DCC013 - Estrutura de Dados / DCC107 - Laboratório de Programação II

Matrizes Especiais

Você pode utilizar qualquer ambiente de programação para desenvolver sua atividade. Ao final, copie e cole o seu código-fonte com a resposta aqui mesmo neste documento, dentro dos espaços indicados para isso e preservando a indentação do código. Depois que terminar sua avaliação, não se esqueça de entregar sua atividade! Fique atento ao relógio, pois as atividades entregues com atraso não serão aceitas.

Considere uma matriz que possui elementos nas metades finais das primeiras linhas e metades iniciais das últimas linhas e o restante dos elementos é zero conforme ilustrado na figura a seguir:

$$Metades_{[6x6]} = \begin{bmatrix} 0 & 0 & 0 & 6 & 2 & 3 \\ 0 & 0 & 0 & 4 & 9 & 6 \\ 0 & 0 & 0 & 1 & 5 & 8 \\ 3 & 5 & 4 & 0 & 0 & 0 \\ 2 & 1 & 9 & 0 & 0 & 0 \\ 7 & 5 & 3 & 0 & 0 & 0 \end{bmatrix}$$

Implemente o TAD `MatrizEspecial` para representar a matriz por meio da representação linear com um único vetor (`vet`) tal que a quantidade de elementos armazenados seja mínima. E desenvolva:

a) O construtor e o destrutor. O construtor deve exigir que a ordem da matriz seja no mínimo igual a 3, deve dimensionar o `vet` para o tamanho mínimo necessário e deve preencher a matriz.

```
MatrizEspecial::MatrizEspecial(int n) {
      ordem = 0;
      tam = 0;
      if (n >= 3) {
         ordem = n;
         if (ordem % 2 == 0)
            tam = 2*(ordem/2)*(ordem/2);
         else
            tam = 2*(ordem-1/2)*(ordem-1/2);
         vet = new float[tam]; // [ ACIMA, ABAIXO]
      }
      else {
         exit(1);
         cout <<"Erro: ordem menor que 3" << endl;
      }
```

```
}

MatrizEspecial::~MatrizEspecial() {
    delete [] vet;
}
```

a) A operação `int getInd(int i, int j)` para verificar se os índices i e j da matriz são válidos e retornar o índice de `vet` de acordo com o formato armazenado (-2 caso nao precise armazenar ou -1 caso não sejam válidos).

```
int MatrizEspecial::getInd(int i, int j) {
    if (i >= 0 && i < ordem && j >= 0 && j < ordem) {
        if (ordem % 2 == 0) {
            if (i < ordem/2 && j >= ordem/2) { // acima: ordem é da esq pra direita e DEPOIS de cima pra baixo
                if (j % 2 == 0) {
                    return i + j - ordem/2;
                else
                    return i + ordem - j;
            }
            if (i >= ordem/2 && j < ordem/2) { // abaixo: ordem é de cima p baixo e depois da esq pra direita
                if (i % 2 == 0)
                    return (j + i - ordem/2) + (ordem/2)*(ordem/2) - 1;
                else
                    return (j + ordem - i) + (ordem/2)*(ordem/2) - 1;
            }
        }
        else if (ordem % 2 != 0) {
            if (i < ordem-1/2 && j >= ordem+1/2) { // acima: ordem é da esq pra direita e DEPOIS de cima pra baixo
                if (j % 2 != 0)
                    return i + j - ordem+1/2;
                else
                    return i + ordem - j;
            }
            if (i >= ordem+1/2 && j < ordem-1/2) { // abaixo: ordem é de cima p baixo e depois da esq pra direita
                if (i % 2 != 0)
                    return (j + i - ordem+1/2) + (ordem-1/2)*(ordem-1/2) - 1;
                else
                    return (j + ordem - i) + (ordem-1/2)*(ordem-1/2) - 1;
            }
        }
        else
            return -2;
    }
    return -1;
```

```
}
```

b) A operação `int get(int i, int j)` que retorna o valor armazenado na posição i, j da matriz, zero caso o índice seja -2 e imprime uma mensagem de erro e sai do programa caso o índice seja inválido.

```
float MatrizEspecial::get(int i, int j) {
        int indice = getInd(i, j);
        if (indice != -1) {
            if (indice == -2)
                return 0;
            return vet[indice];
        }
        else {
            cout <<"Erro: tentar getar valor fora da matriz" << endl;
            exit(1);
        }
    }
```

c) A operação `void set (int i, int j, int val)` que atribui o valor `val` à posição i, j da matriz e imprime mensagem de erro para índice incorreto.

```
void MatrizEspecial::set(int i, int j, float val) {
        int indice = getInd(i, j);
        if (indice != -1) {
            if (indice == -2)
                cout << ""; // erro: tentar setar o 0
            vet[indice] = val;
            }
        else {
            //exit(1);
            cout <<"Erro: setar elemento de fora da matriz" << endl;
        }
    }
```