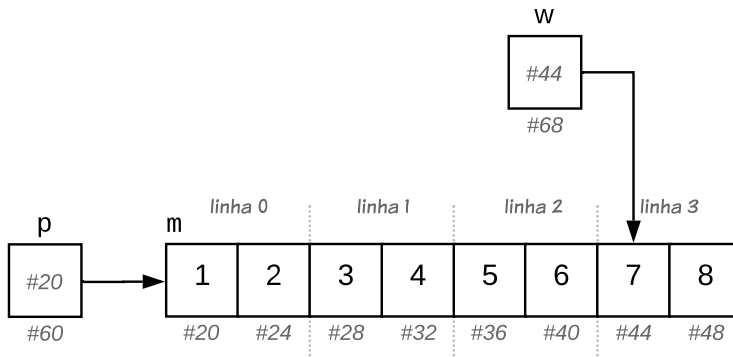


Matriz com alocação estática



Escreva o código que reproduz a ilustração acima

```
int m[4][2] = {{1,2},{3,4},{5,6},{7,8}};
int* p = (int*) m;
int* w = m[3];
```

Determine os valores com base na ilustração

```
m : #20
m[0] : #20
p : #20
m+1 : #28
m[1] : #28
p + 3 : #32
m[0][0] : 1
*m[0] : 1
*p : 1
p[0] : 1
m[3][1] : 8
*(m[3] + 1) : 8
*(p+(3*2)+1) : 8
w[1] : 8
*(w+1) : 8
```

Com base na ilustração, escreva um trecho de código que percorra todos os elementos da matriz e imprima o endereço de memória e o valor armazenado. O código deve ser genérico, ou seja, deve ser capaz de percorrer qualquer matriz. Para isso, utilize 2 variáveis para representar o número de linhas e colunas da matriz. Por exemplo:

```
int num_linhas = 4, num_colunas = 2;
```

Neste trecho, utilize a variável m por meio da notação de colchetes.

```
int num_linhas = 4, num_colunas = 2;
int i,j;
for(i=0; i<num_linhas; i++){
  for(j=0; j<num_colunas; j++){
    printf("m[%d][%d]: %d\n",i,j, m[i][j]);
  }
}
```

Neste trecho, utilize a variável p e a notação de ponteiros. Não é permitido o uso dos colchetes.

```
for(i=0; i<num_linhas; i++){
  for(j=0; j<num_colunas; j++){
    printf("(p+(%d*%d)+%d) : %d\n",i,num_colunas, j, *(p+(i*num_colunas)+j));
  }
}

for(i=0; i<num_linhas*num_colunas; i++){
  printf("(p+%d) : %d\n",i, *(p+i));
}
```