[n] != '\0' );
  /* Aloca ptr */
  ptr = mallocc( n );
  i = 0;
  do {
    ptr[i] = str[i];
    if ( str[i] >= 'a' && str[i] <= 'z' )
      ptr[i] -= 32;
    i++;
  } while( str[i] != '\0' );
  printf( "%s\n", ptr );
  return ptr;
}
```

Ponteiros para função

```c
#include <stdio.h>
#include <stdlib.h>
int maior( int a, int b ) {
  return a > b;
}

int menor( int a, int b ) {
  return a < b;
}

void minmax( int n, int *v, int (*compara)( int a, int b ) ) {
  int i, elem;
  elem = v[0];
  for ( i = 1; i < n; i++ )
    if ( (*compara)( v[i], elem ) )
      elem = v[i];
  printf( "elem = %d\n", elem );
}

int main() {
  int v[5] = { 5, 2, 4, 8, 6 };
  minmax( 5, v, maior );
  minmax( 5, v, menor );
  return 0;
}
```