```
typedef struct no No;
struct no{
   int valor;
   No *cima, *baixo;
};
```

A estrutura Nó contém o conteúdo do nó, um ponteiro que aponta para cima e outro para baixo.

```
typedef struct matriz Matriz;
struct matriz{
   No *primeiro;
   int m, n;
};
```

A estrutura Matriz contém um ponteiro que aponta para o primeiro nó, o tamanho de linhas e colunas.

Matriz *criaMatriz();

A função criaMatriz cria uma matriz inicializando um nó e fazendo os apontamentos para NULL.

Matriz *alocaMatriz(Matriz *mat, int m, int n);

A função alocaMatriz recebe como parâmetro uma matriz existente e o tamanho dela, verifica se ela não é nula, aloca todos os elementos na memória, e em seguida insere os dados digitados pelo usuário.

Matriz *deletaMatriz(Matriz *mat);

A função deletaMatriz recebe como parâmetro uma matriz existente, verifica se ela não é nula e a libera.

void printaMatriz(Matriz *mat);

A função printaMatriz recebe como parâmetro uma matriz existente, verifica se ela não é nula e percorre toda a matriz mostrando os valores.

No *insereValor(Matriz *mat, int i, int j, int valor);

A função insereValor recebe como parâmetro uma matriz existente, a linha e coluna onde será inserido o valor, o valor, verifica se a matriz não é nula, verifica se as coordenadas estão no intervalo da matriz, aloca um elemento e seu valor nas coordenadas indicadas.

No *buscaValor(Matriz *mat, int valor);

A função buscaValor recebe como parâmetro uma matriz existente e um valor, verifica se a matriz não é nula e retorna as coordenadas do valor buscado.

No *printaVizinhos(Matriz *mat, int i, int j);

A função printaVizinhos recebe como parâmetro uma matriz existente e as coordenadas onde desejamos visualizar os vizinhos. Verifica se a matriz não é nula e se as coordenadas recebidas estão no intervalo da matriz, e retorna os vizinhos da coordenada indicada.