

Computação Eletrônica Juliano M. Iyoda jmi@cin.ufpe.br



- Definição
- Exemplo

- Definição
- Exemplo

Outros assuntos Definição

- Podemos alocar memória durante a execução do programa
- malloc()

p inicialmente não aponta para uma memória alocada (reservada)	Precisamos alocar (reservar) uma memória para papontar para ela.
#include <stdio.h></stdio.h>	#include <stdio.h></stdio.h>
<pre>int main() {</pre>	<pre>#include<stdlib.h></stdlib.h></pre>
int *p;	<pre>int main() {</pre>
*p = 30; // ERRO!	int *p;
return 0;	<pre>p = (int *) malloc(sizeof(int));</pre>
}	*p = 30; // OK!
	free(p);
	return 0;
	}

Outros assuntos Definição

malloc() aloca um pedaço de memória do tamanho de um número inteiro. E retorna o endereço desse pedaço de memória.

free() libera essa memória.

- Podemos alocar memória durante a execução do programa
- malloc()

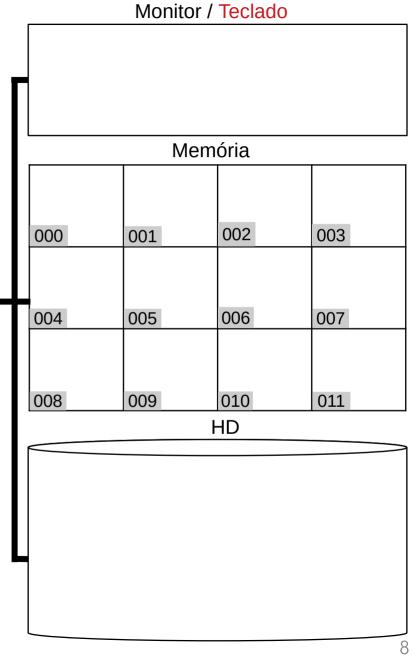
p inicialmente não aponta para uma memória alocada (reservada)	Precisamos alocar (reservar) uma memória para papontar para ela.
<pre>#include<stdio.h> int main() {</stdio.h></pre>	<pre>#include<stdio.h> #include<stdlib.h></stdlib.h></stdio.h></pre>
<pre>int *p; *p = 30; // ERRO! return 0;</pre>	<pre>int main() { int *p; p = (int *) malloc(sizeof(int));</pre>
}	*p = 30; // OK! free(p); return 0; }

- Definição
- Exemplo

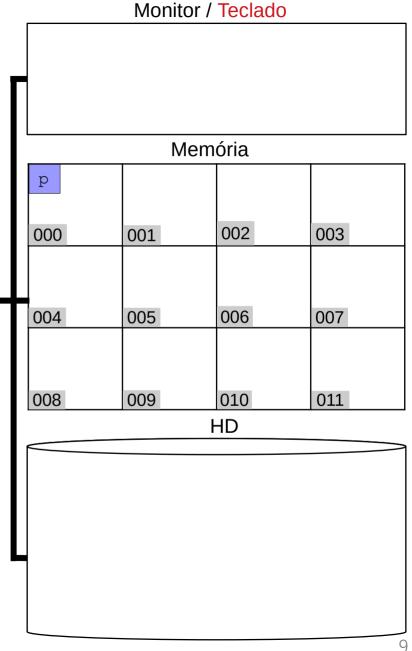
Outros assuntos Exemplo

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
   printf("*p = %d\n", *p);
   free(p);
   return 0;
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
   printf("*p = %d\n", *p);
   free(p);
   return 0;
```



```
#include<stdio.h>
#include<stdlib.h>
lint main() {
  int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
   printf("*p = %d\n", *p);
   free(p);
   return 0;
```



```
#include<stdio.h>
#include<stdlib.h>
lint main() {
   int *p;
   printf("p = pn', p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
   printf("*p = %d\n", *p);
   free(p);
   return 0;
 Pode aparecer (nil) ou outro número arbitrário.
 Independente disso, é um endereço que não está
 alocado (reservado) para usarmos.
```

Monitor / Teclado p = ((nil))Memória 000 002 003 001 006 004 005 007 800 009 010 011 HD

```
#include<stdio.h>
#include<stdlib.h>
lint main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
   printf("*p = %d\n", *p);
   free(p);
   return 0;
 malloc() reserva um local na memória do tamanho de
 um inteiro (sizeof(int)). E retorna esse endereço.
 Assumimos que malloc() reservou no endereço 007.
```

```
#include<stdio.h>
#include<stdlib.h>
|int main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
  printf("p = %p\n",p);
   *p = 30;
   printf("*p = %d\n", *p);
   free(p);
   return 0;
```

```
Monitor / Teclado
p = (nil)
p = 007
              Memória
  007
                   002
000
         001
                             003
004
                   006
         005
                             007
800
          009
                   010
                             011
                  HD
```

```
#include<stdio.h>
#include<stdlib.h>
|int main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
  *p = 30;
   printf("*p = %d\n",*p);
   free(p);
   return 0;
```

```
Monitor / Teclado
p = (nil)
p = 007
              Memória
  007
000
          001
                    002
                              003
004
                   006
          005
                              007
800
          009
                   010
                              011
                  HD
```

```
p = 007
#include<stdio.h>
                                                        Memória
#include<stdlib.h>
|int main() {
                                                007
   int *p;
                                                           002
                                              000
                                                     001
                                                                  003
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
                                                                    30
   printf("p = p \in n",p);
                                              004
                                                           006
                                                     005
                                                                  007
   *p = (30);
   printf("*p = %d\n", *p);
                                              008
                                                     009
                                                           010
                                                                  011
   free(p);
                                                           HD
   return 0;
```

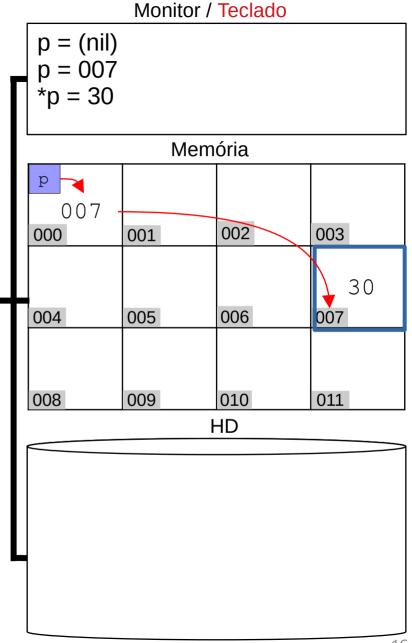
Monitor / Teclado

p = (nil)

```
#include<stdio.h>
#include<stdlib.h>
|int main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
  printf("*p = %d\n",(*p));
   free(p);
   return 0;
```

```
Monitor / Teclado
p = (nil)
p = 007
              Memória
  007
000
         001
                   002
                             003
                                 30
004
                   006
         005
                             007
800
          009
                   010
                              011
                  HD
```

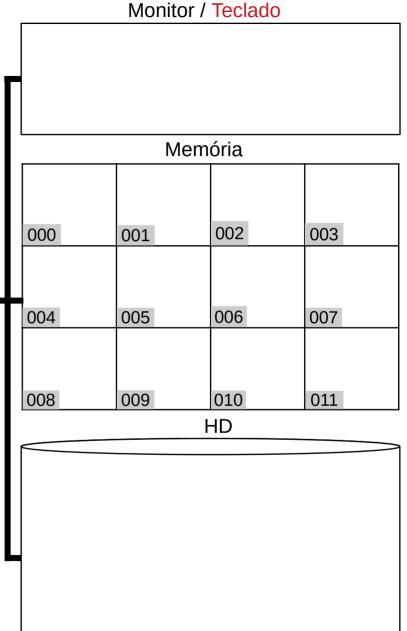
```
#include<stdio.h>
#include<stdlib.h>
|int main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
  printf("*p = %d\n",(*p));
   free(p);
   return 0;
```



```
#include<stdio.h>
#include<stdlib.h>
|int main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
   printf("*p = %d\n", *p);
  free(p);
   return 0;
```

```
p = (nil)
p = 007
*p = 30
              Memória
  007
                   002
000
         001
                             003
004
                   006
         005
                             007
800
         009
                   010
                             011
                  HD
```

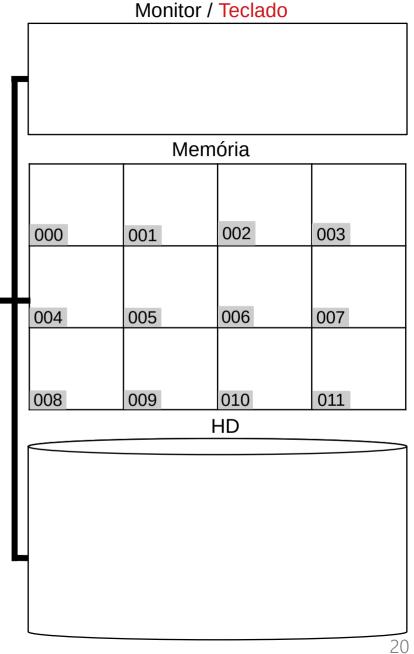
```
#include<stdio.h>
#include<stdlib.h>
lint main() {
   int *p;
   printf("p = p \in n",p);
   p = (int *) malloc(sizeof(int));
   printf("p = p \in n",p);
   *p = 30;
   printf("*p = %d\n", *p);
   free(p);
   return 0;
```



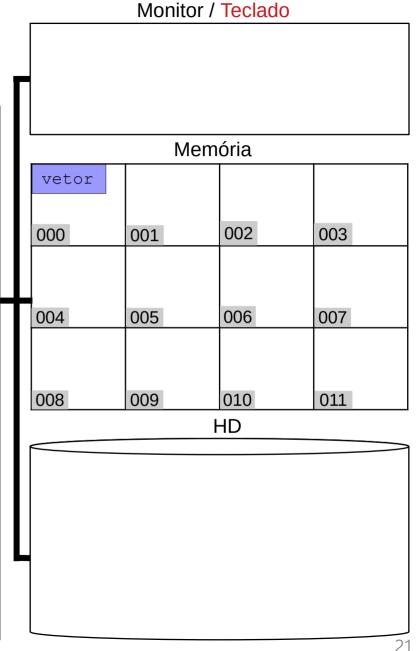
Outros assuntos Exemplo

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
  printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
  vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```

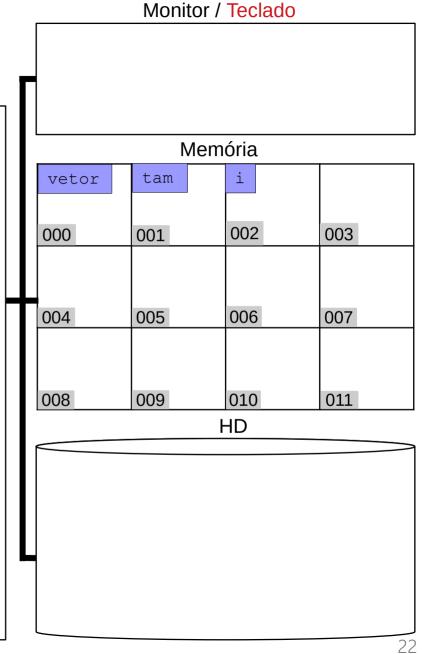
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



```
#include<stdio.h>
#include<stdlib.h>
int main() {
  int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



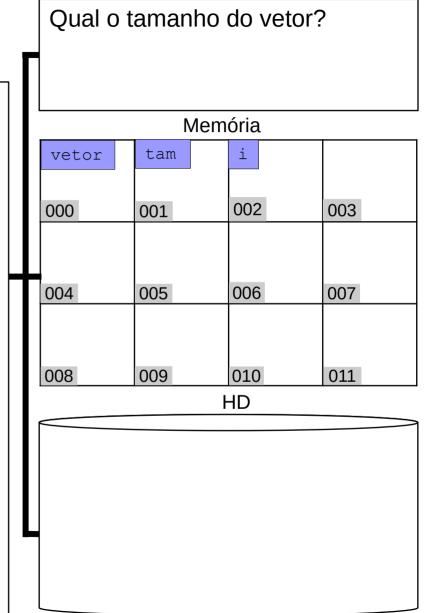
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
  int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



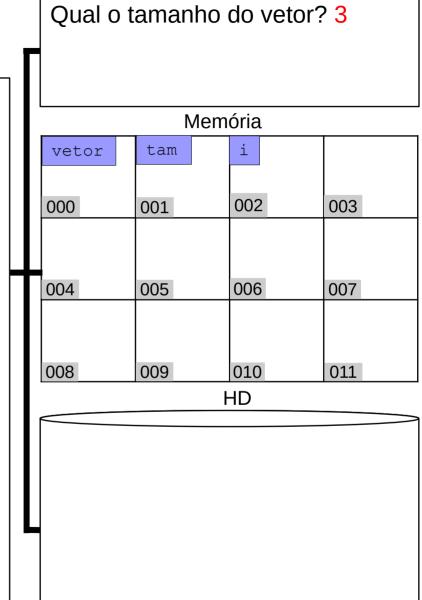
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```

Qual o tamanho do vetor? Memória i vetor tam 000 002 003 001 004 006 005 007 800 009 010 011 HD

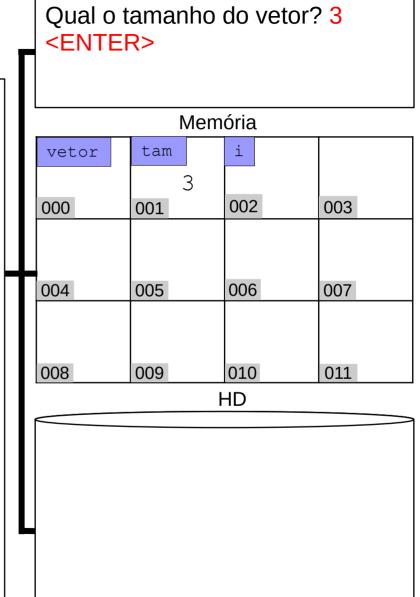
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



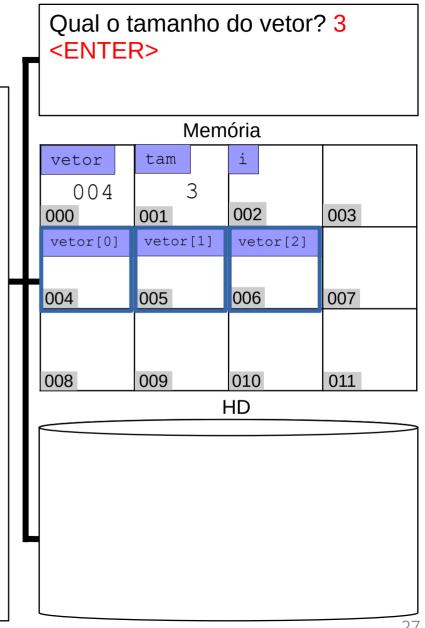
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



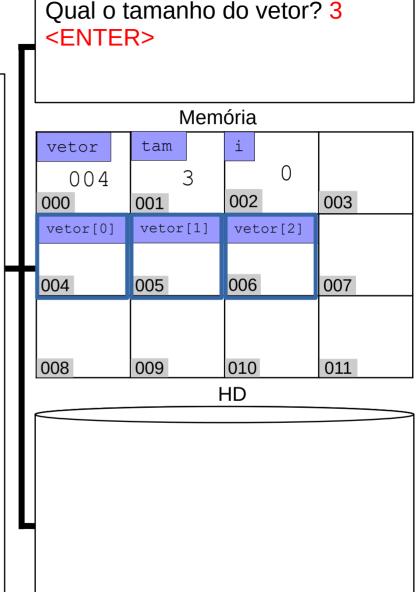
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



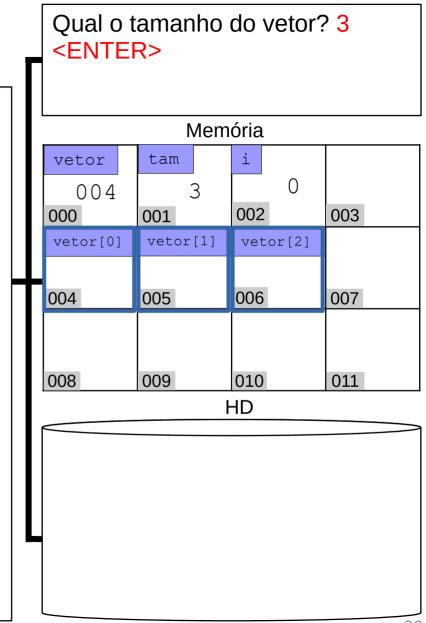
```
#include<stdio.h>
#include<stdlib.h>
lint main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) mallo\alpha(tam)* sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
  malloc() reserva um local na memória do tamanho de
  3 inteiros (3 * sizeof(int)). E retorna o endereço do
  primeiro elemento.
```



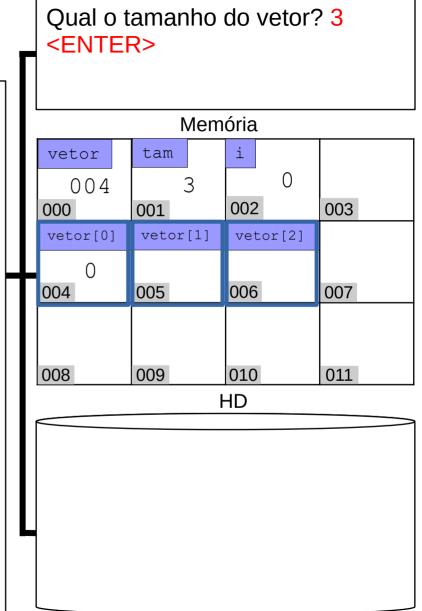
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0) i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; (i < tam) i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0:
                             O computador calcula:
                             i < tam
                             = 0 < 3
                             = 1 (verdadeiro)
```



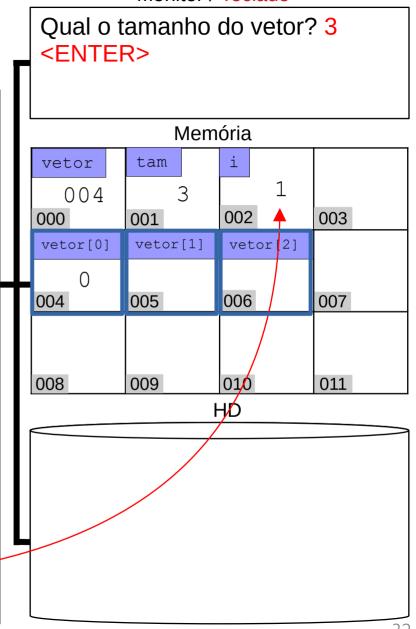
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



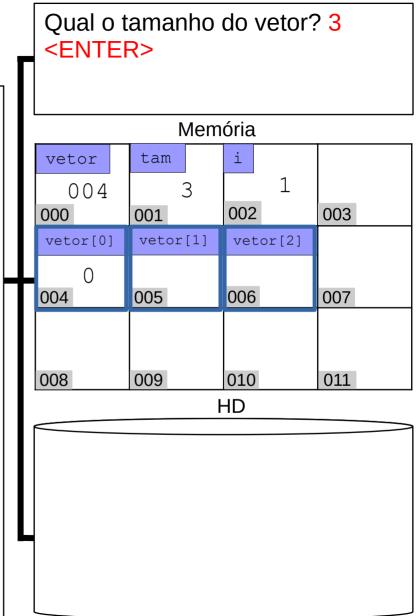
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; (i++)) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                             O computador calcula:
                             i+1
                             = 0 + 1
                             = 1
```

Oual o tamanho do vetor? 3 <FNTFR> Memória i vetor tam 0.04002 003 000 001 vetor[1] vetor[0] vetor[2] 006 004 005 007 800 009 010 011 HD

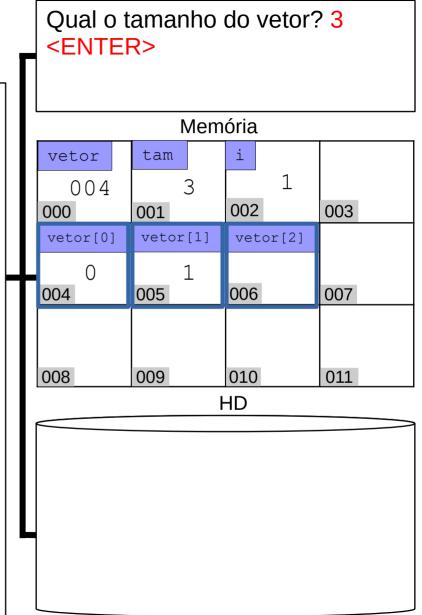
```
#include<stdio.h>
#include<stdlib.h>
int main() {
                                                        vetor
                                                                tam
   int *vetor;
                                                          0.04
   int tam, i;
                                                                001
                                                        000
   printf("Qual o tamanho do vetor? ");
                                                        vetor[0]
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
                                                        004
                                                                005
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
                                                        800
                                                                009
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                              O computador calcula:
                              i + 1
                              = 0 + 1
```



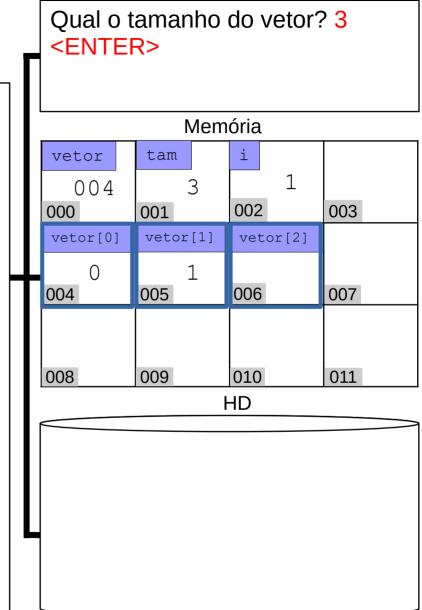
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; (i < tam); i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0:
                             O computador calcula:
                             1 < 3
                             = 1 < 3
                             = 1 (verdadeiro)
```



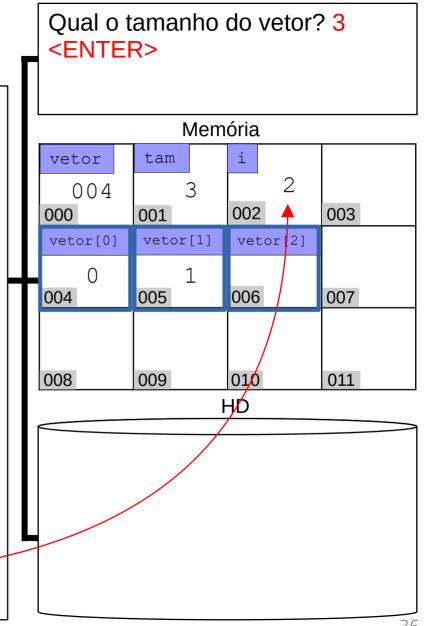
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
     vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



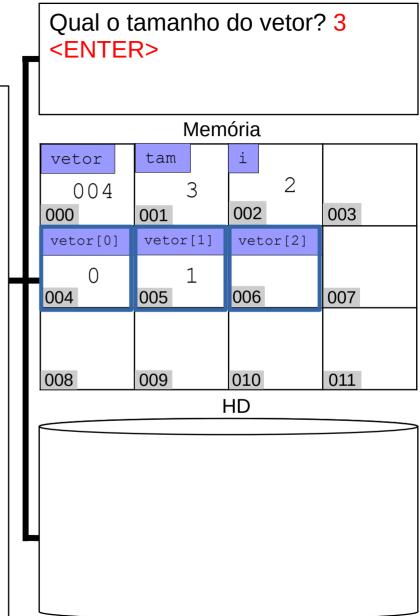
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; (i++)) {
     vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                             O computador calcula:
                             i+1
                             = 1 + 1
                             = 2
```



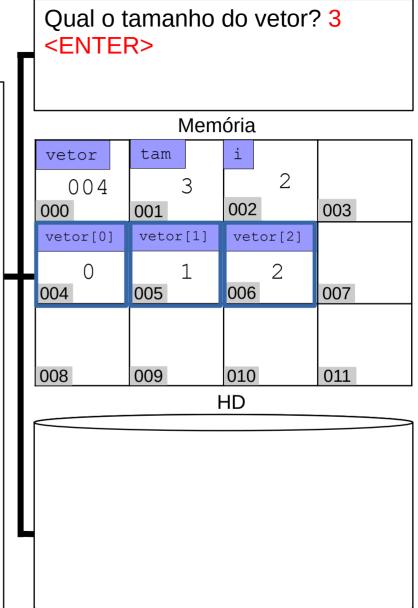
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
                                                       000
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; (i++)) {
                                                       004
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
                                                       800
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                             O computador calcula:
                             i + 1
                             = 1 + 1
```



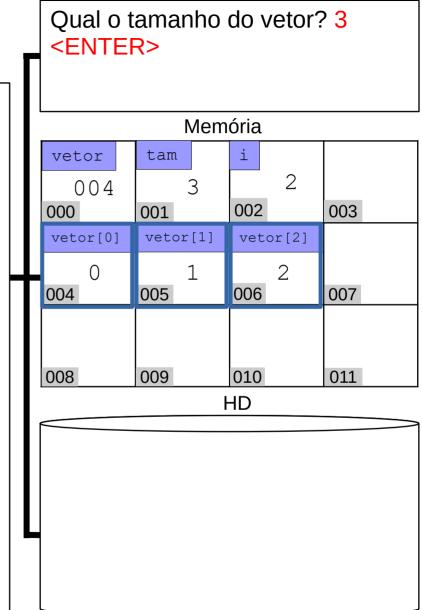
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; (i < tam); i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0:
                             O computador calcula:
                             1 < 3
                             = 2 < 3
                             = 1 (verdadeiro)
```



```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
     vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



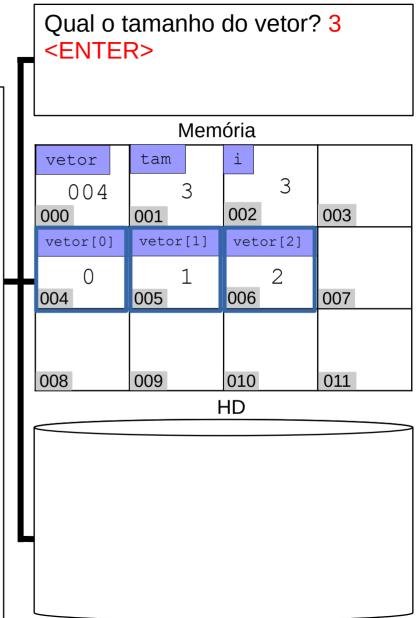
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam;(i++)) {
     vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                             O computador calcula:
                             i+1
                             = 2 + 1
                             = 3
```



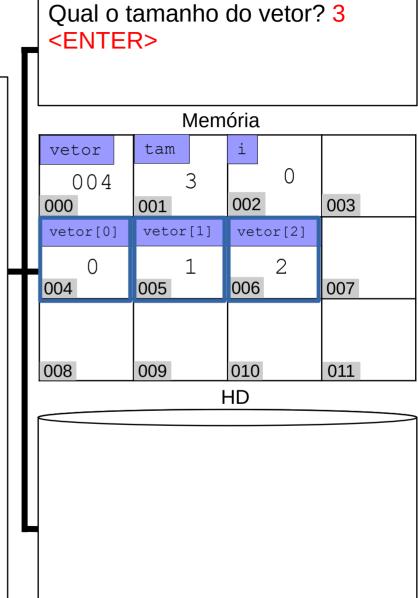
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam;(i++)) {
     vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                             O computador calcula:
                             i+1
                             = 2 + 1
```

Monitor / Teclado Qual o tamanho do vetor? 3 <FNTFR> Memória vetor tam 0.04001 002 003 000 vetor[1] vetor 21 vetor[0] 006 004 005 007 800 009 010 011 HD 40

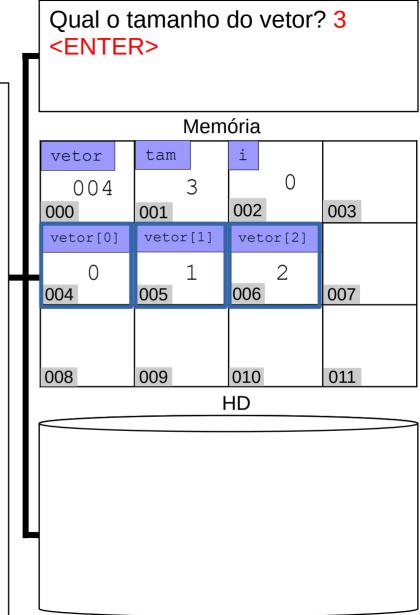
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; (i < tam;) i++) {
      vetor[i] = 1;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0:
                             O computador calcula:
                             i < 3
                             = 3 < 3
                             = O(falso)
```



```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0) i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0:
                             O computador calcula:
                             1 < 3
                             = 0 < 3
                             = 1 (verdadeiro)
```



```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
     printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```

Monitor / Teclado Qual o tamanho do vetor? 3 <ENTER> vetor[0] = 0Memória i vetor tam 0.04002 003 000 001 vetor[1] vetor[0] vetor[2] 004 006 005 007 800 009 010 011 HD

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam;(i++)) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                             O computador calcula:
                             i+1
                             = 0 + 1
                             = 1
```

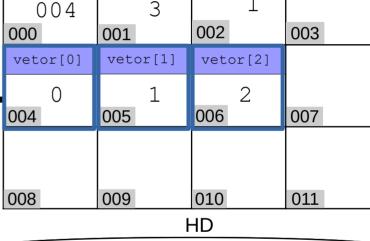
Qual o tamanho do vetor? 3 <ENTER> vetor[0] = 0Memória i vetor tam 0.04002 003 000 001 vetor[0] vetor[1] vetor[2] 006 004 005 007 800 009 010 011 HD

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam;(i++)) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                             O computador calcula:
                             i+1
                             = 0 + 1
```

Qual o tamanho do vetor? 3 <ENTER> vetor[0] = 0Memória vetor tam 0.04002 003 000 001 vetor[1] vetor 21 vetor[0] 006 004 005 007 800 009 01/0 011 ΗĎ 46

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0:
                             O computador calcula:
                             1 < 3
                             = 1 < 3
                             = 1 (verdadeiro)
```

Monitor / Teclado Qual o tamanho do vetor? 3 <ENTER> vetor[0] = 0 Memória vetor tam i 1 004 3 1 000 001 002 003 vetor[0] vetor[1] vetor[2]



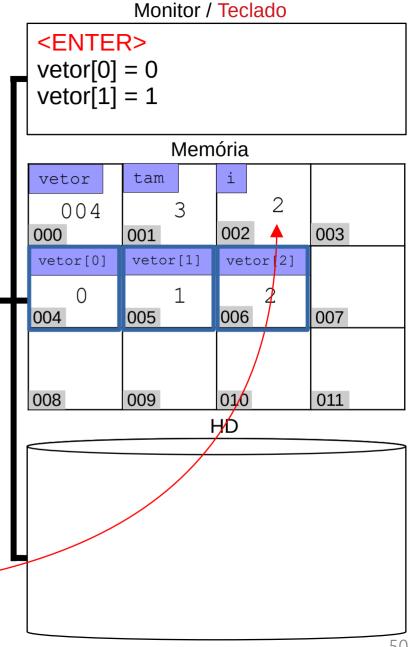
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```

```
<ENTER>
vetor[0] = 0
vetor[1] = 1
              Memória
                   i
vetor
          tam
  0.04
                   002
                            003
000
         001
          vetor[1]
vetor[0]
                   vetor[2]
004
                   006
         005
                            007
800
         009
                  010
                            011
                  HD
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam;<mark>(</mark>i++)) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                              O computador calcula:
                              i+1
                              = 1 + 1
                              = 2
```

```
<ENTER>
vetor[0] = 0
vetor[1] = 1
              Memória
                   i
vetor
          tam
   0.04
                   002
                            003
000
         001
vetor[0]
          vetor[1]
                    vetor[2]
004
                   006
         005
                            007
800
         009
                  010
                             011
                  HD
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam;<mark>(</mark>i++)) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                              O computador calcula:
                              i + 1
                              = 1 + 1
```



```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; (i < tam;) i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0:
                             O computador calcula:
                             1 < 3
                             = 2 < 3
                             = 1 (verdadeiro)
```

```
<ENTER>
vetor[0] = 0
vetor[1] = 1
              Memória
                   i
vetor
          tam
  0.04
                   002
                            003
000
         001
vetor[0]
          vetor[1]
                   vetor[2]
                   006
004
         005
                            007
800
         009
                  010
                            011
                  HD
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
     printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
              Memória
                    i
vetor
          tam
   0.04
                   002
                             003
000
         001
          vetor[1]
vetor[0]
                    vetor[2]
004
                   006
         005
                             007
800
          009
                   010
                             011
                  HD
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam;<mark>(</mark>i++)) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                              O computador calcula:
                              i+1
                              = 2 + 1
                              = 3
```

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
              Memória
                    i
vetor
           tam
   0.04
                   002
                             003
000
          001
vetor[0]
          vetor[1]
                    vetor[2]
004
                   006
         005
                             007
800
          009
                   010
                             011
                  HD
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam;<mark>(</mark>i++)) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
                              O computador calcula:
                              i+1
                              = 2 + 1
```

```
Monitor / Teclado
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
              Memória
vetor
          tam
  0.04
000
                    002
                              003
         001
                    vetor 21
vetor[0]
          vetor[1]
                    006
004
         005
                              007
800
          009
                   01/0
                              011
                   ΗĎ
                                      54
```

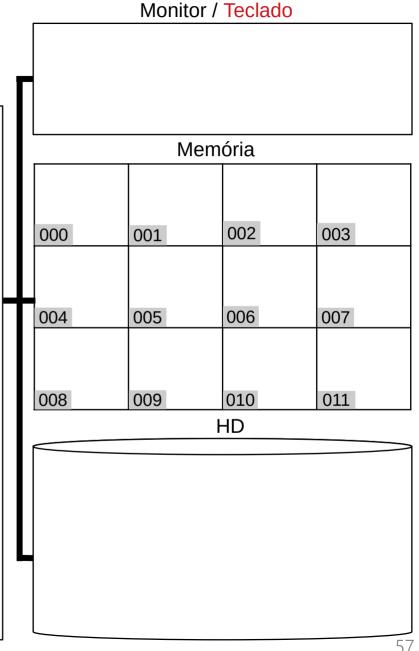
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; (i < tam;) i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0:
                             O computador calcula:
                             i < 3
                             = 3 < 3
                             = O(falso)
```

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
              Memória
                    i
vetor
          tam
  0.04
         001
                   002
                             003
000
vetor[0]
          vetor[1]
                    vetor[2]
                   006
004
         005
                             007
800
         009
                   010
                             011
                  HD
```

```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
              Memória
                   i
vetor
          tam
  0.04
000
         001
                   002
                            003
004
                   006
         005
                            007
800
         009
                  010
                             011
                  HD
```

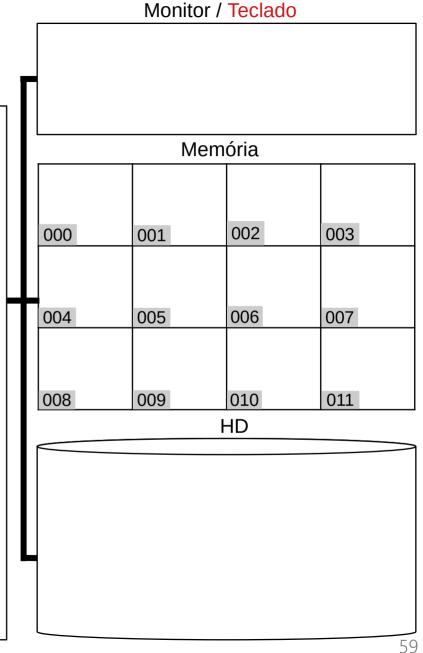
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d", &tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   for (i = 0; i < tam; i++) {
      printf("vetor[%d] = %d\n",i,vetor[i]);
   free (vetor);
   return 0;
```



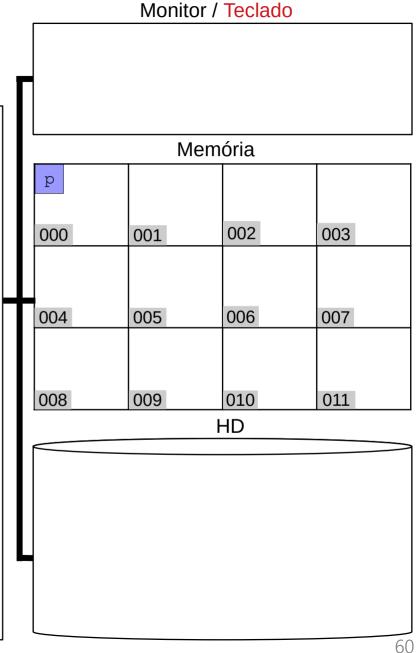
Outros assuntos Exemplo

```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
};
int main() {
   struct ponto *p;
   p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
   p->y = 6.28;
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```

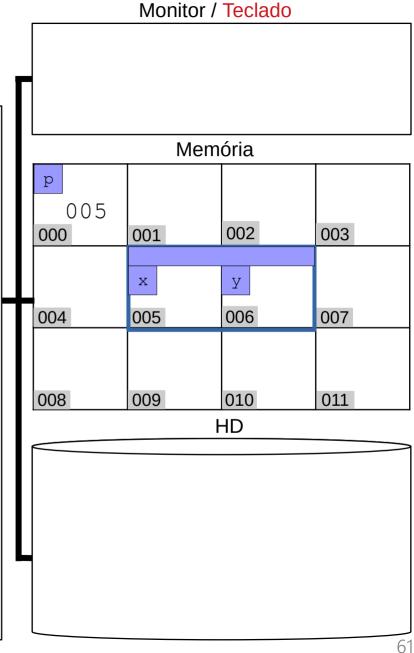
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
int main() {
   struct ponto *p;
   p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
   p->y = 6.28;
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```



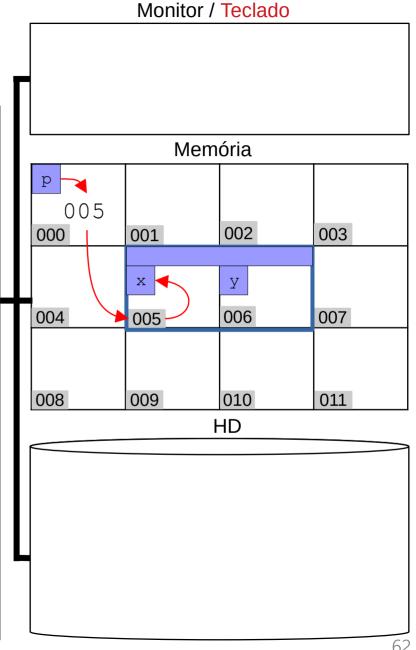
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
int main() {
  struct ponto *p;
  p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
   p->y = 6.28;
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```



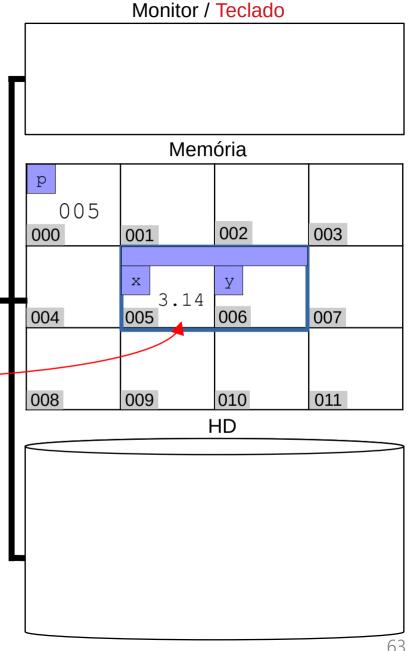
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
int main() {
struct ponto *p;
  p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
  p->y = 6.28;
  printf("x = %.2f y = %.2f\n",p->x,p->y);
  free(p);
   return 0;
```



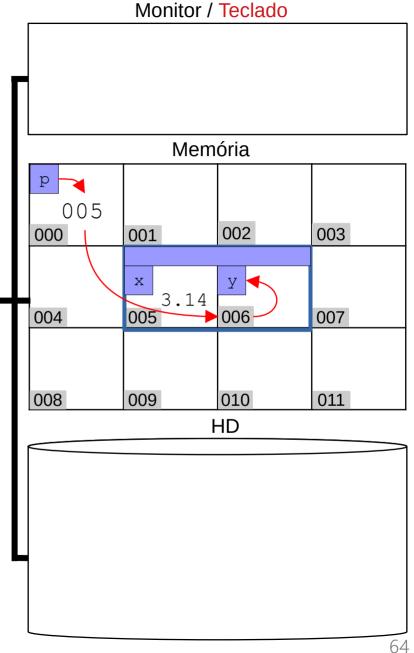
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
int main() {
   struct ponto *p;
  p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
   p->y = 6.28;
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```



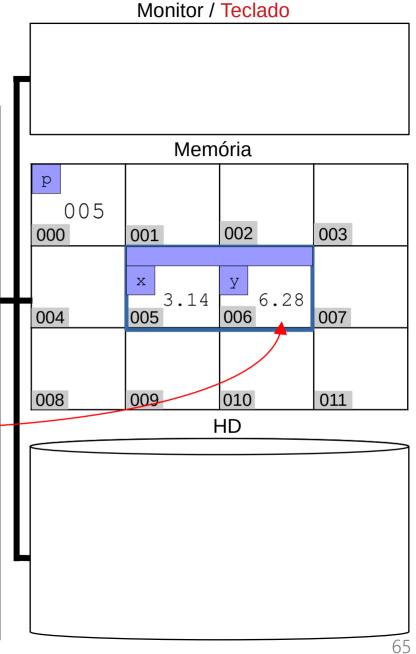
```
#include<stdio.h>
#include<stdlib.h>
                                                                          Memória
struct ponto {
   float x;
   float y;
                                                               005
                                                                              002
                                                             000
                                                                      001
int main() {
   struct ponto *p;
                                                                      X
  p = (struct ponto *) malloc(sizeof(struct ponto));
                                                                         3.14
   (*p).x = (3.14)
                                                             004
                                                                              006
                                                                      005
  p->y = 6.28;
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
                                                             800
                                                                      009
                                                                              010
                                                                              HD
```



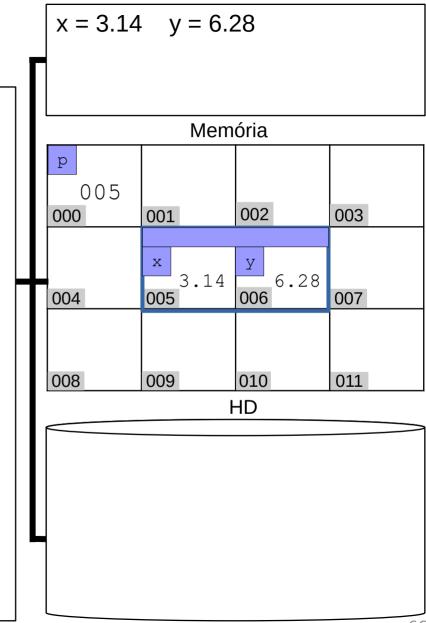
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
int main() {
   struct ponto *p;
   p = (struct ponto *) malloc(sizeof(struct ponto));
  (*p).x = 3.14;
  p->y = 6.28;
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```



```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
int main() {
   struct ponto *p;
   p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
   p - y = (6.28)
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```



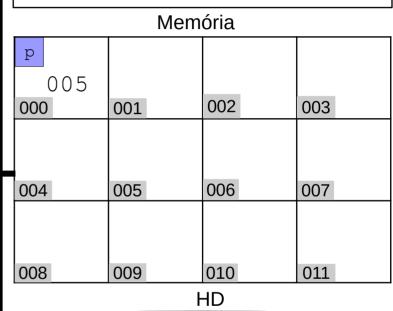
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
int main() {
   struct ponto *p;
  p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
  p->y = 6.28;
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```



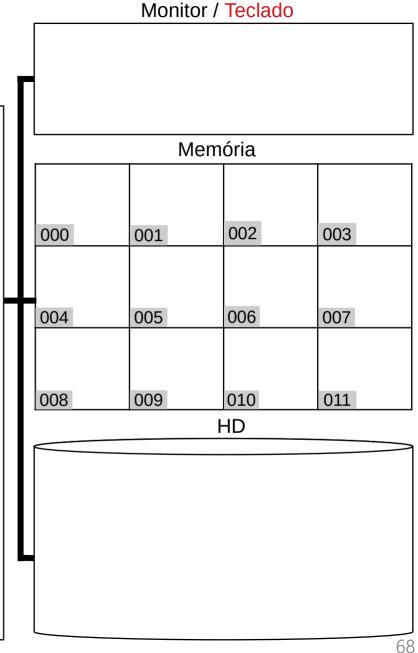
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
  float y;
int main() {
   struct ponto *p;
  p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
  p->y = 6.28;
  printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```

$\frac{\text{Monitor / Teclado}}{= 3.14 \quad \text{v} = 6.28}$





```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
int main() {
   struct ponto *p;
   p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
   p->y = 6.28;
   printf("x = %.2f y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
```



- Definição
- Exemplo

- Definição
 - Aloca (reserva) memória durante a execução do programa
 - Flexibilidade para usar a memória de acordo com a demanda
- Exemplo

Definição

Exemplo

```
include<stdio.h>
#include<stdlib.h>
int main() {
    int *p;
    printf("p = %p\n",p);
    p = (int *) malloc(sizeof(int));
    printf("p = %p\n",p);
    *p = 30;
    printf("*p = %d\n",*p);
    free(p);
    return 0;
}
```

- Definição
- Exemplo



Computação Eletrônica Juliano M. Iyoda jmi@cin.ufpe.br

