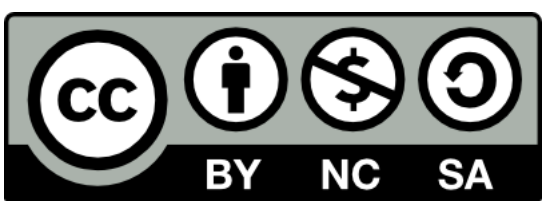
Centro de Informática
UFPE

Outros assuntos

malloc()

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



Outros assuntos

malloc()

- Definição
- Exemplo

Outros assuntos

malloc()

- 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 p apontar para ela.

`#include<stdio.h>`
`int main() {`
`  int *p;`
`  *p = 30; // ERRO!`
`  return 0;`
`}`
	`#include<stdio.h>`
	`#include<stdlib.h>`
	`int main() {`
	`  int *p;`
	`  p = (int *) malloc(sizeof(int));`
	`  *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 p apontar para ela.

`#include<stdio.h>`
`int main() {`
`  int *p;`
`  *p = 30; // ERRO!`
`  return 0;`
`}`
	`#include<stdio.h>`
	`#include<stdlib.h>`
	`int main() {`
	`  int *p;`
	`  p = (int *) malloc(sizeof(int));`
	`  *p = 30; // OK!`
	`  free(p);`
	`  return 0;`
	`}`

Outros assuntos

malloc()

- Definição
- Exemplo

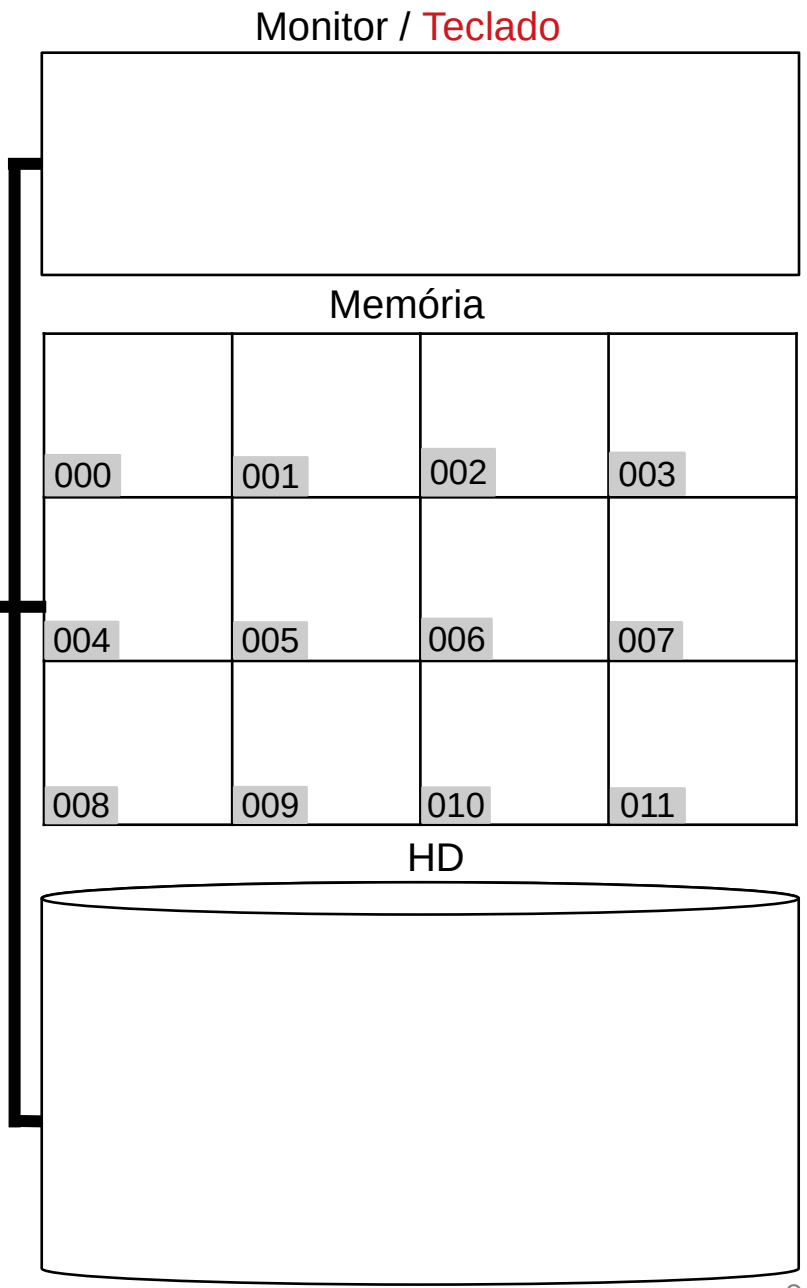
Outros assuntos

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;
}
```

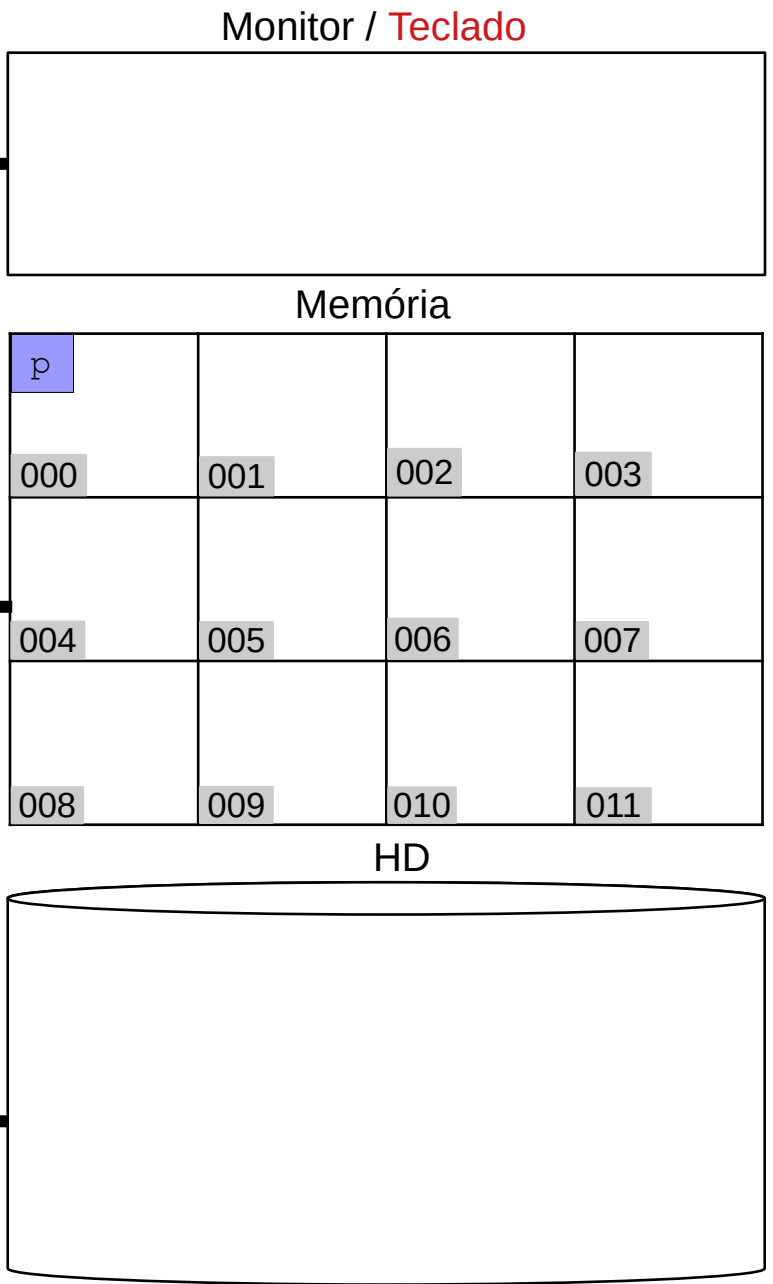
Execução

```
#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;
}
```



Execução

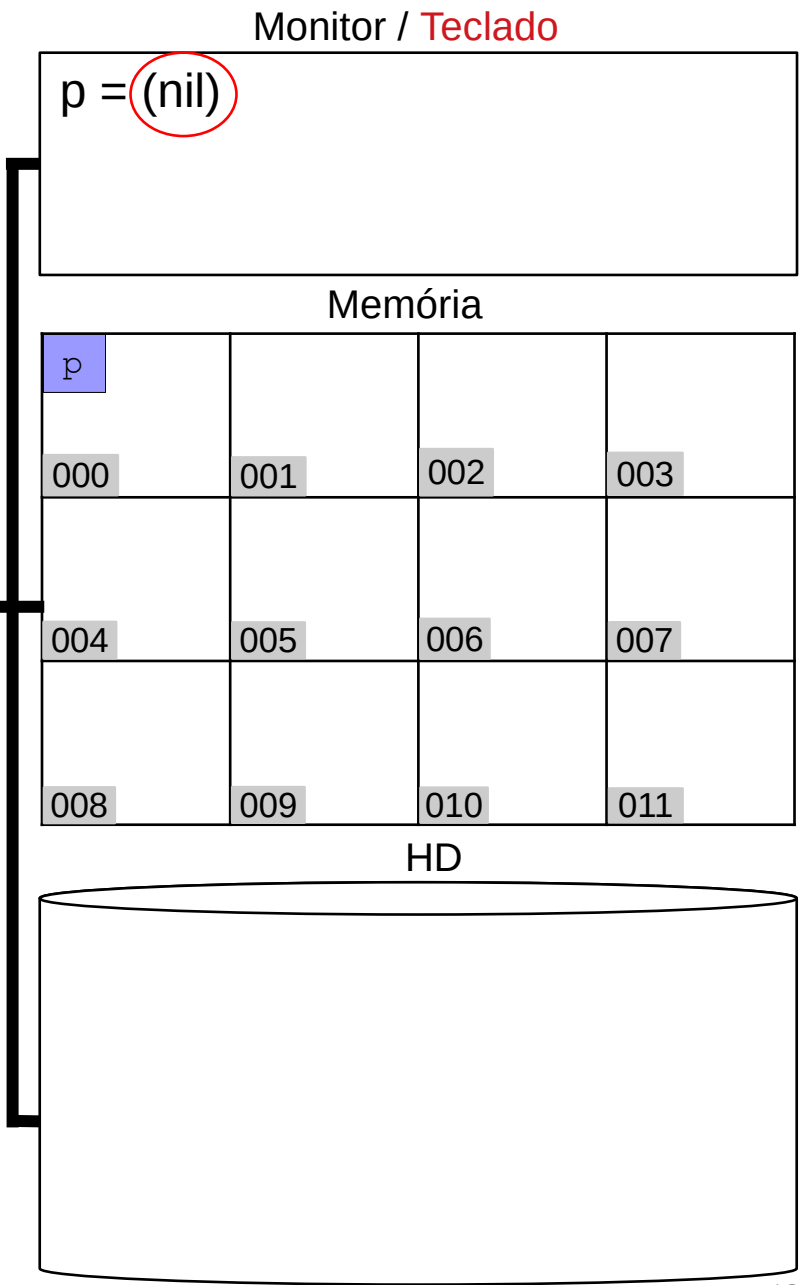
```
#include<stdio.h>
#include<stdlib.h>
int main() {
    int *p;
    printf("p = %p\n",p);
    p = (int *) malloc(sizeof(int));
    printf("p = %p\n",p);
    *p = 30;
    printf("*p = %d\n",*p);
    free(p);
    return 0;
}
```



Execução

```
#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;
}
```

Pode aparecer (nil) ou outro número arbitrário. Independente disso, é um endereço que não está alocado (reservado) para usarmos.

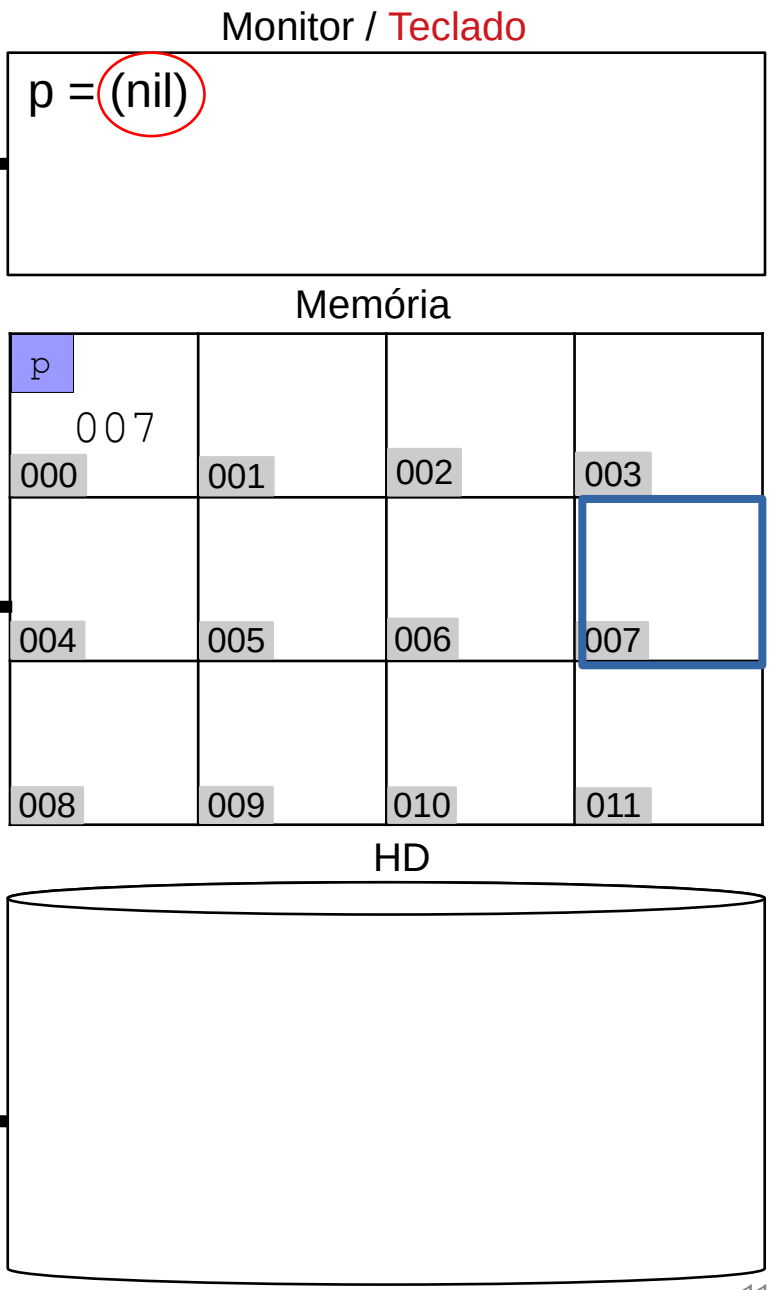


Execução

```
#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;
}
```

malloc() reserva um local na memória do tamanho de um inteiro (sizeof(int)). E retorna esse endereço.

Assumimos que malloc() reservou no endereço 007.



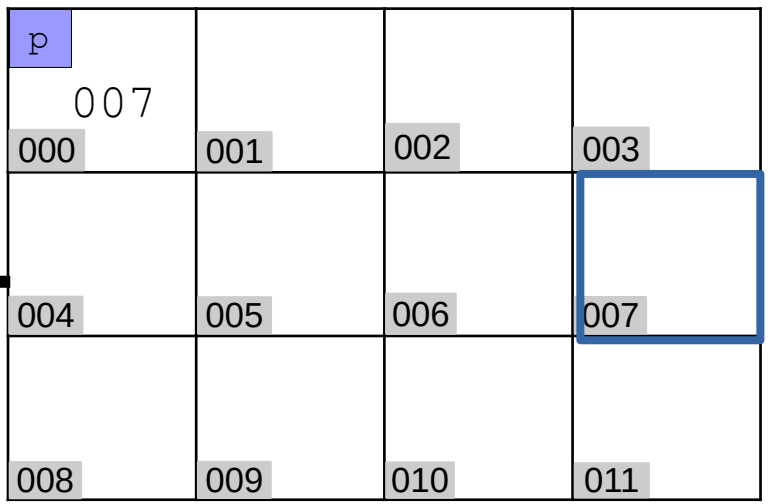
Execução

```
#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;
}
```

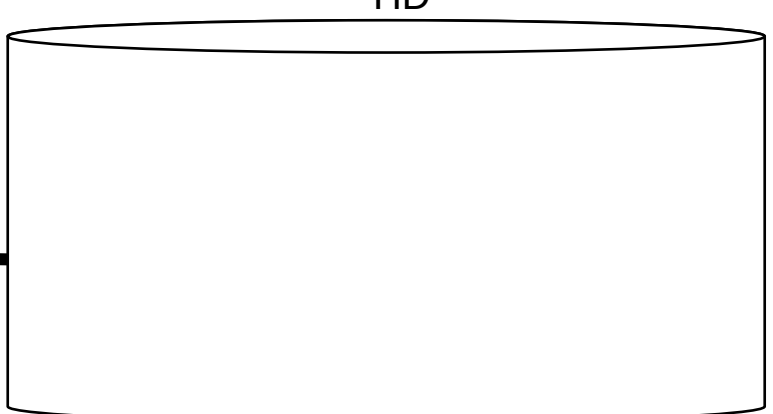
Monitor / Teclado

```
p = (nil)
p = 007
```

Memória

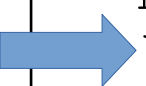


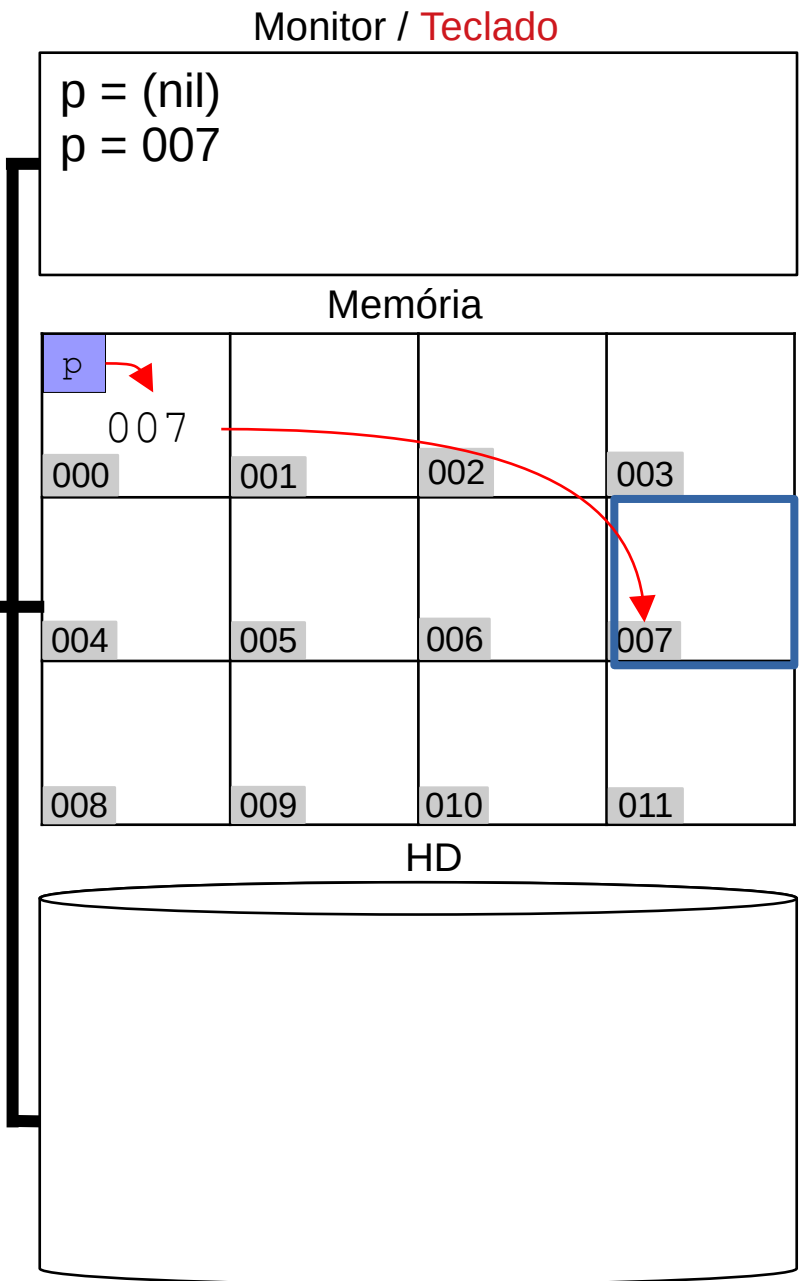
HD



Execução

```
#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;
}
```





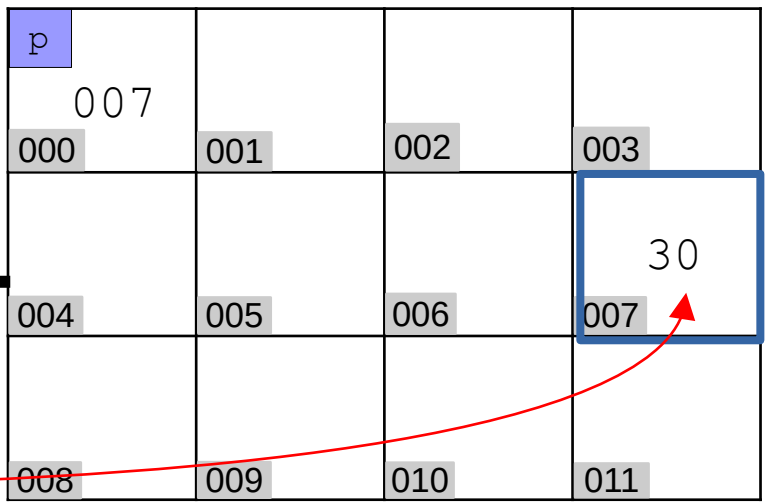
Execução

```
#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;
}
```

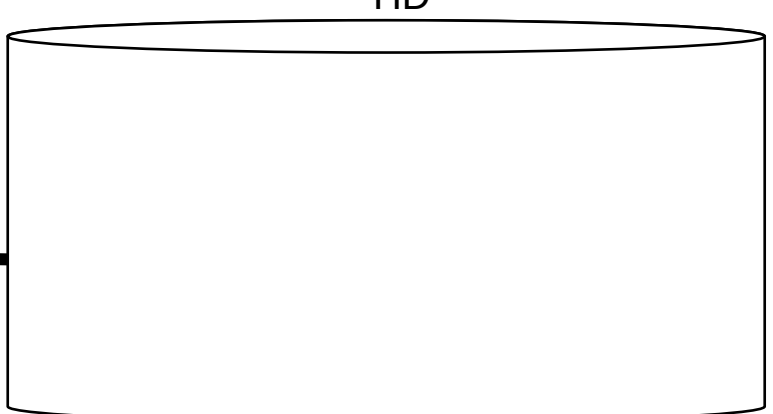
Monitor / Teclado

```
p = (nil)
p = 007
```

Memória



HD



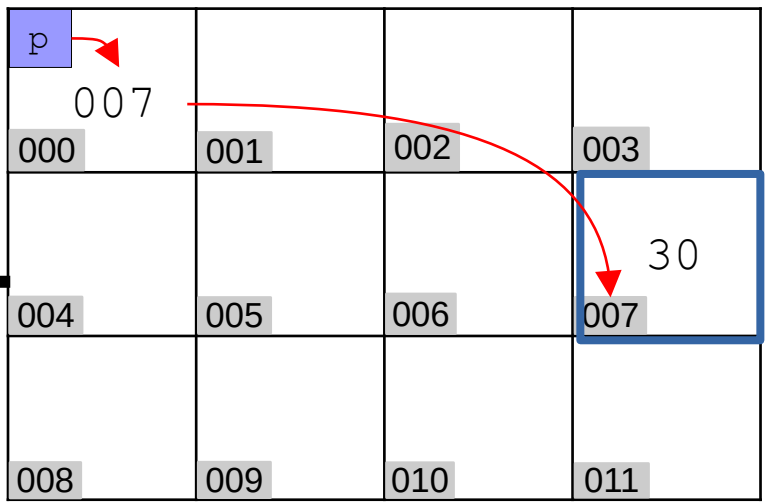
Execução

```
#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;
}
```

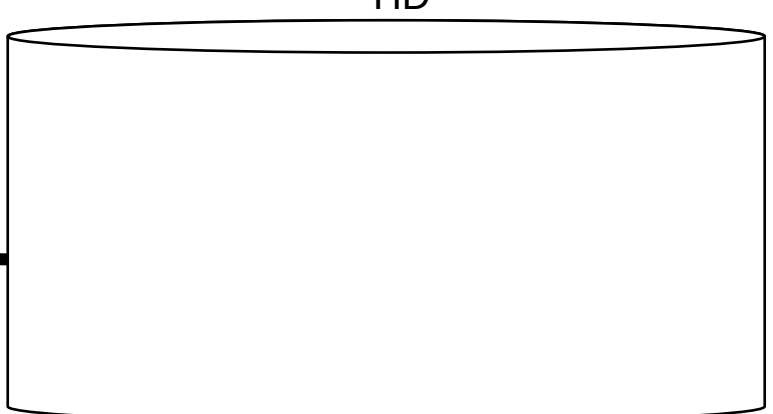
Monitor / Teclado

```
p = (nil)
p = 007
```

Memória



HD



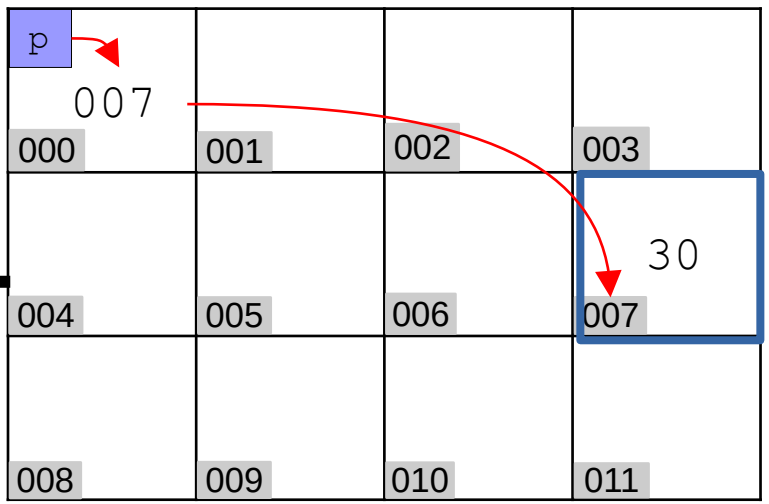
Execução

```
#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;
}
```

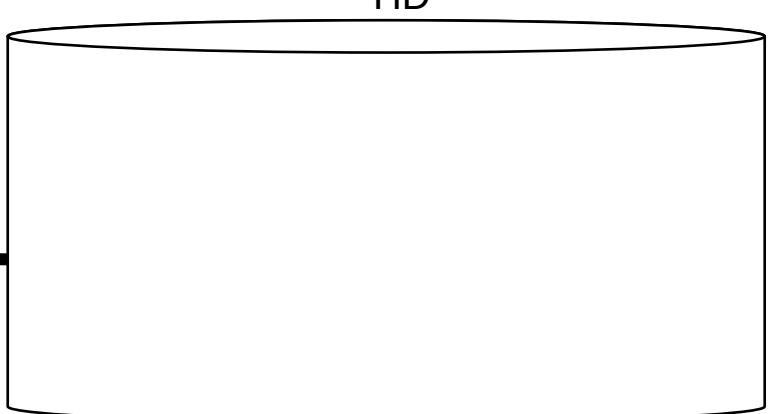
Monitor / Teclado

```
p = (nil)
p = 007
*p = 30
```

Memória



HD



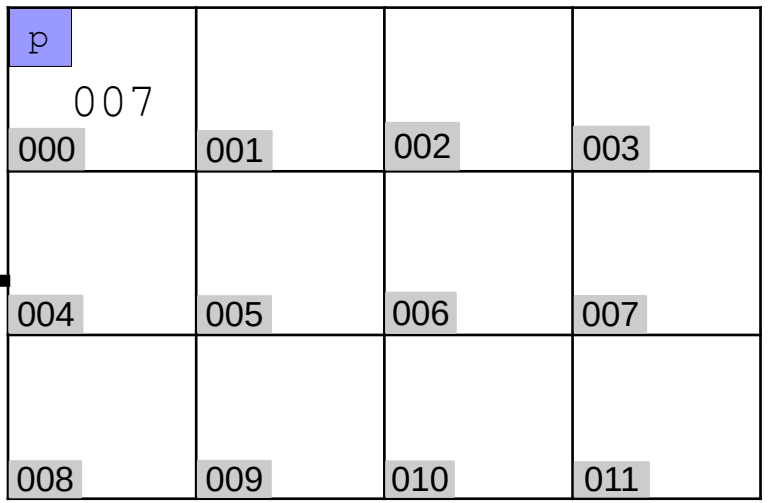
Execução

```
#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;
}
```

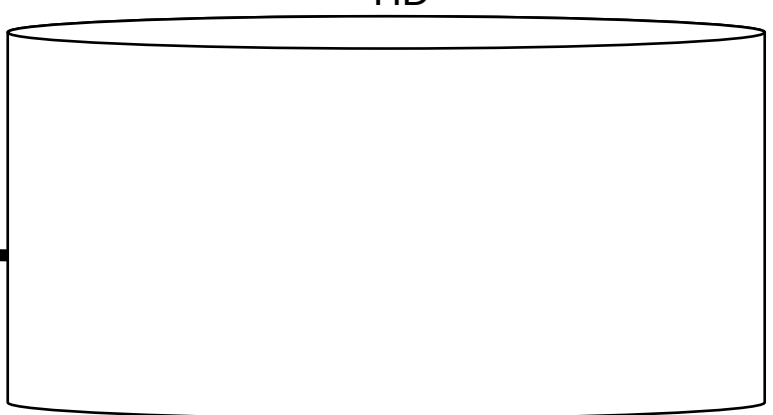
Monitor / Teclado

```
p = (nil)
p = 007
*p = 30
```

Memória

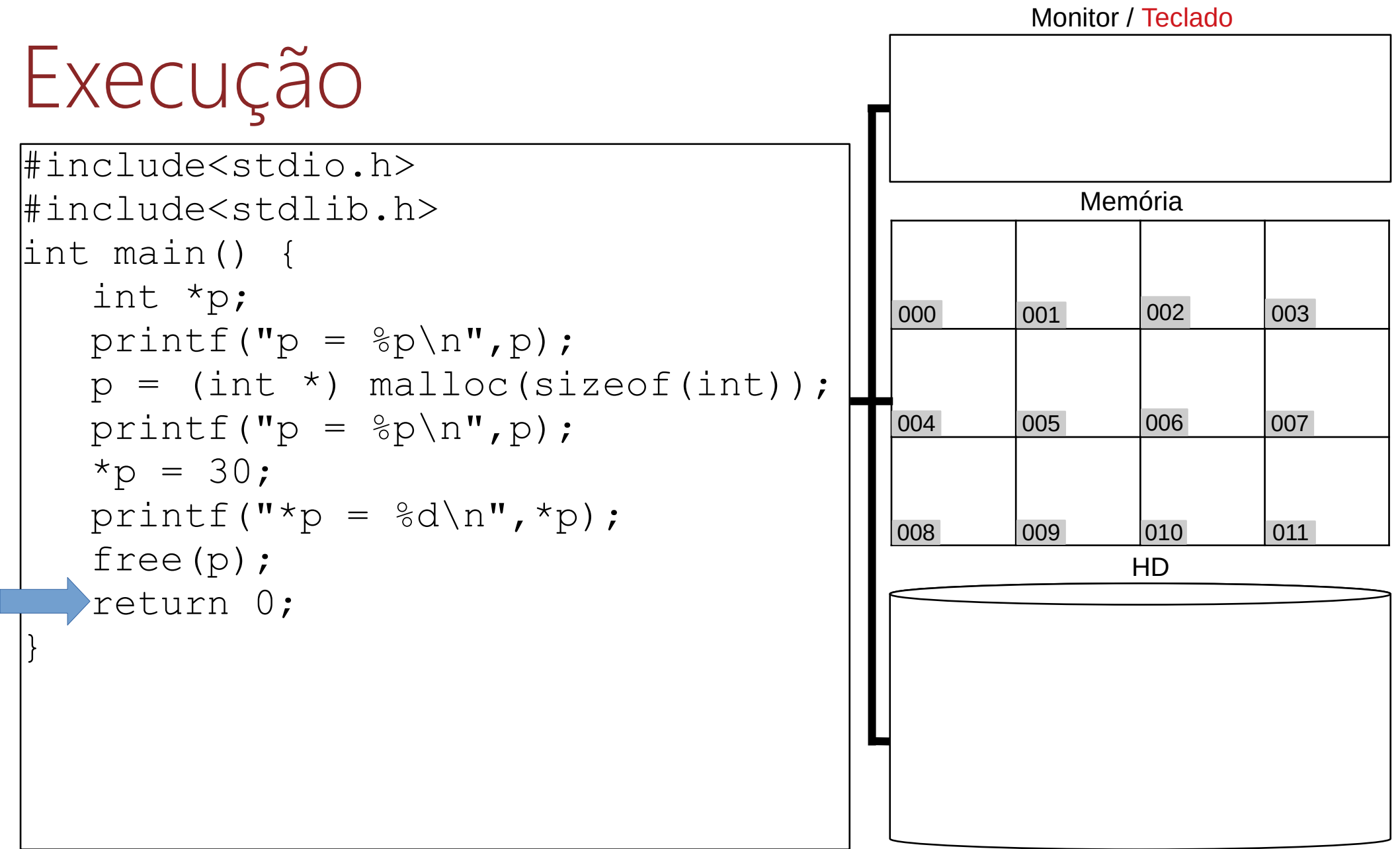


HD



Execução

```
#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;
}
```



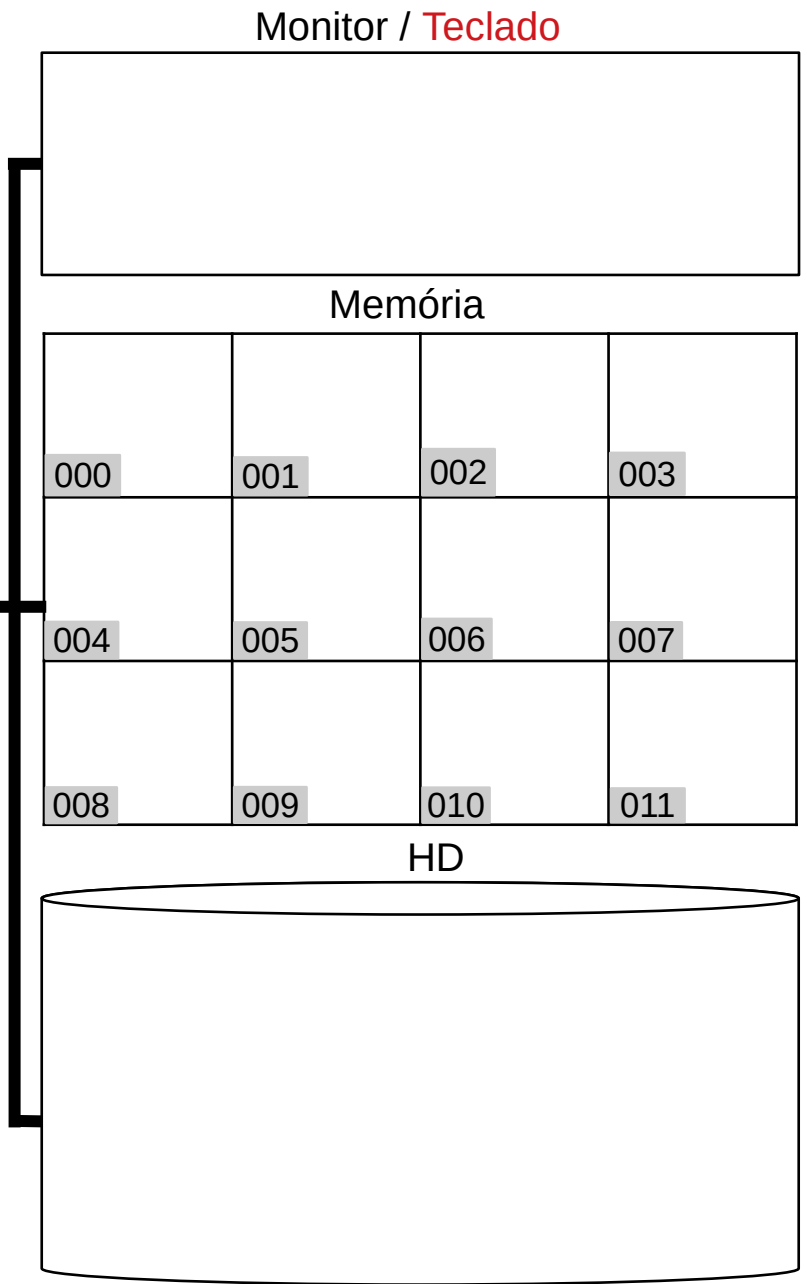
Outros assuntos

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;
}
```

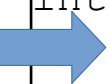
Execução

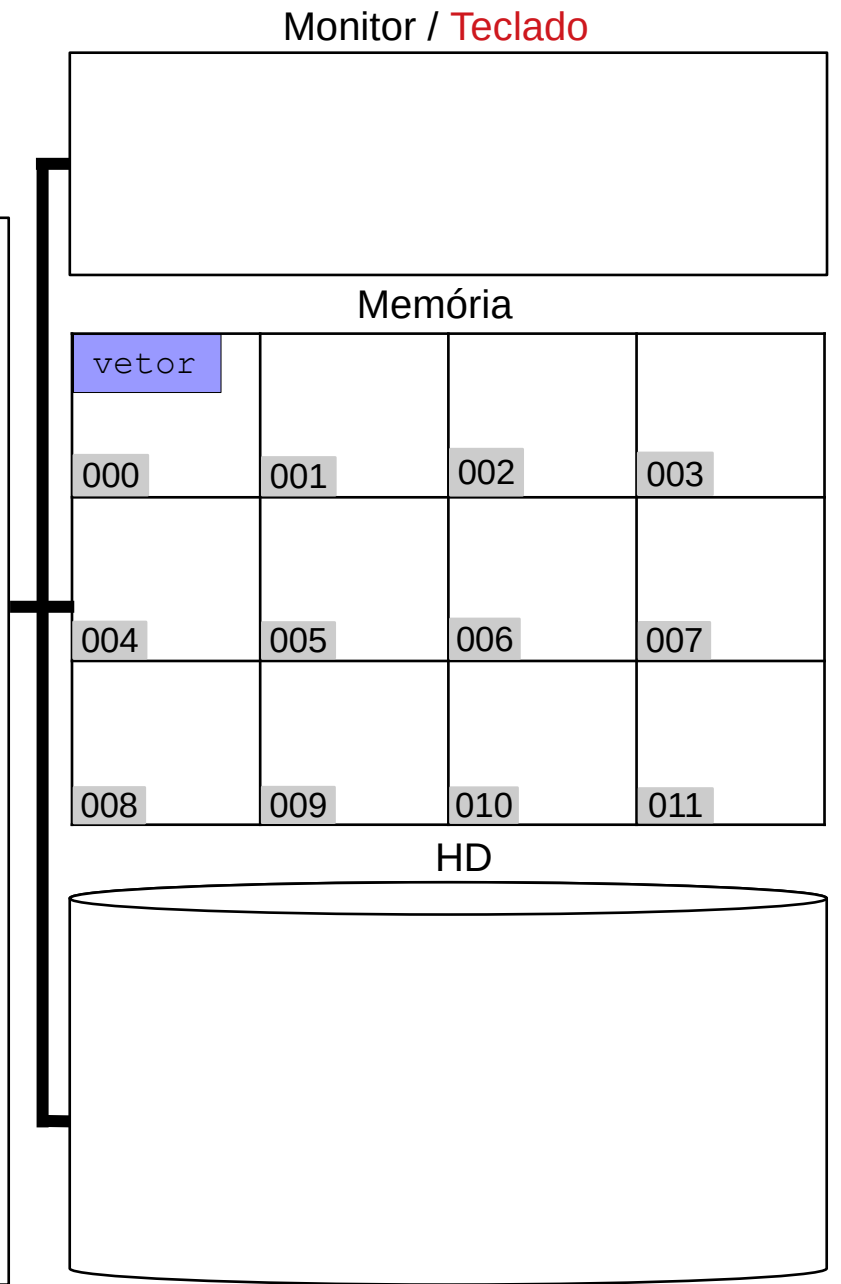
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   }
   for (i = 0; i < tam; i++) {
      printf("vetor[%d]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



Execução

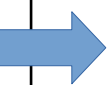
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   }
   for (i = 0; i < tam; i++) {
      printf("vetor[%d]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

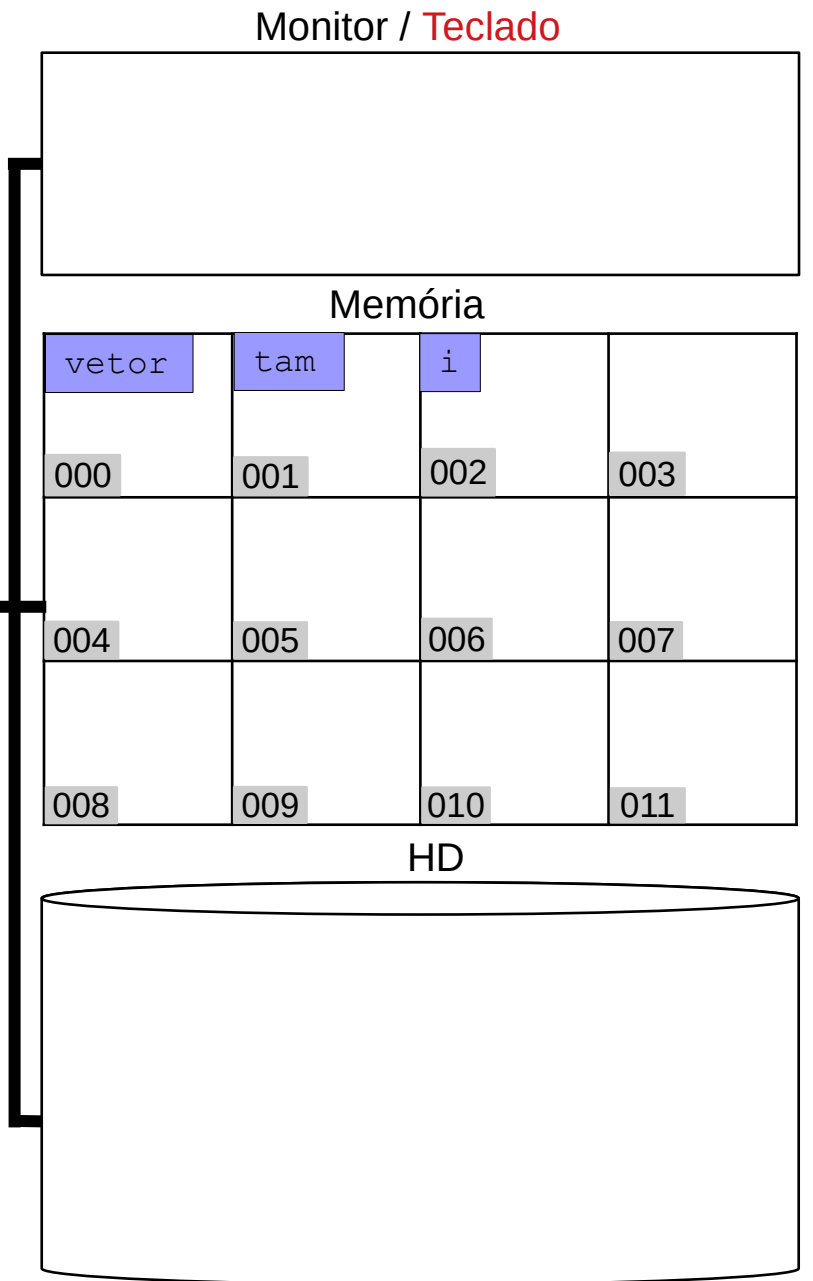




Execução

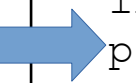
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   }
   for (i = 0; i < tam; i++) {
      printf("vetor[%d]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

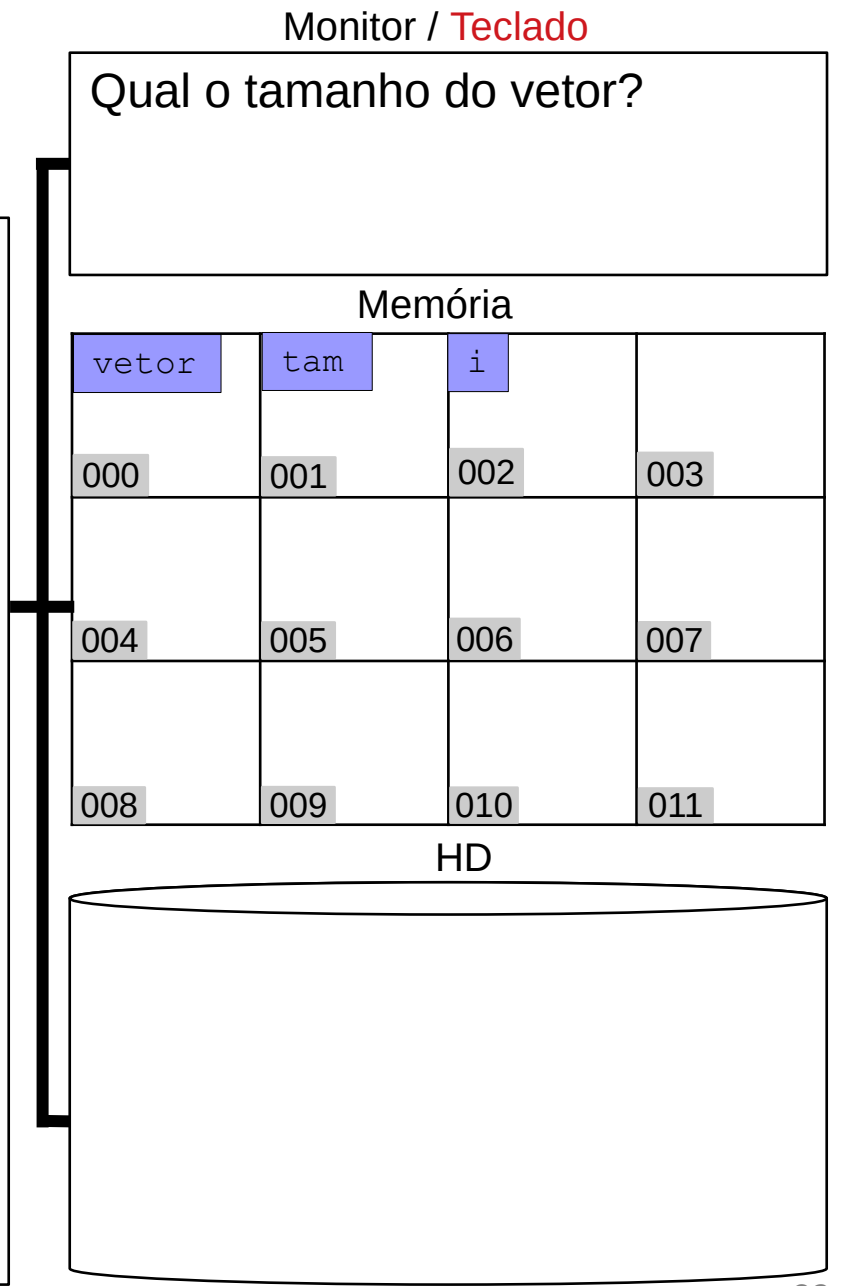




Execução

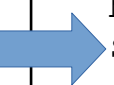
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   }
   for (i = 0; i < tam; i++) {
      printf("vetor[%d]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

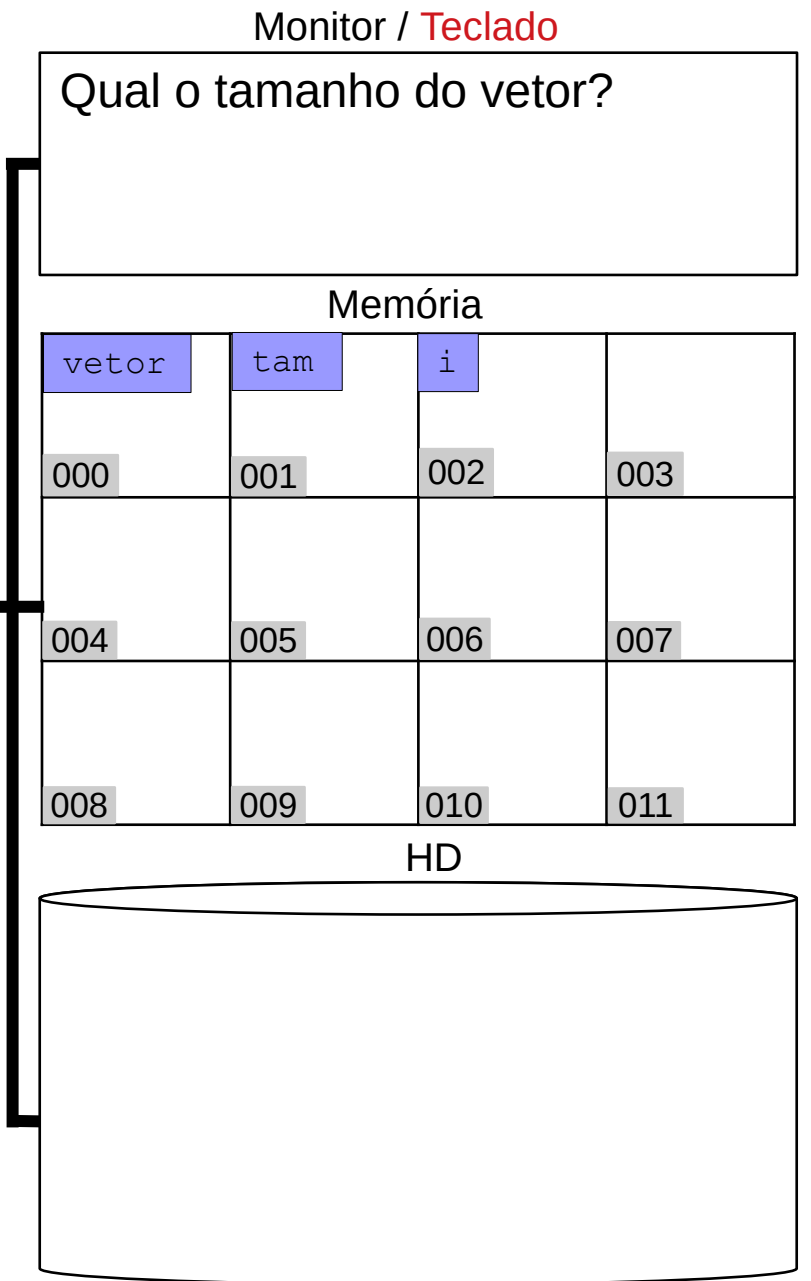




Execução

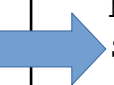
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   }
   for (i = 0; i < tam; i++) {
      printf("vetor[%d]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

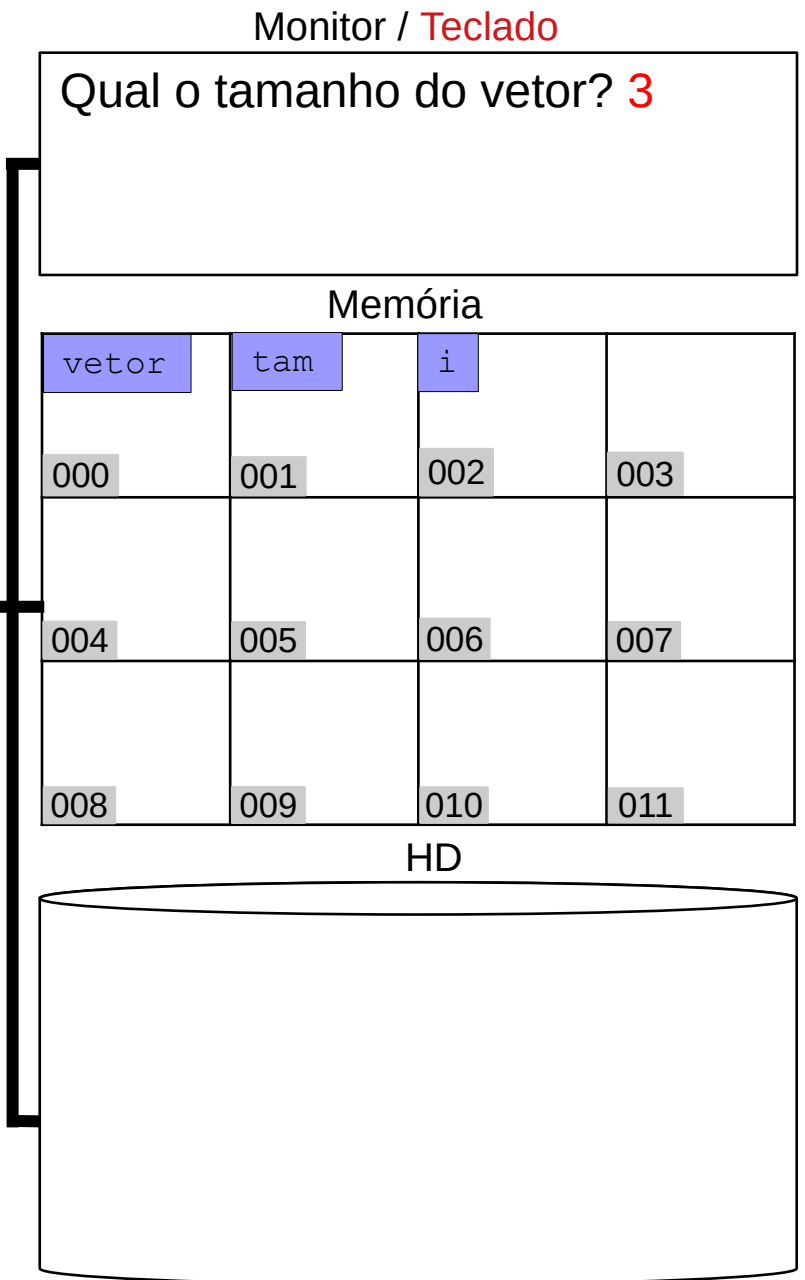




Execução

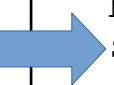
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   }
   for (i = 0; i < tam; i++) {
      printf("vetor[%d]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

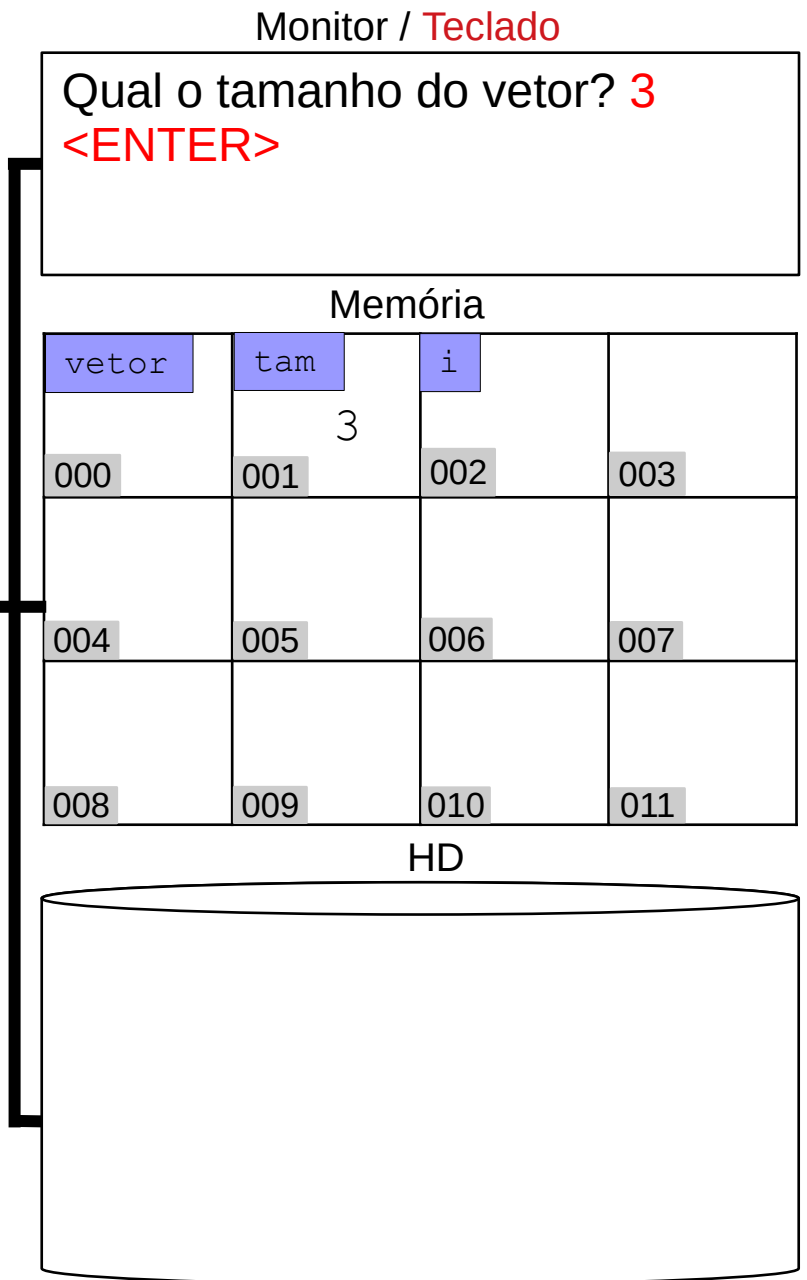




Execução

```
#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;
}
```





Execução

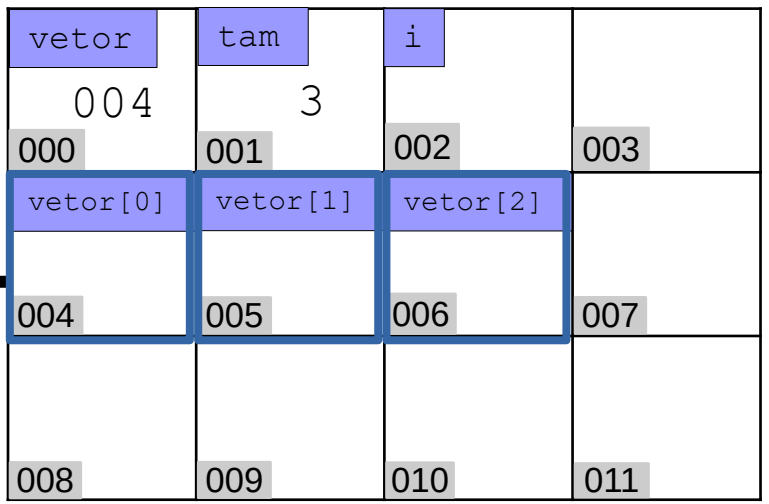
```
#include<stdio.h>
#include<stdlib.h>
int main() {
   int *vetor;
   int tam, i;
   printf("Qual o tamanho do vetor? ");
   scanf("%d",&tam);
   vetor = (int *) malloc(tam * sizeof(int));
   for (i = 0; i < tam; i++) {
      vetor[i] = i;
   }
   for (i = 0; i < tam; i++) {
      printf("vetor[%d]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

*malloc() reserva um local na memória do tamanho de 3 inteiros (3 * sizeof(int)). E retorna o endereço do primeiro elemento.*

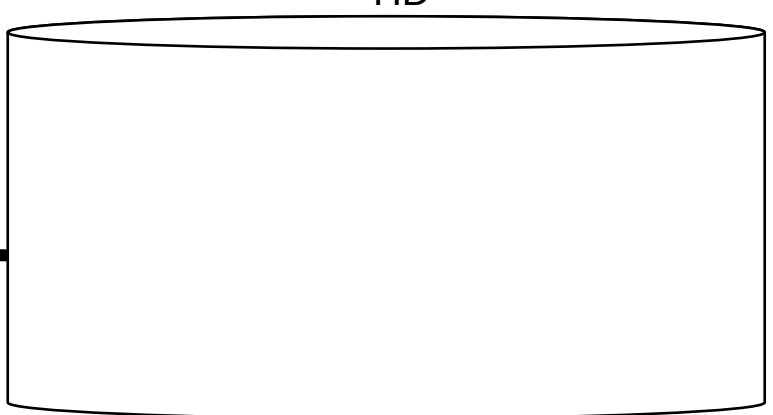
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



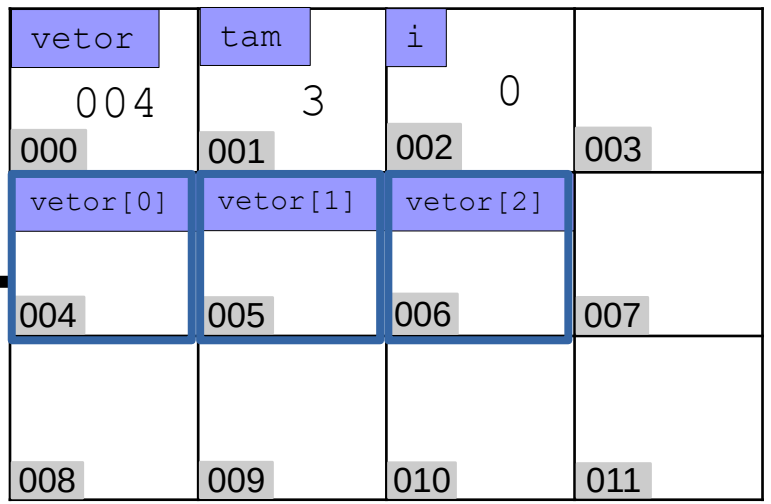
Execução

```
#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>

Memória



HD

Execução

```
#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

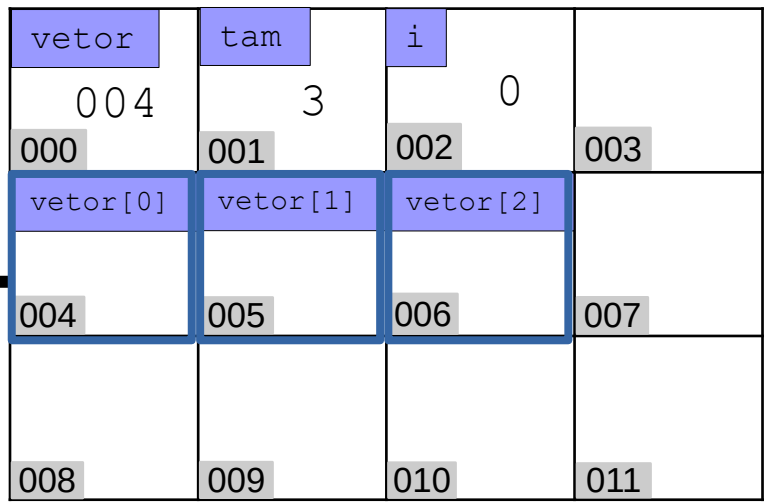
= 0 < 3

= 1 (verdadeiro)

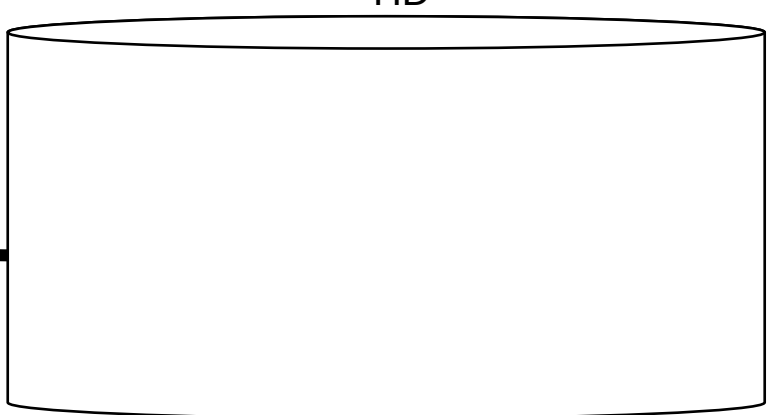
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



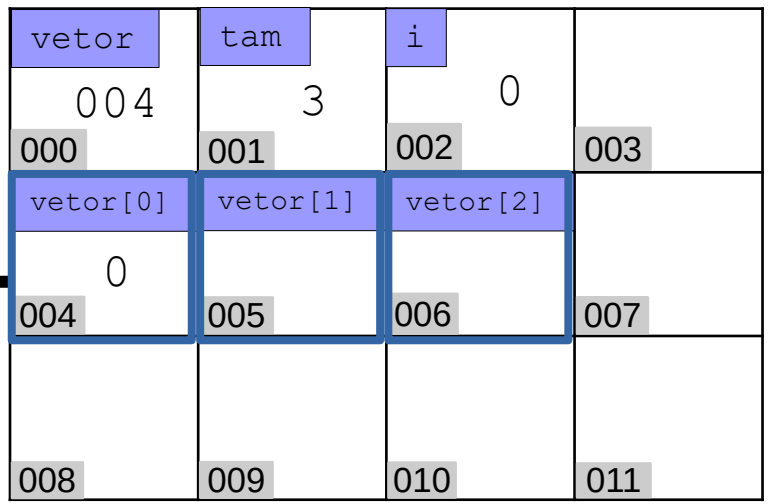
Execução

```
#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;
}
```

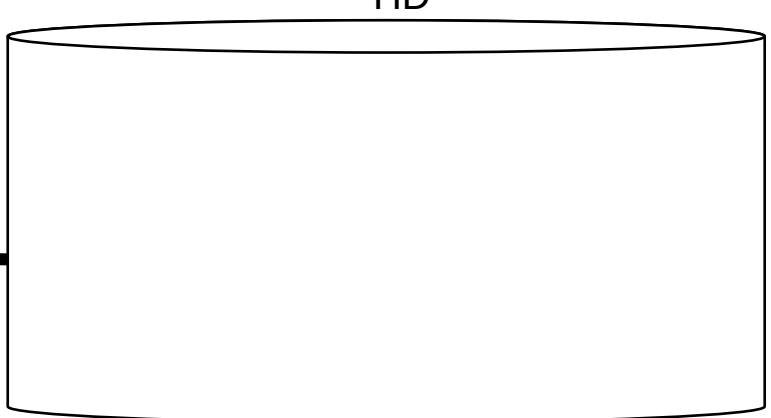
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

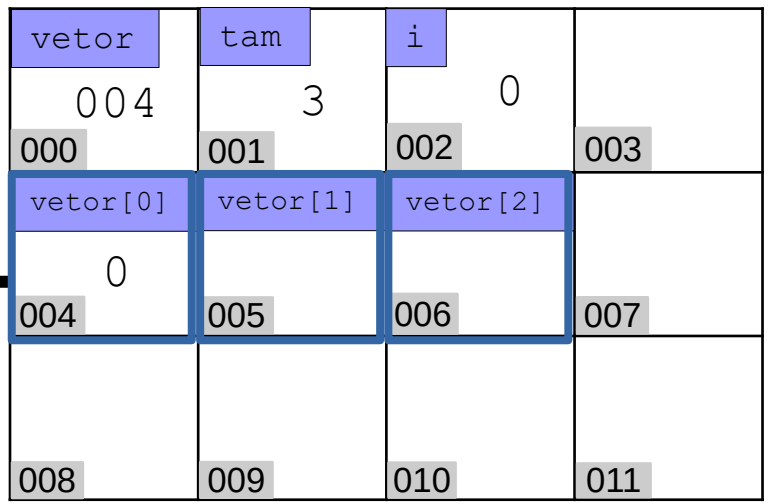
= 0 + 1

= 1

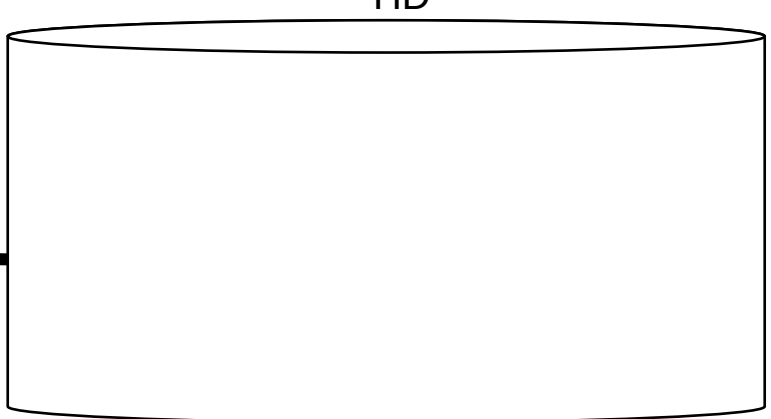
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

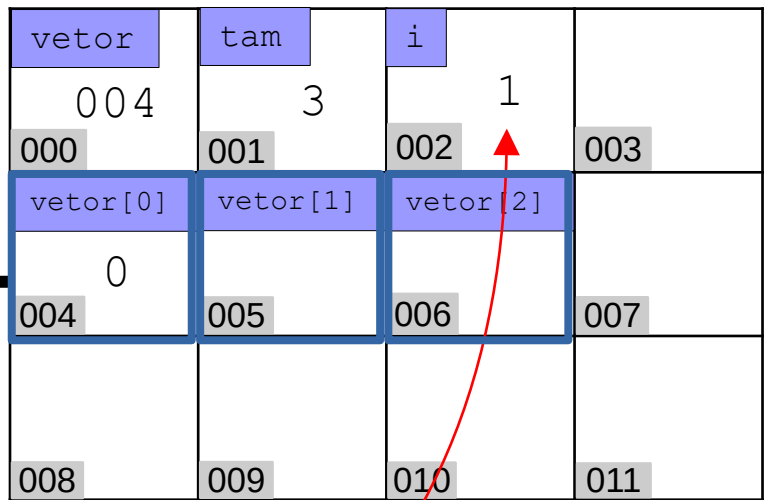
= 0 + 1

= 1

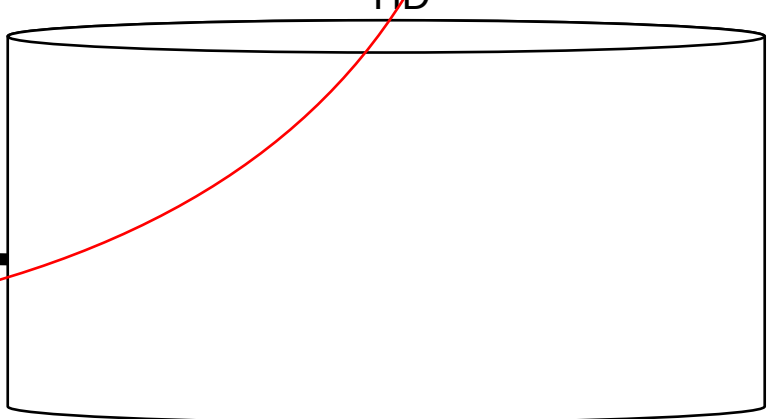
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

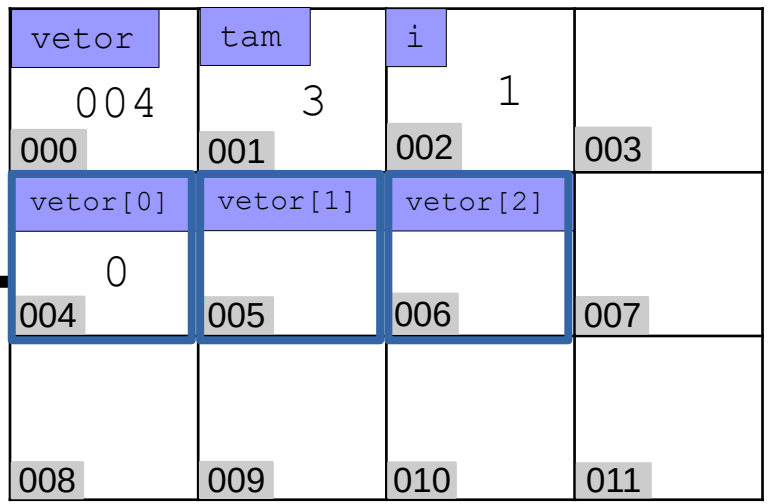
= 1 < 3

= 1 (verdadeiro)

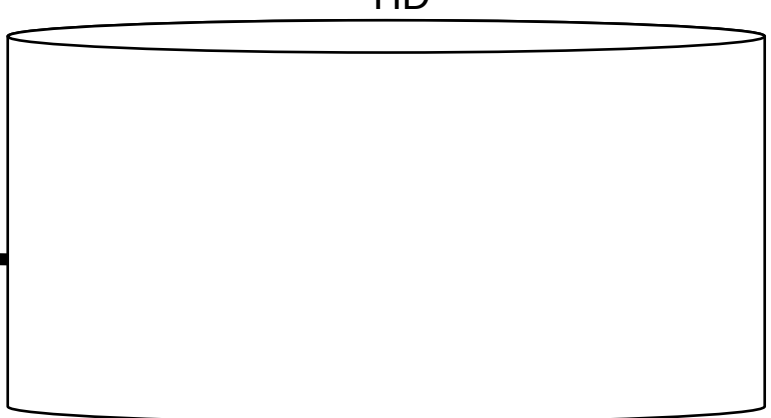
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



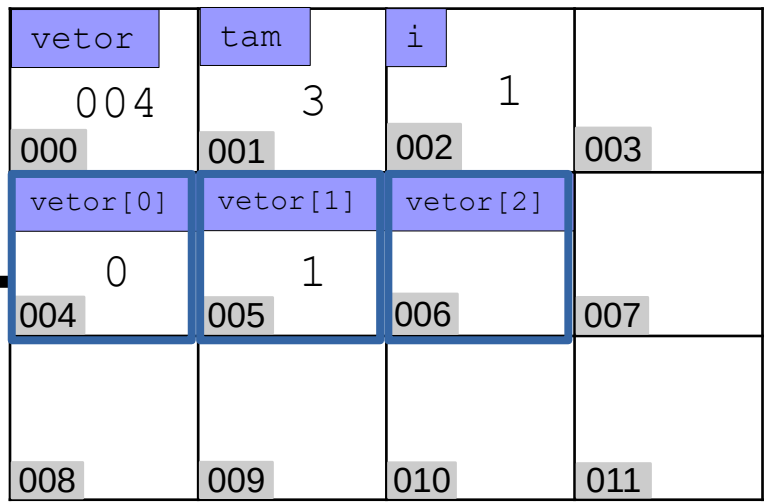
Execução

```
#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;
}
```

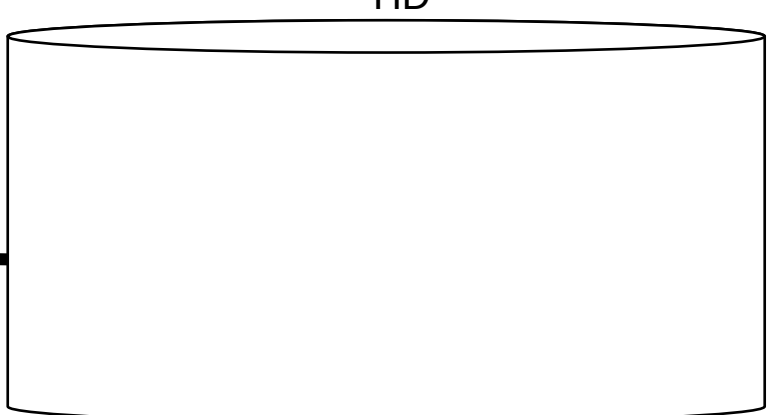
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

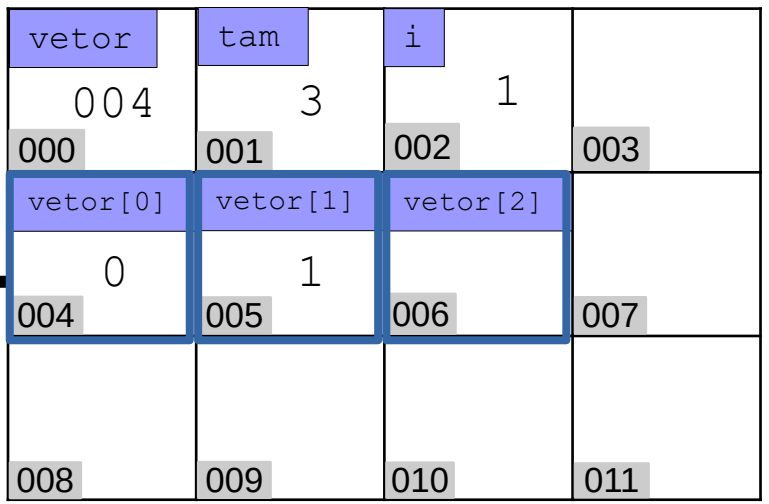
= 1 + 1

= 2

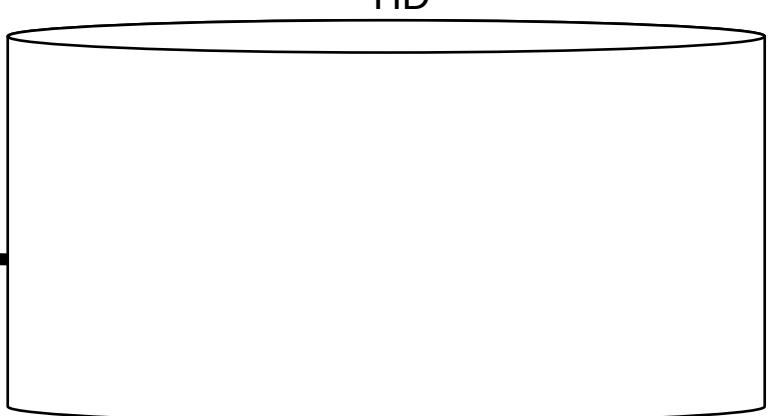
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

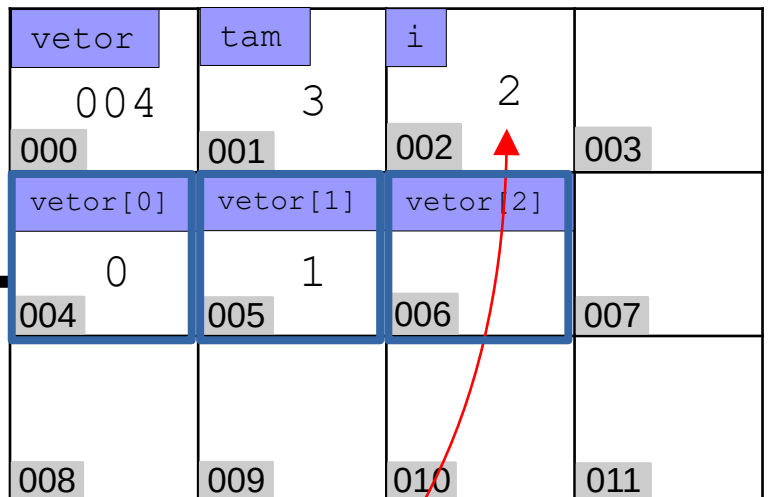
= 1 + 1

= 2

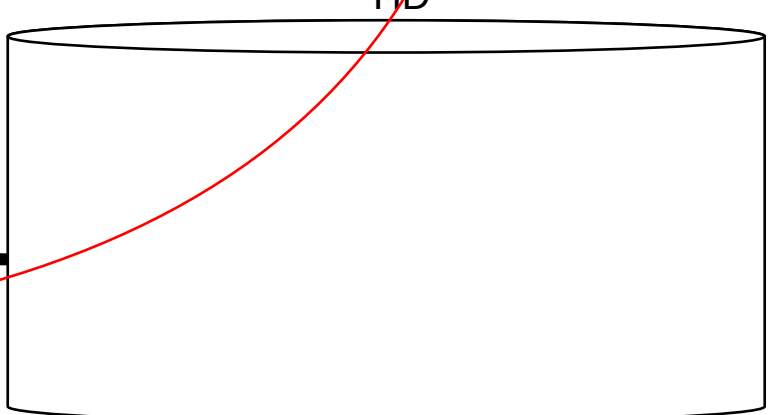
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

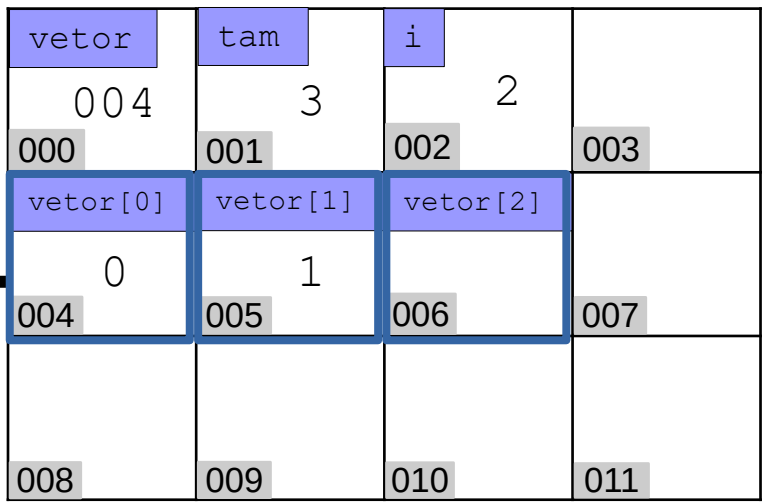
= 2 < 3

= 1 (verdadeiro)

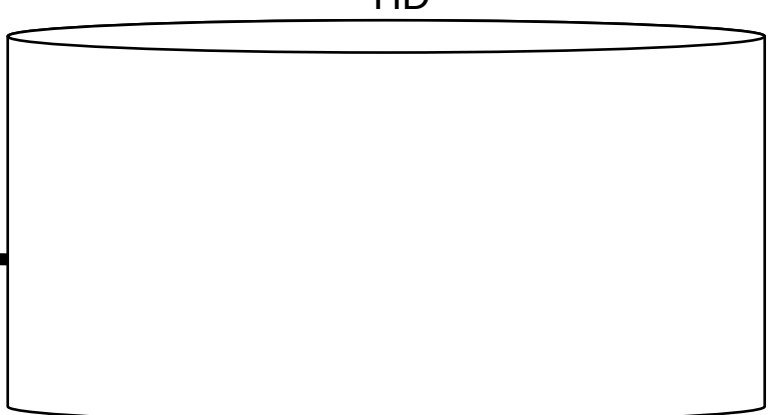
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



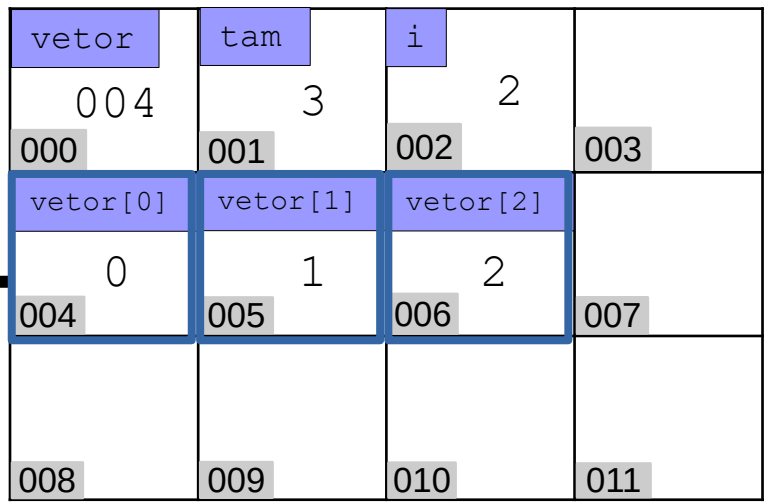
Execução

```
#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;
}
```

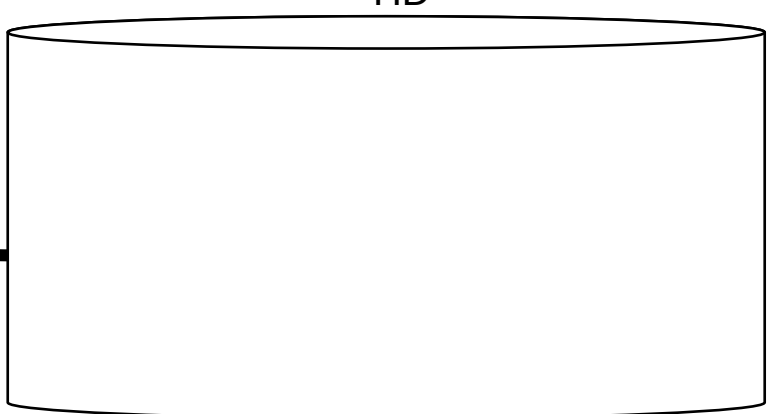
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

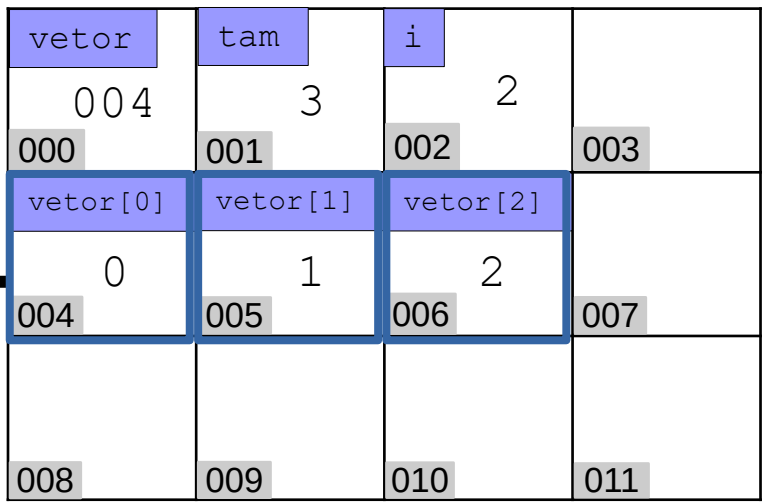
= 2 + 1

= 3

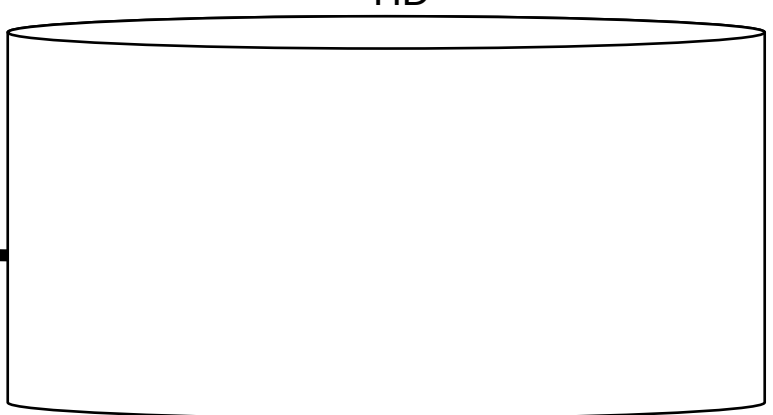
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

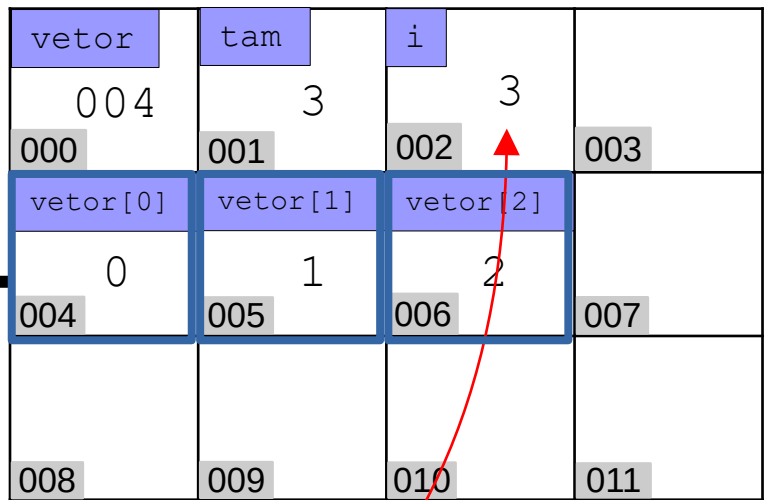
= 2 + 1

= 3

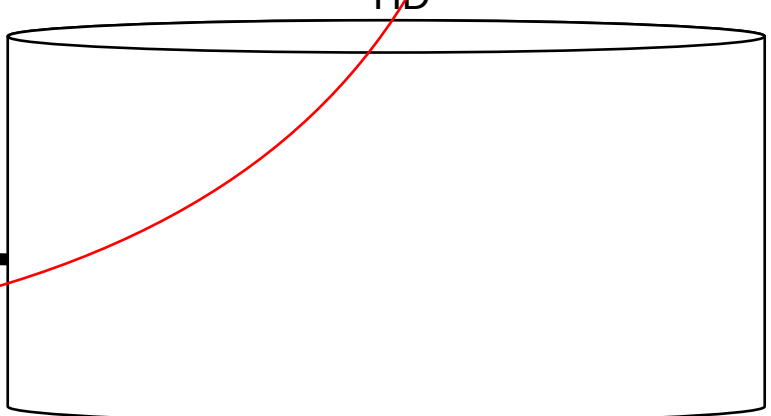
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

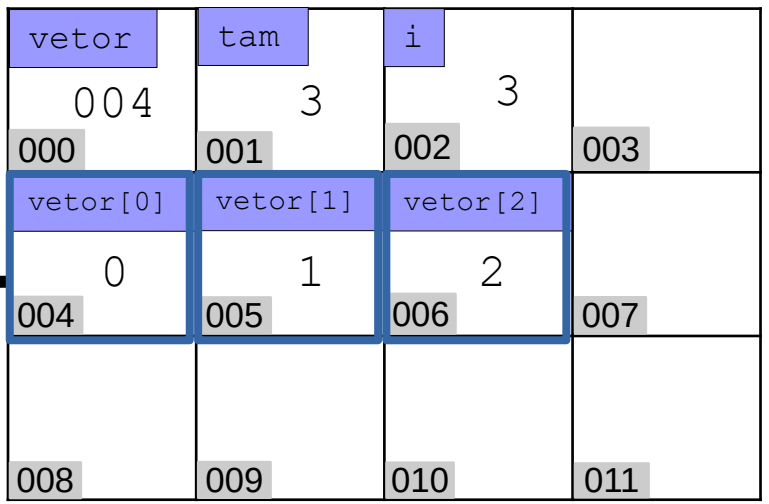
= 3 < 3

= 0 (falso)

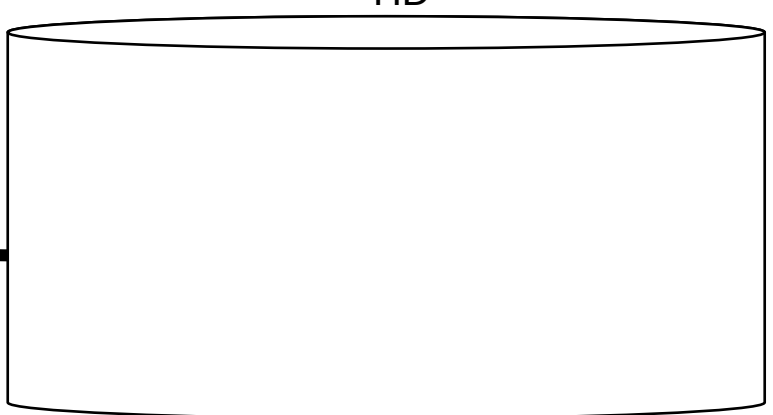
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



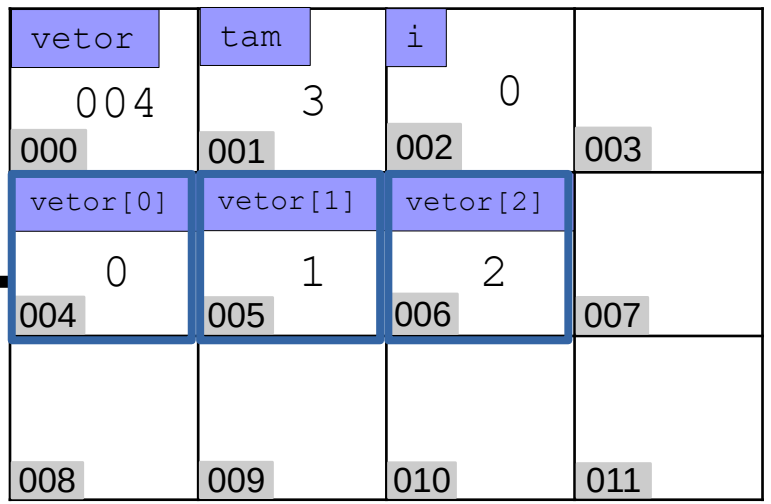
Execução

```
#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;
}
```

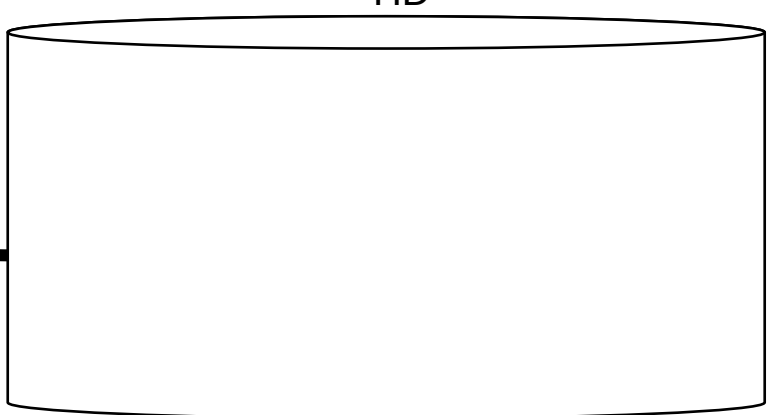
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



Execução

```
#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

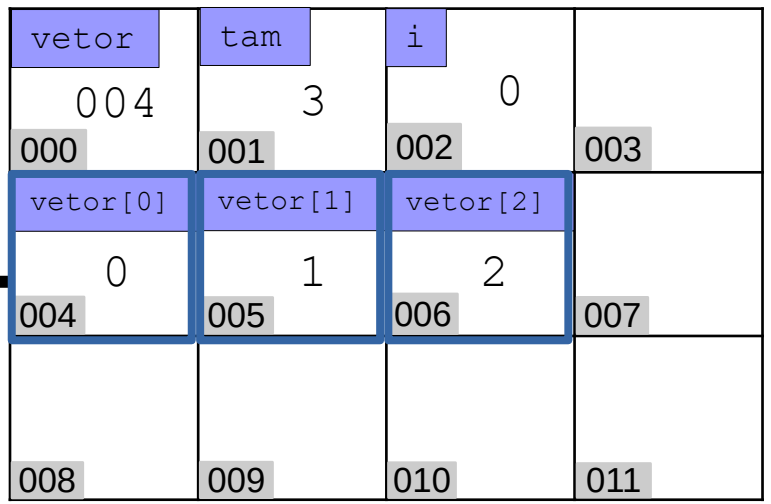
= 0 < 3

= 1 (verdadeiro)

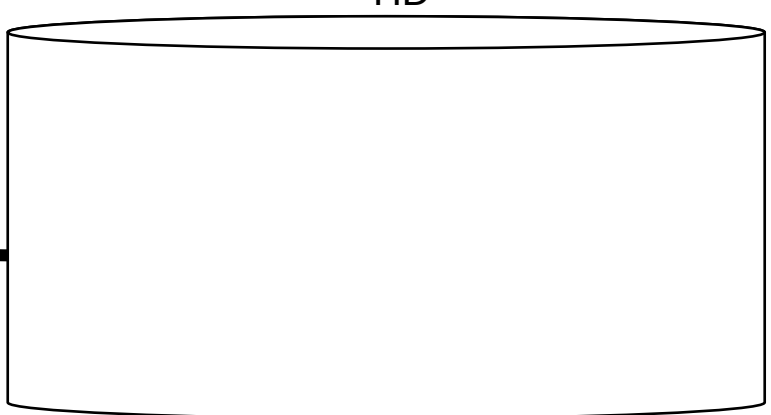
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>

Memória



HD



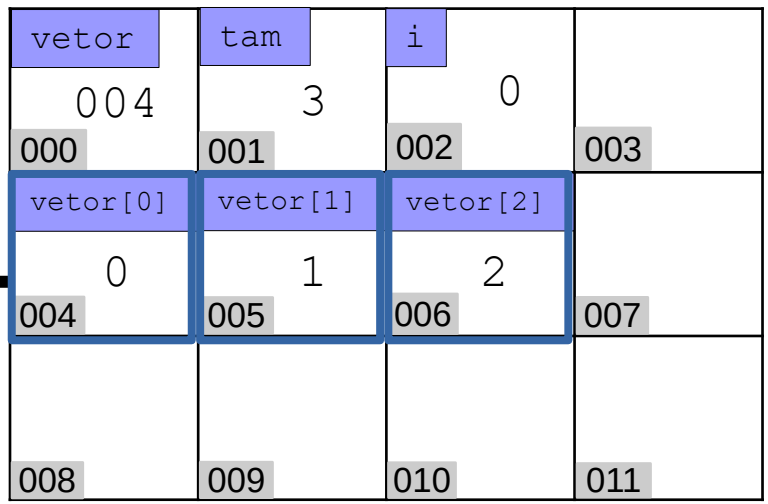
Execução

```
#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;
}
```

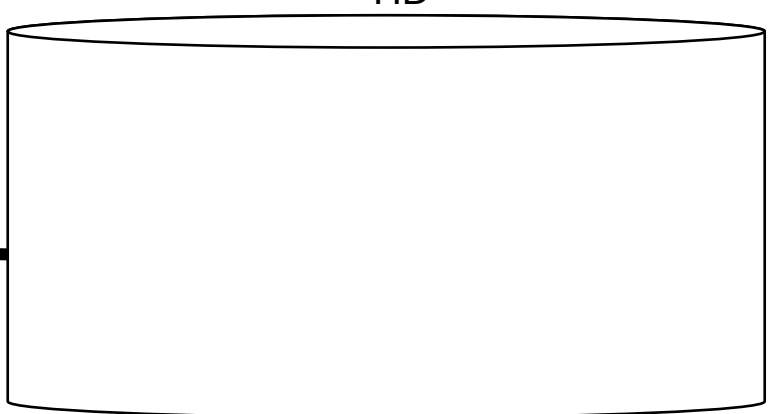
Monitor / Teclado

```
Qual o tamanho do vetor? 3
<ENTER>
vetor[0] = 0
```

Memória



HD



Execução

```
#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

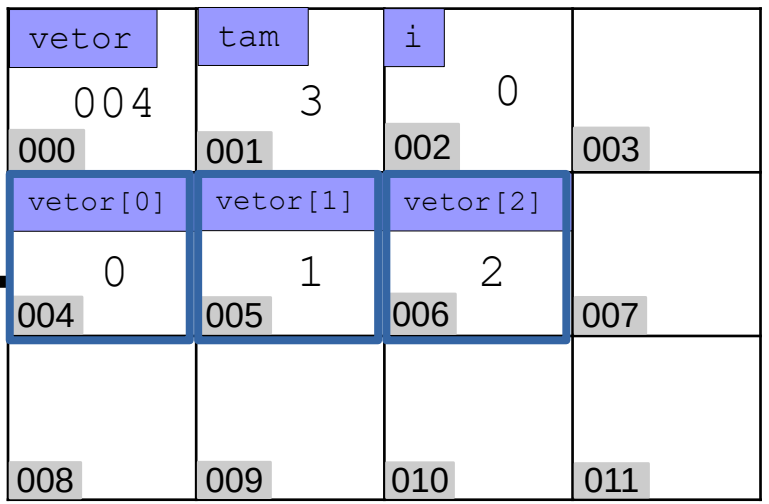
= 0 + 1

= 1

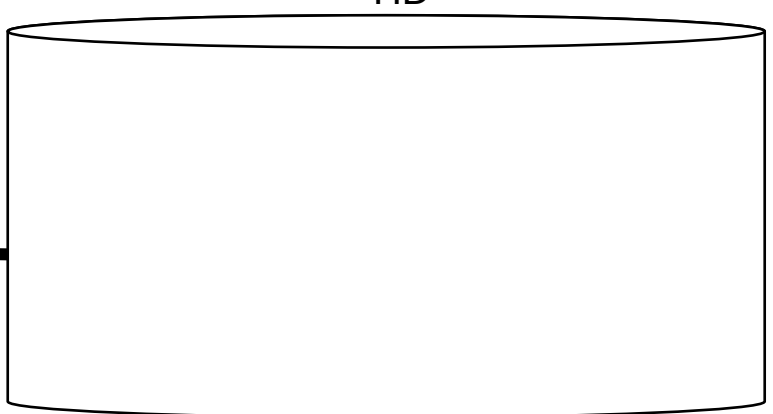
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>
vetor[0] = 0

Memória



HD



Execução

```
#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

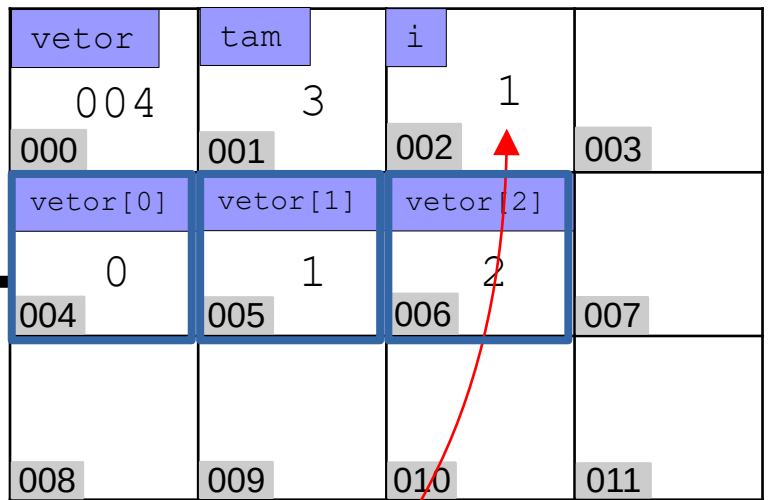
= 0 + 1

= 1

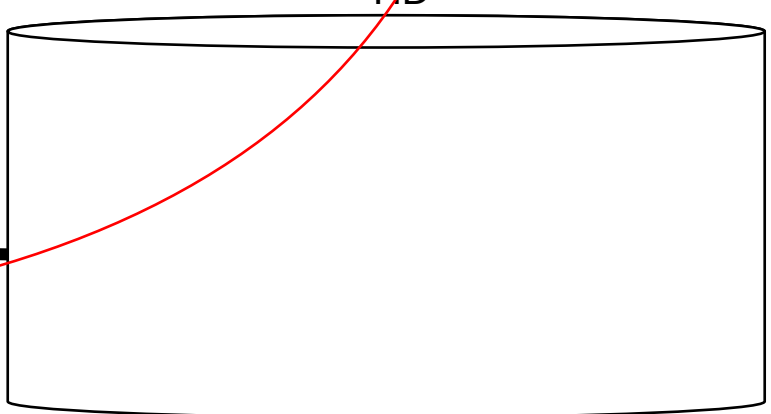
Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>
vetor[0] = 0

Memória



HD



Execução

```
#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

= 1 < 3

= 1 (verdadeiro)

Monitor / Teclado

Qual o tamanho do vetor? 3
<ENTER>
vetor[0] = 0

Memória

vetor	tam	i	
004	3	1	
000	001	002	003
vetor[0]	vetor[1]	vetor[2]	
0	1	2	
004	005	006	007
008	009	010	011

HD

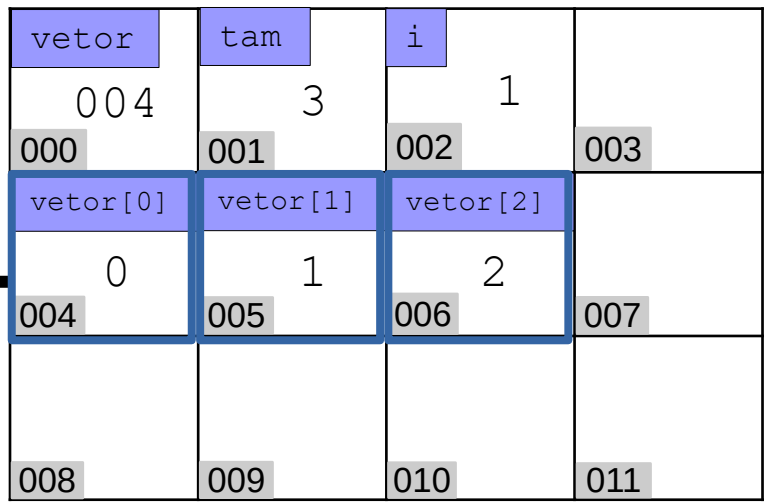
Execução

```
#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;
}
```

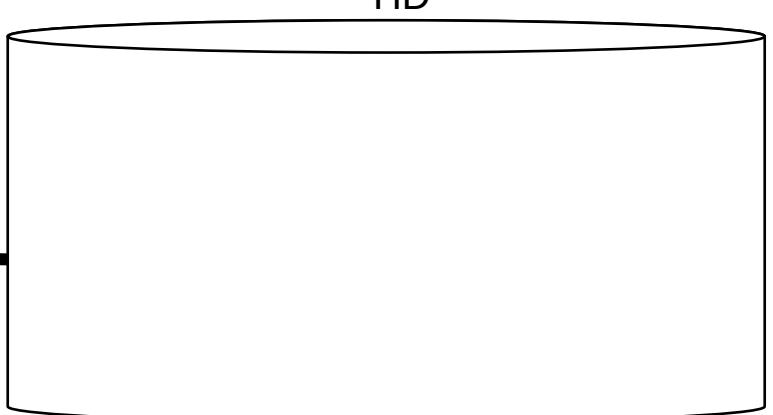
Monitor / Teclado

<ENTER>
vetor[0] = 0
vetor[1] = 1

Memória



HD



Execução

```
#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

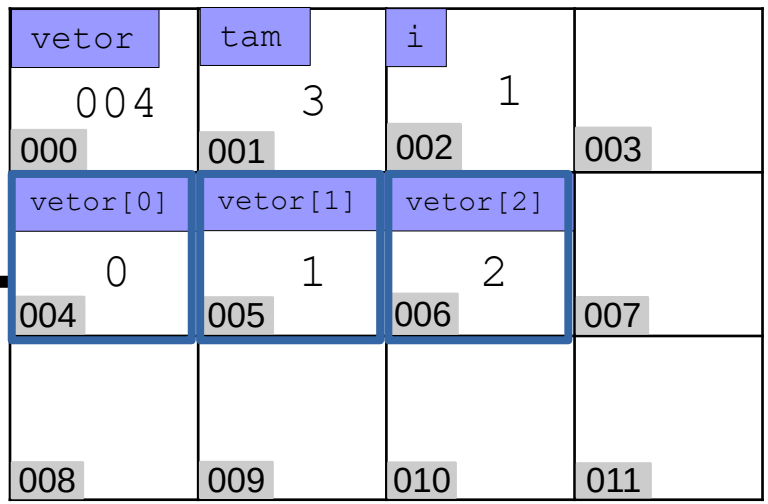
= 1 + 1

= 2

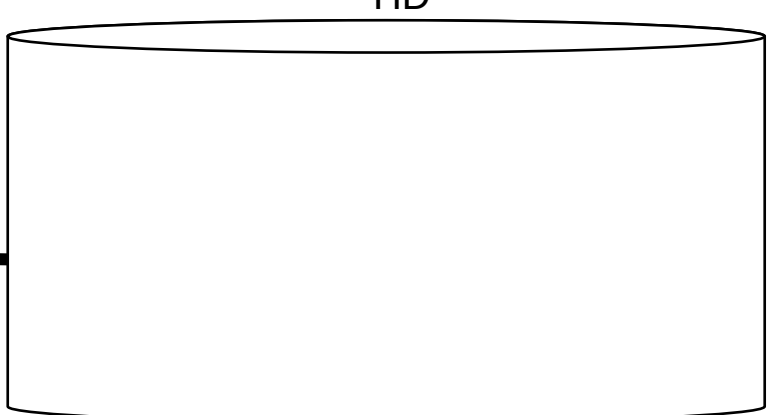
Monitor / Teclado

<ENTER>
vetor[0] = 0
vetor[1] = 1

Memória



HD



Execução

```
#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

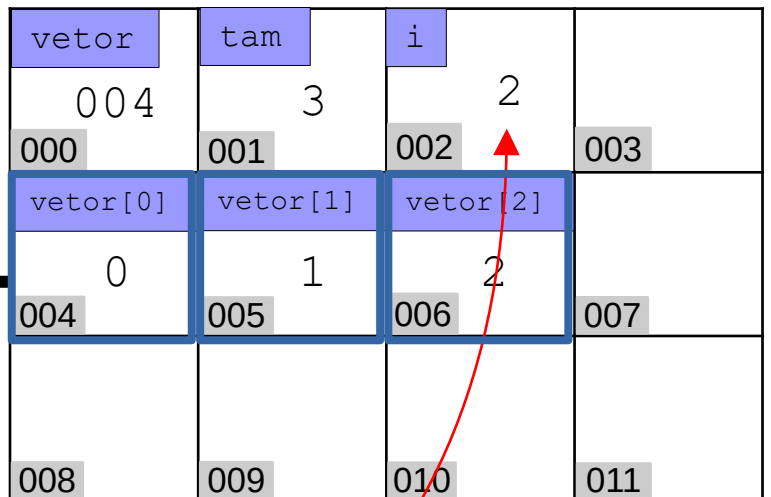
= 1 + 1

= 2

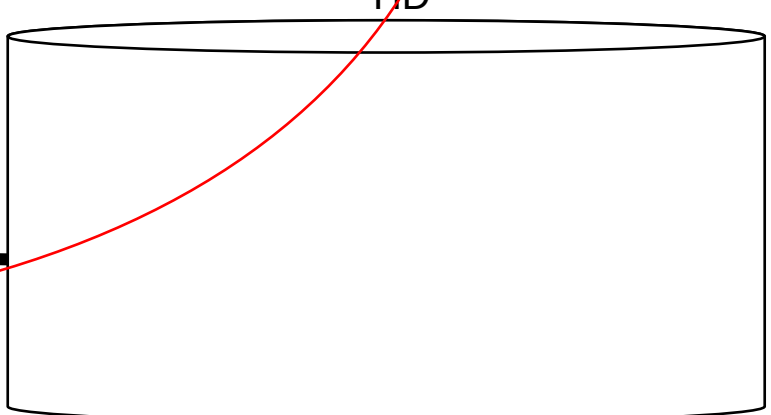
Monitor / Teclado

```
<ENTER>
vetor[0] = 0
vetor[1] = 1
```

Memória



HD



Execução

```
#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

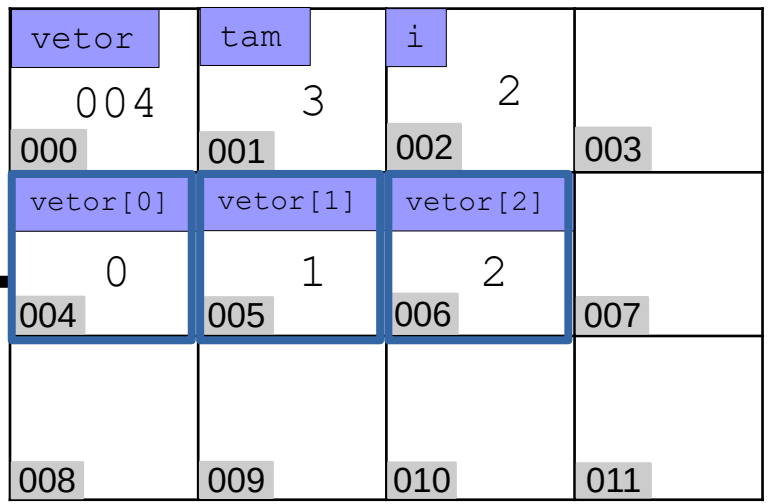
= 2 < 3

= 1 (verdadeiro)

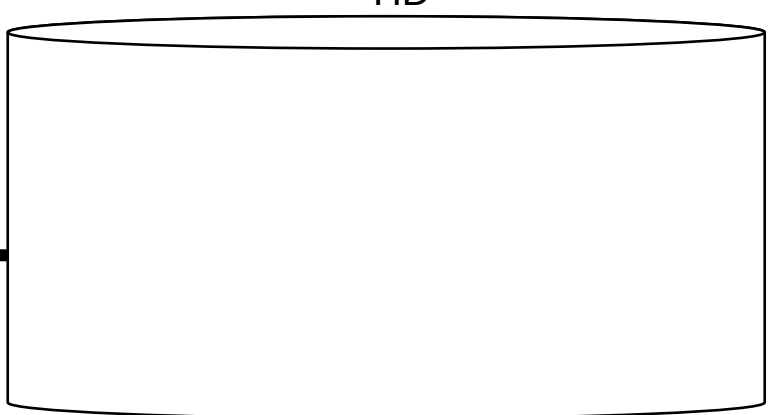
Monitor / Teclado

<ENTER>
vetor[0] = 0
vetor[1] = 1

Memória



HD



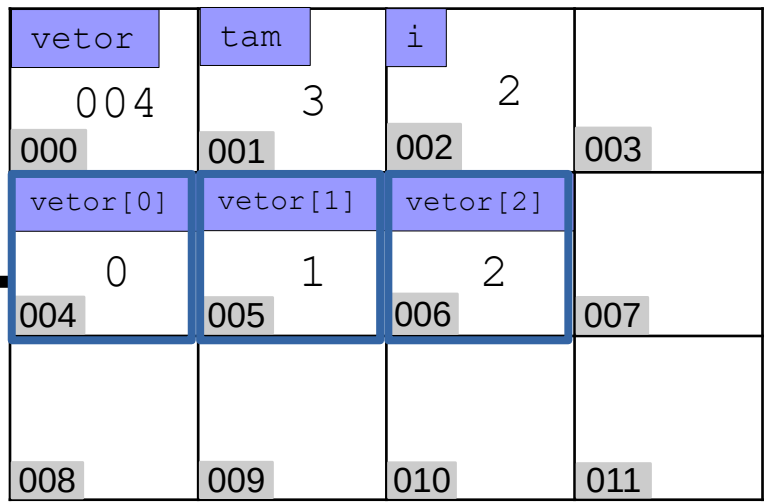
Execução

```
#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;
}
```

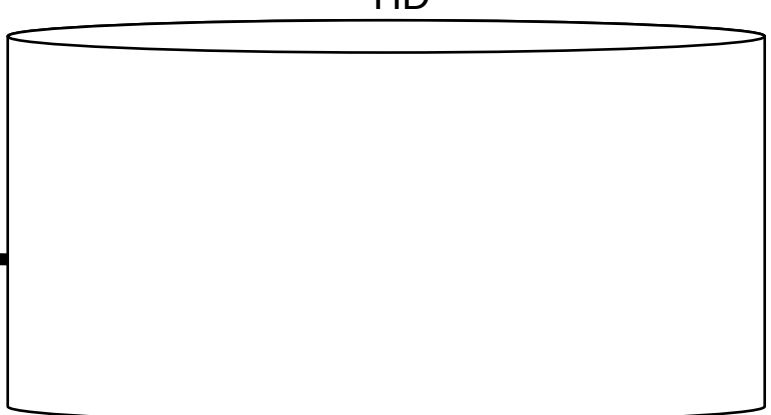
Monitor / Teclado

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
```

Memória



HD



Execução

```
#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

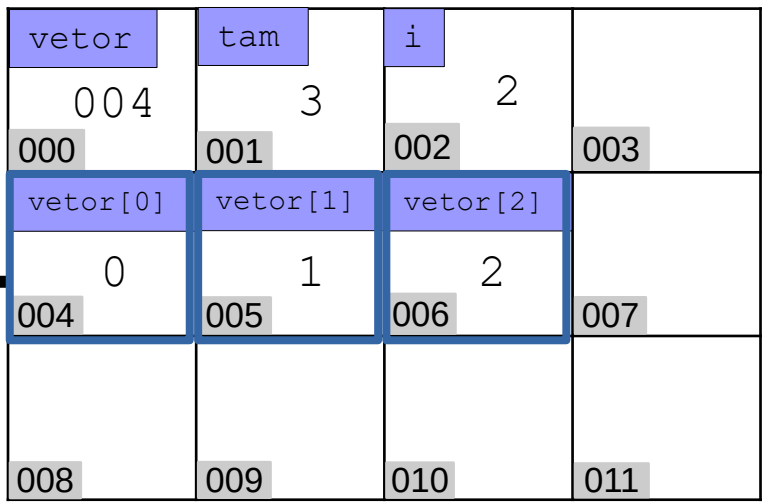
= 2 + 1

= 3

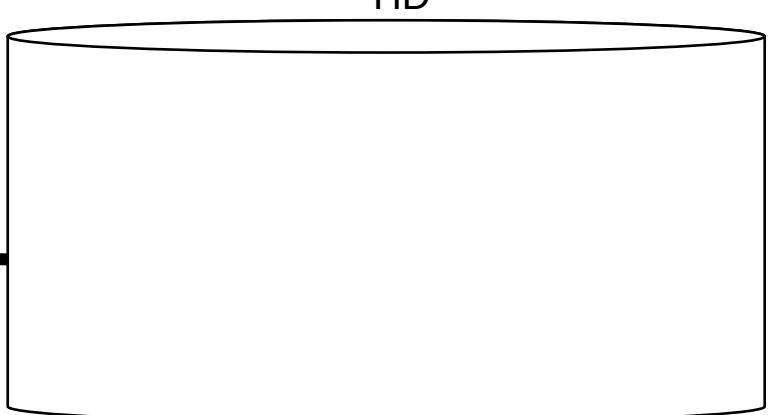
Monitor / Teclado

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
```

Memória



HD



Execução

```
#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

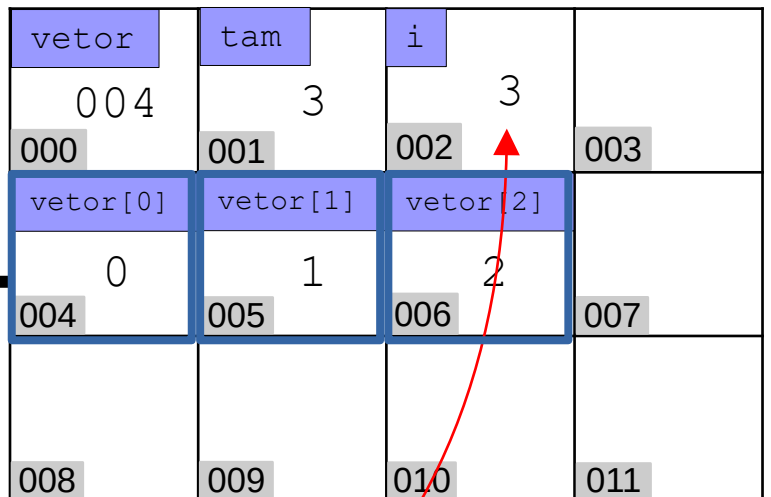
= 2 + 1

= 3

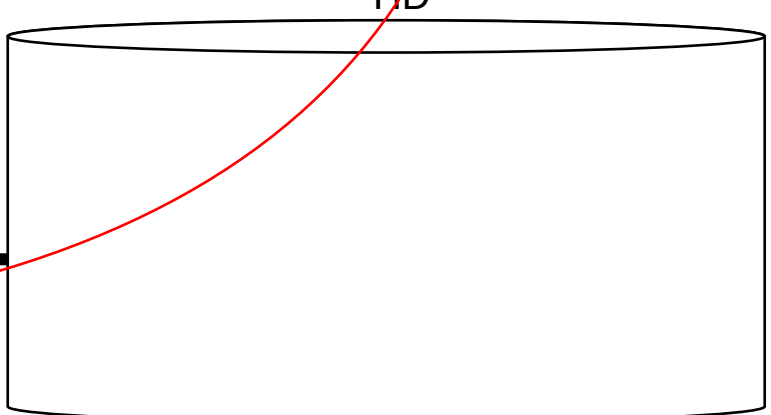
Monitor / Teclado

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
```

Memória



HD



Execução

```
#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

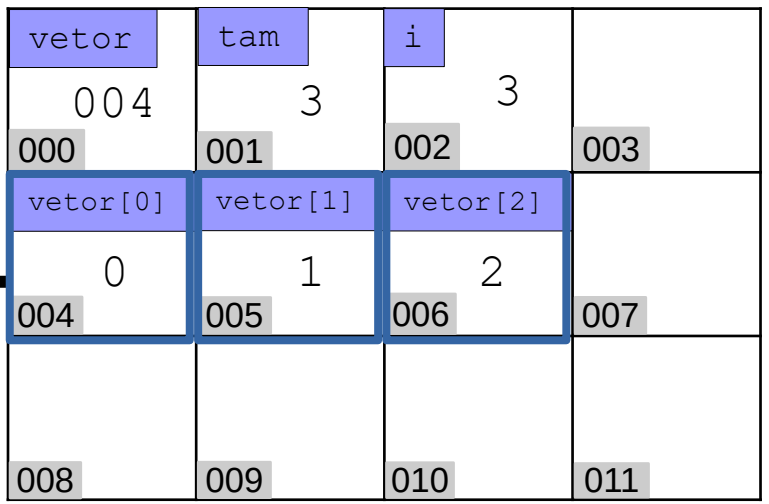
= 3 < 3

= 0 (falso)

Monitor / Teclado

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
```

Memória



HD

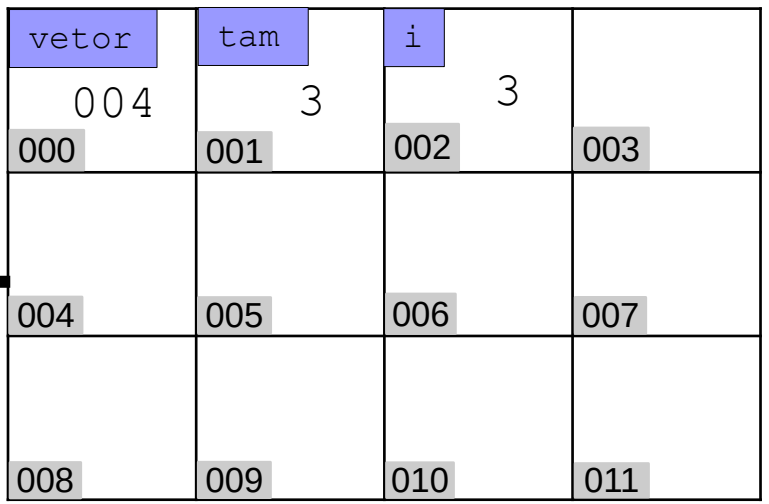
Execução

```
#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;
}
```

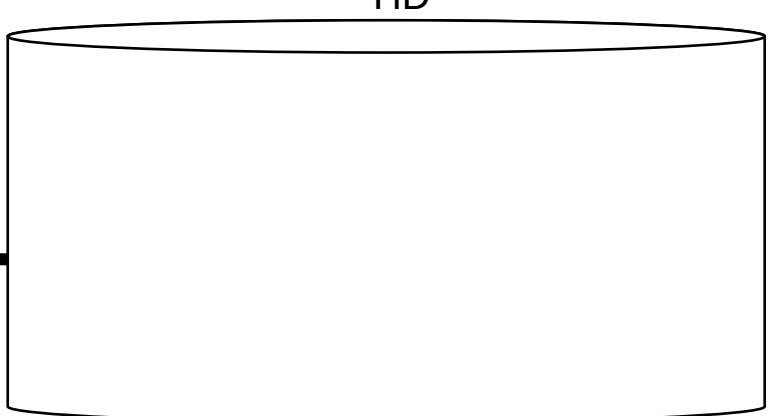
Monitor / Teclado

```
vetor[0] = 0
vetor[1] = 1
vetor[2] = 2
```

Memória

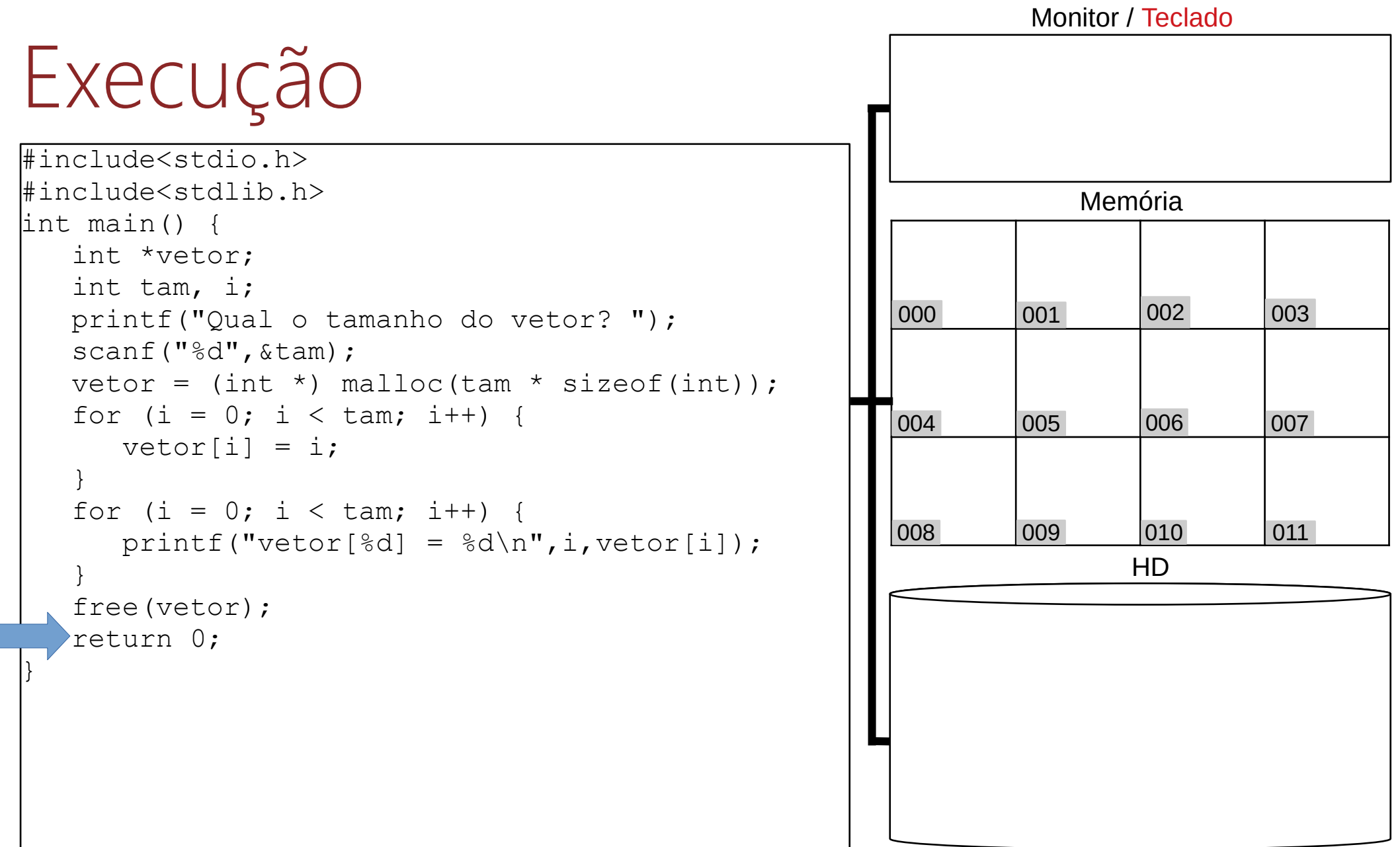


HD



Execução

```
#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;
}
```



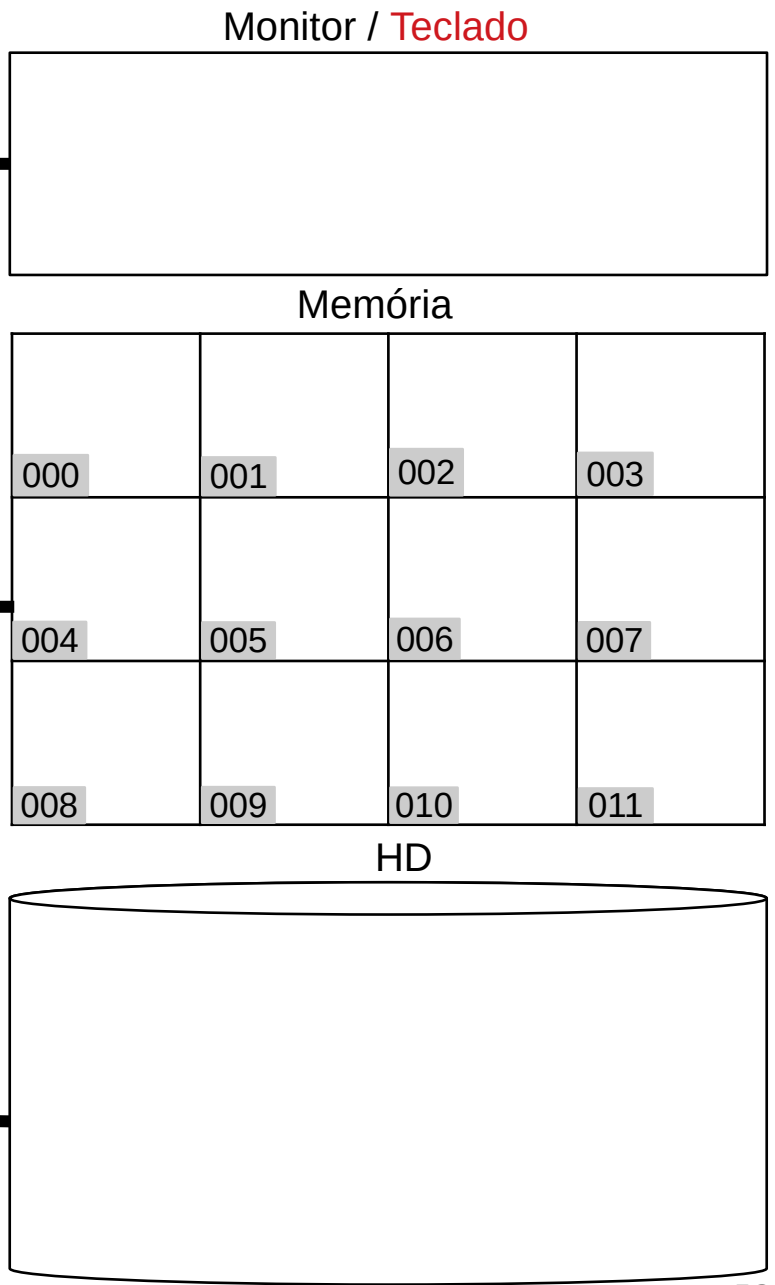
Outros assuntos

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;
}
```

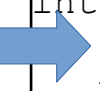
Execução

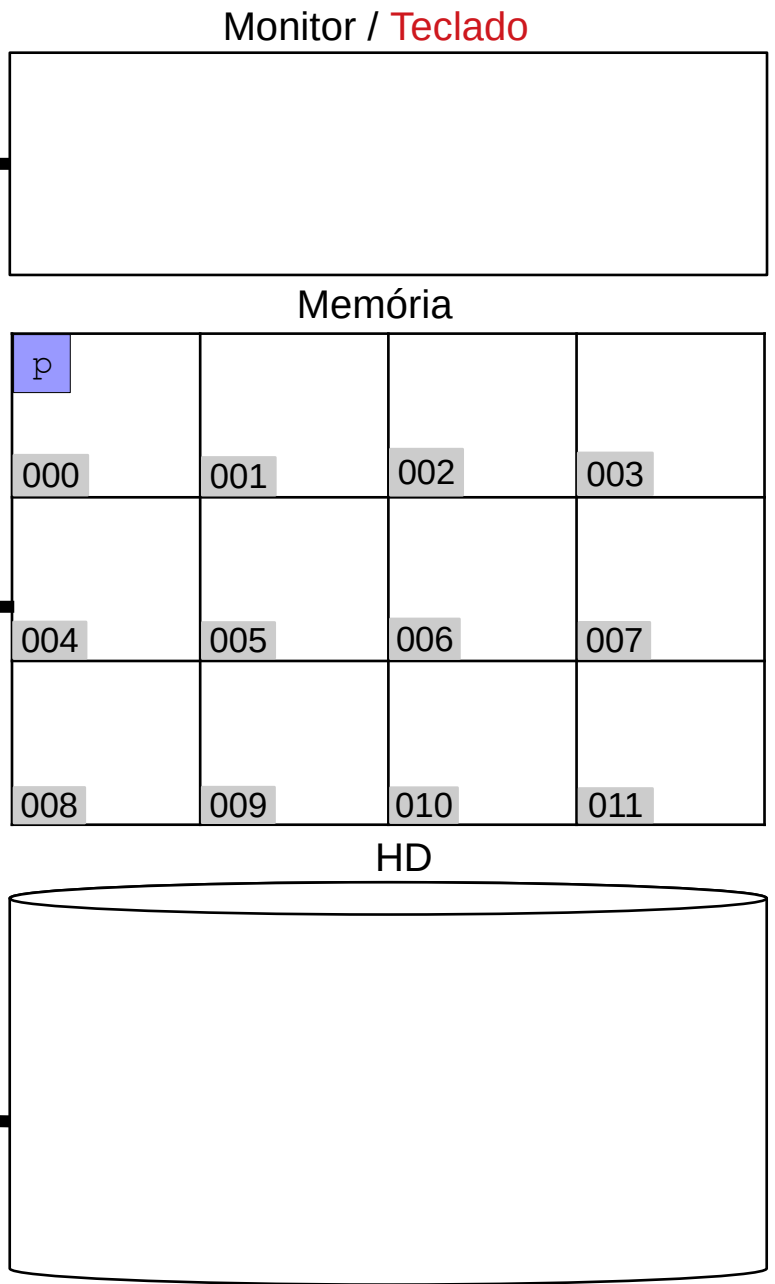
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
};
int main() {
   struct ponto *p;
   p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
   p->y = 6.28;
   printf("x = %.2f  y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
}
```



Execução

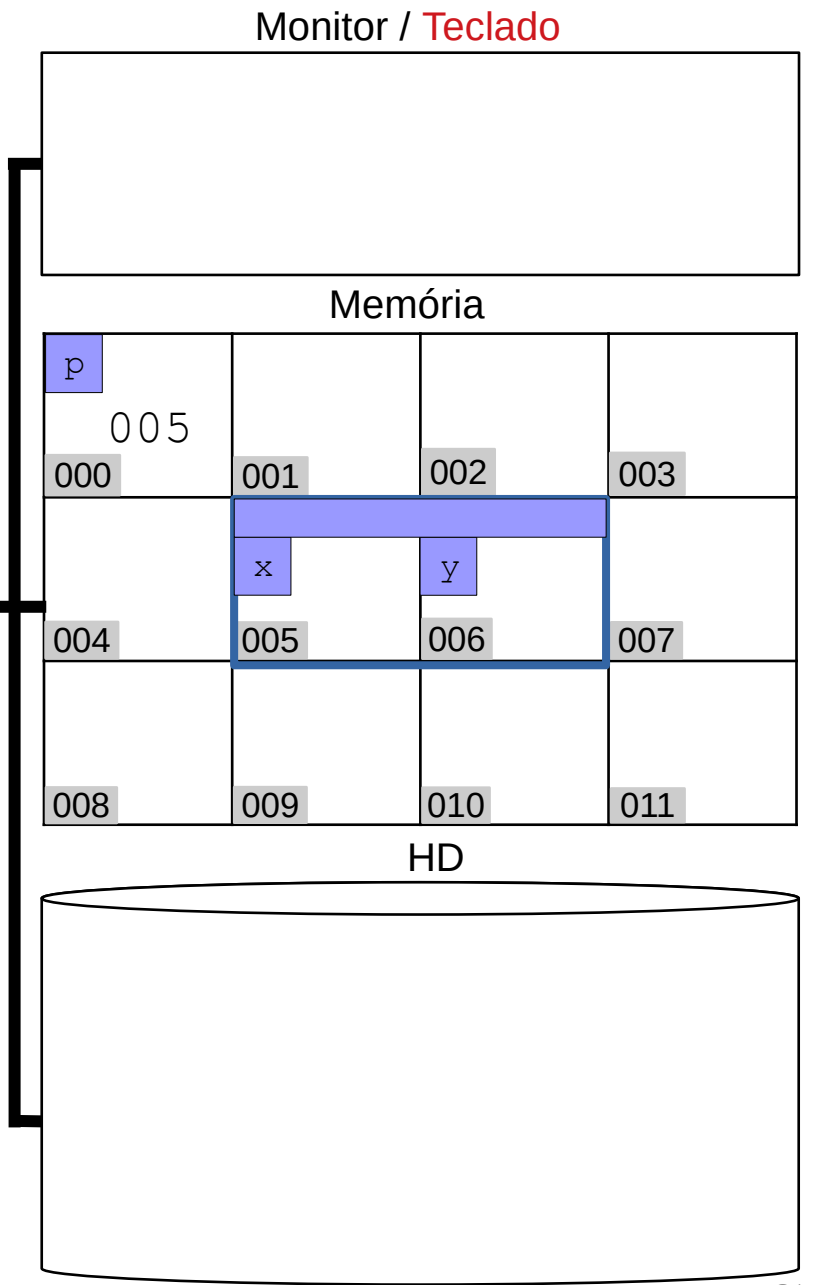
```
#include<stdio.h>
#include<stdlib.h>
struct ponto {
   float x;
   float y;
};
int main() {
   struct ponto *p;
   p = (struct ponto *) malloc(sizeof(struct ponto));
   (*p).x = 3.14;
   p->y = 6.28;
   printf("x = %.2f  y = %.2f\n",p->x,p->y);
   free(p);
   return 0;
}
```





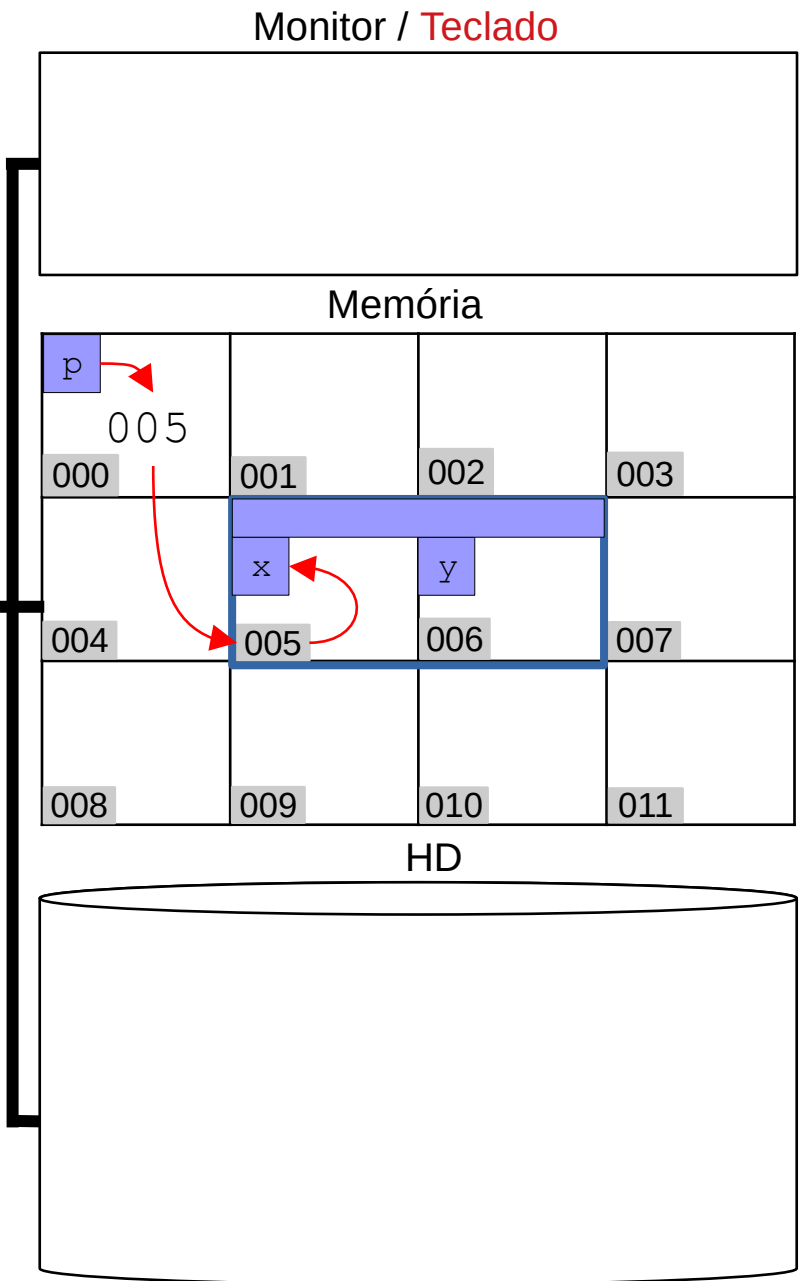
Execução

```
#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;
}
```



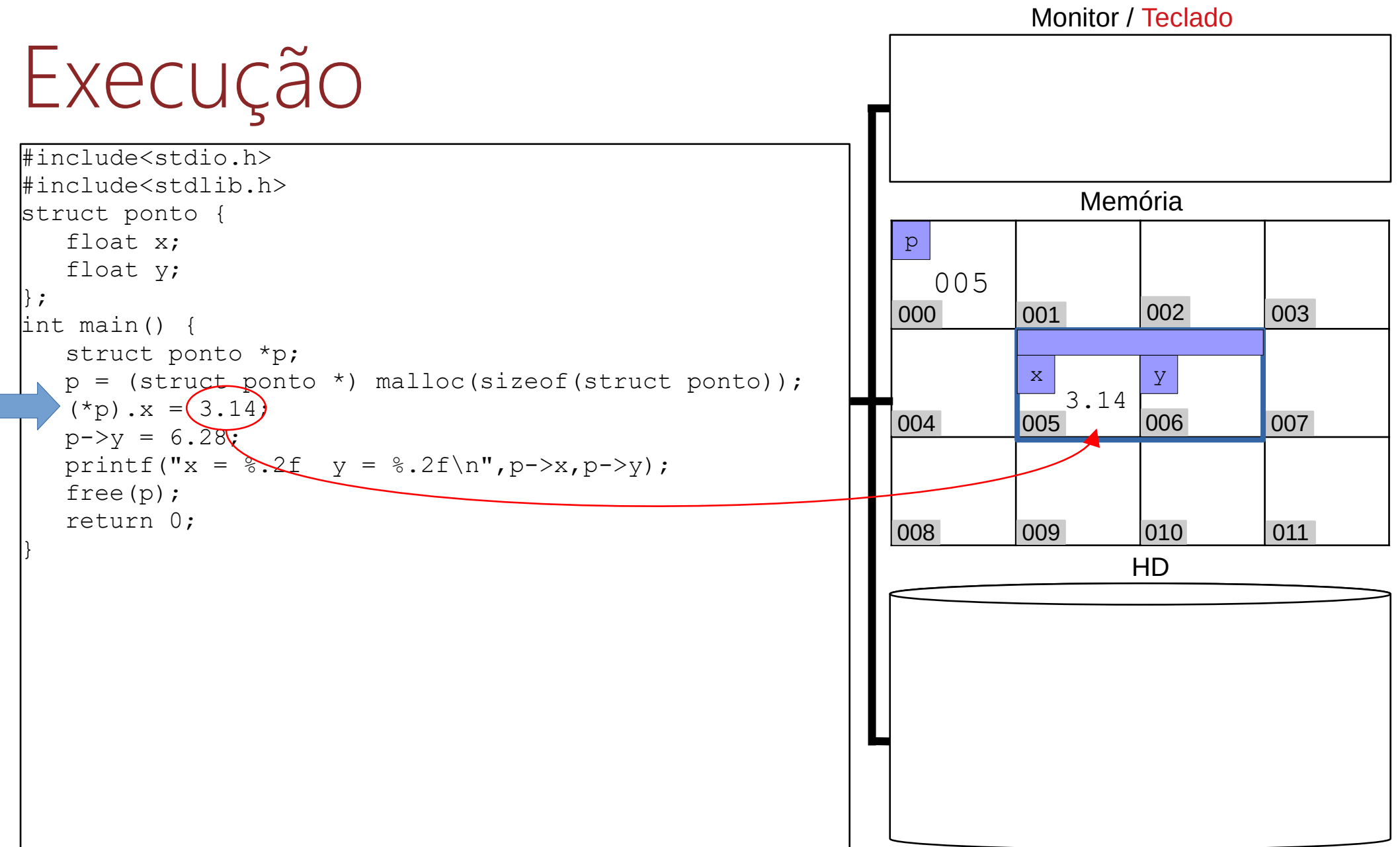
Execução

```
#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;
}
```



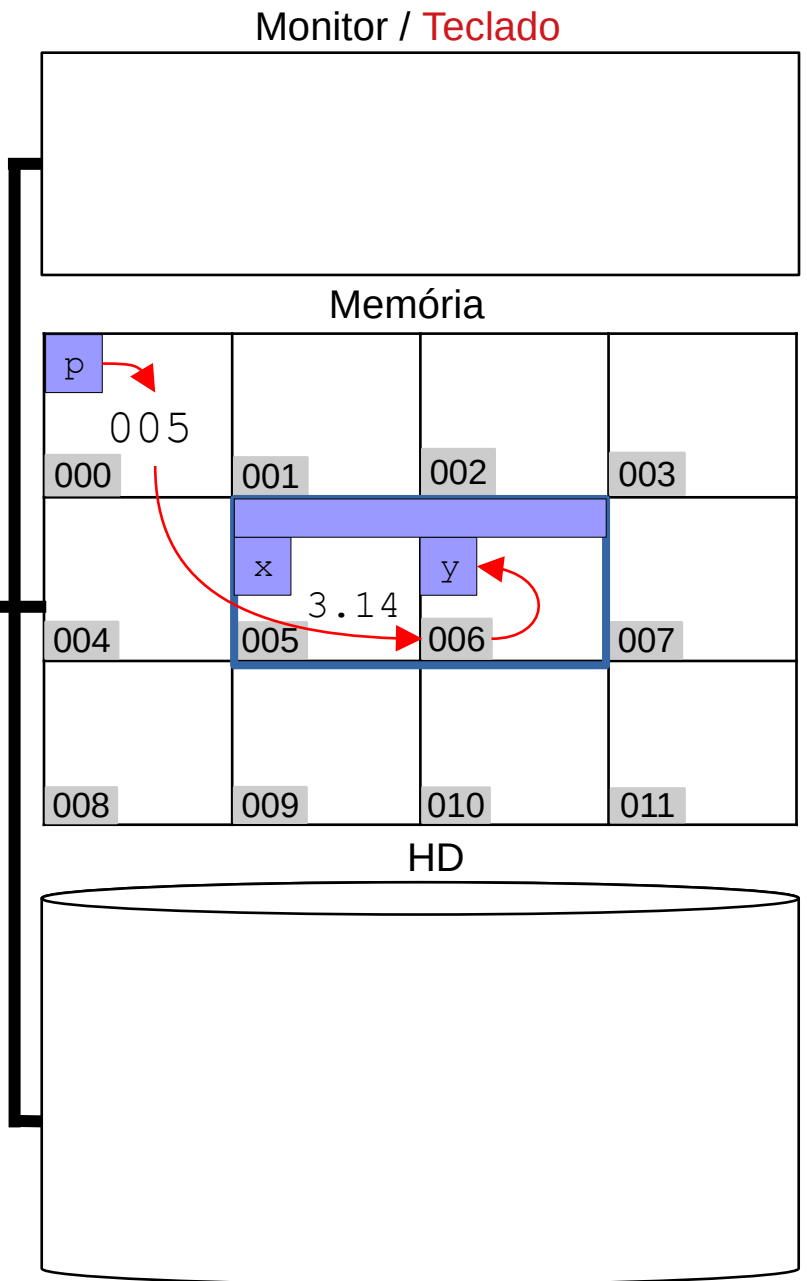
Execução

```
#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;
}
```



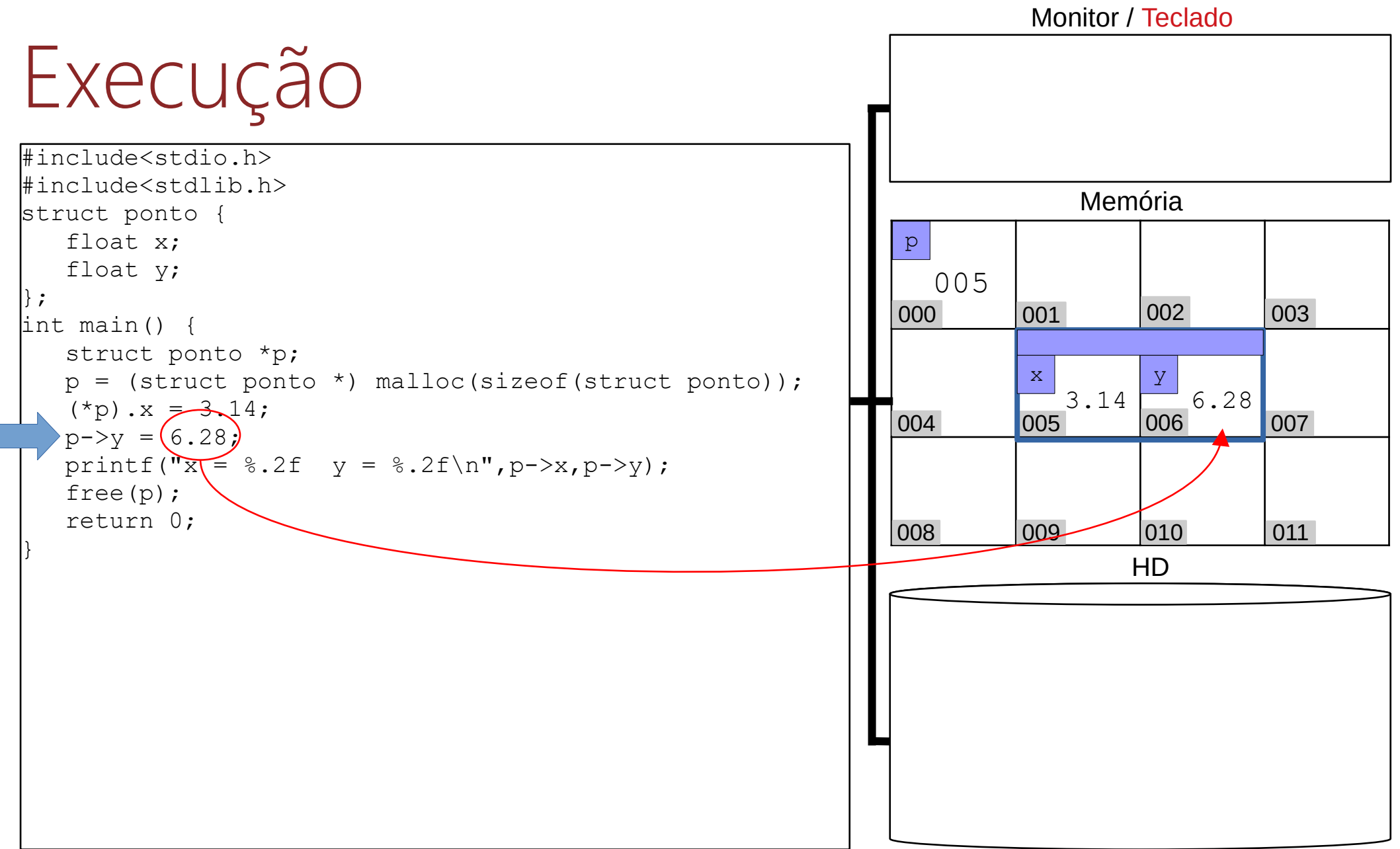
Execução

```
#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;
}
```



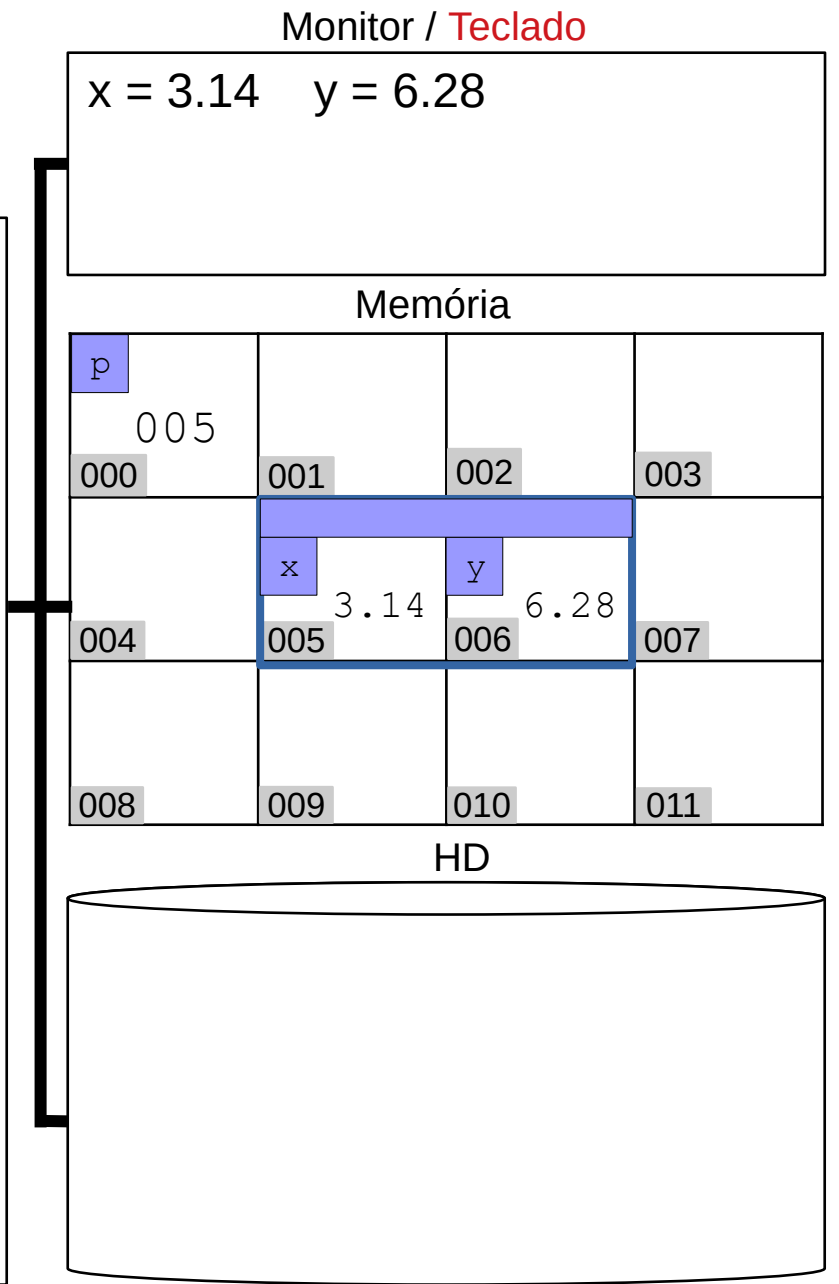
Execução

```
#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;
}
```



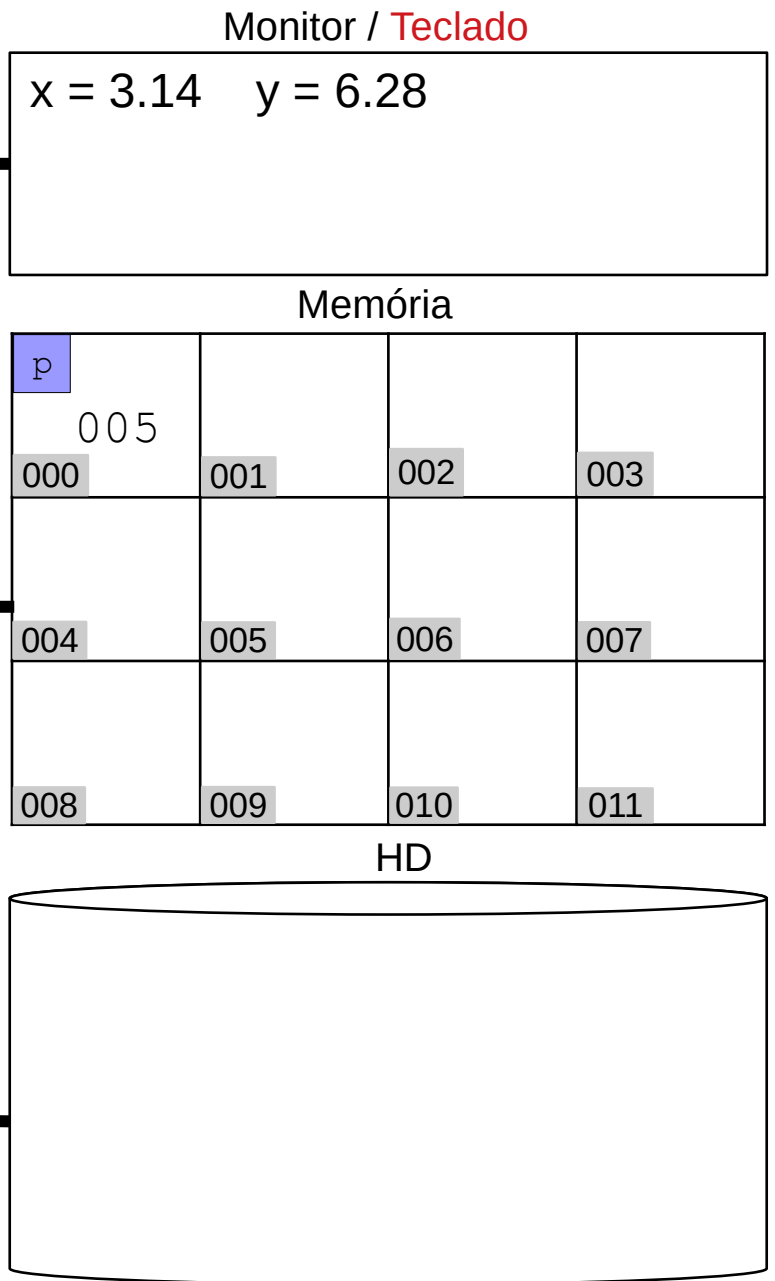
Execução

```
#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;
}
```



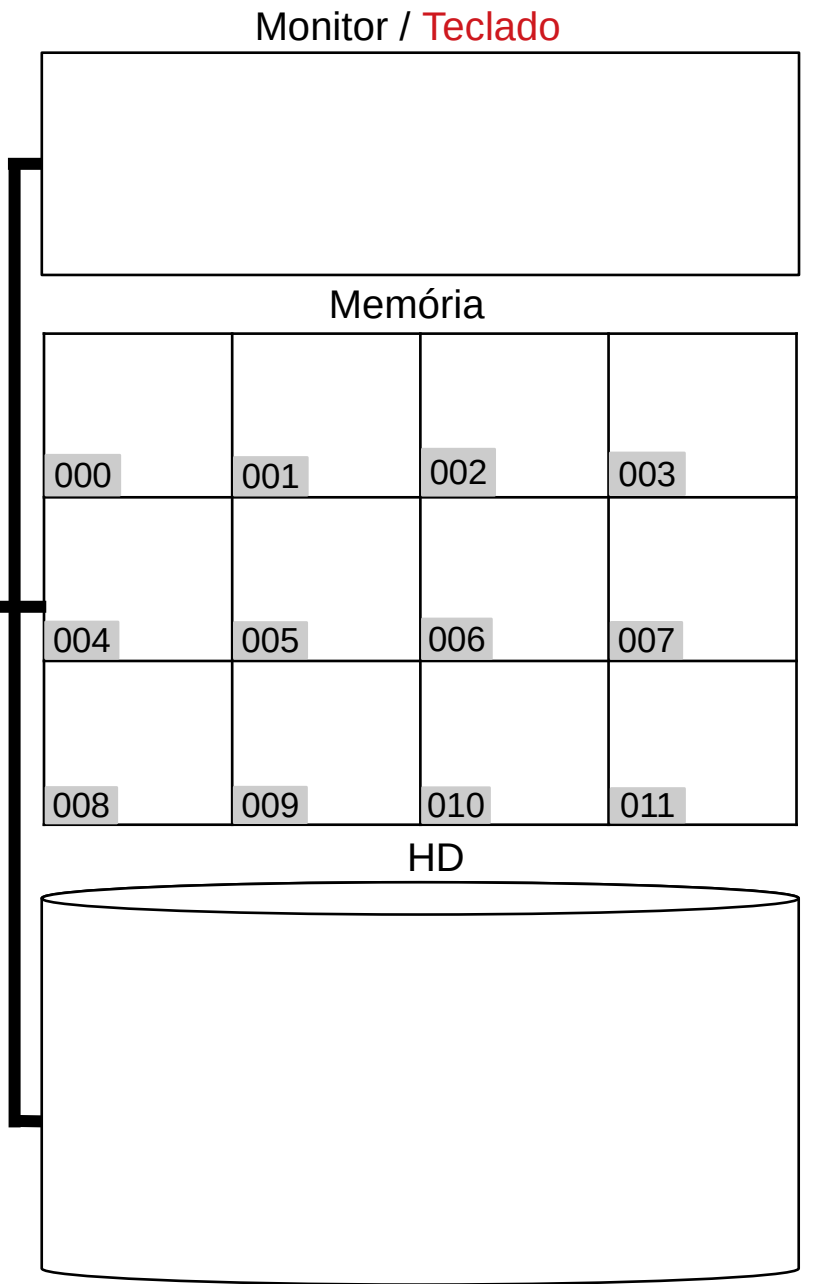
Execução

```
#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;
}
```



Execução

```
#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;
}
```



Outros assuntos

malloc()

- Definição
- Exemplo

Outros assuntos

malloc()

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

Outros assuntos
malloc()

- 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;
}
```

Outros assuntos

malloc()

- Definição
- Exemplo

Centro de Informática
UFPE

Outros assuntos

malloc()

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