EAF I Feladat dokumentáció III. feladatcsoport 9. házi feladathoz

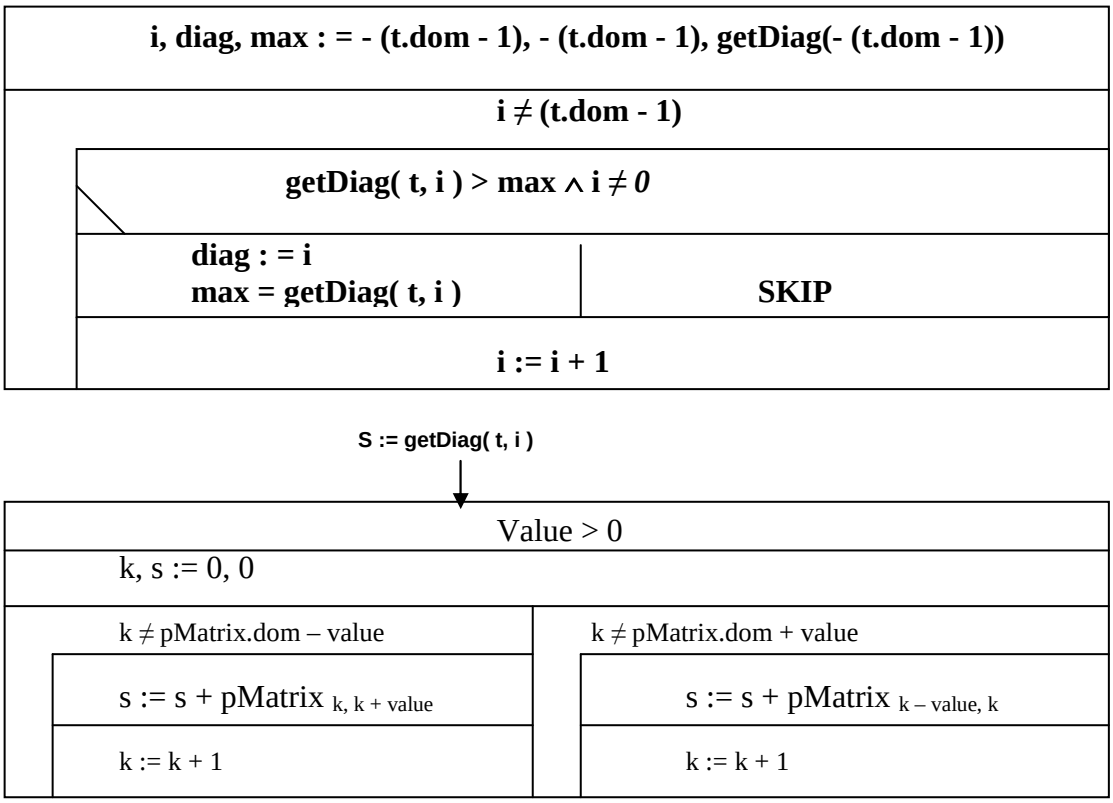
Feladat

Keressük meg a t négyzetes mátrixnak azt a fődiagonálissal párhuzamos átlóját, amelyben az elemek összege a legnagyobb!

Specifikáció

$A = M \times Z \times R \times Z$
$\quad\; v \quad i \quad max \quad k$
$B = M$
$\quad\; t'$
$Q = (t = t' \wedge t.dom > 1 \wedge \forall i \in [t.lob,t.hib] : t[i].dom = t.dom)$
$R = (Q \wedge i \in [t.lob,t.hib] \wedge \forall j \in [t.lob,t.hib] : getDiag(t, i) >= getDiag(t, j))$

Absztrakt program



Tesztelési terv

Tesztesetek a feladat alapján (fekete doboz tesztelés)

1. Bevitelek tesztelése
2. Fájból felolvasás tesztelése, elsőnek a vect mérete kell, majd space-el elválasztva a számjegyek, ennek is helyességét ellenőrizve.

Tesztesetek a kód alapján (fehér doboz tesztelés)

1. Olyan számadatok (helyes és hibás) megadása, amellyel a beolvasó ellenőrző ágát letesztelhetjük.
2. Hibás adat esetén, új adat bekérése

Megoldás C++

```
//Keressük meg a t négyzetes mátrixnak azt a fődiagonálissal párhuzamos átlóját,
//amelyben az elemek összege a legnagyobb!
struct matrix
{
	int** pData;
	int dom;
};
bool GetMatrix(matrix &pMatrix/*, int tmpX, int tmpY*/); //négyzetes mátrix
void getNum(int &n, int lob, int hib);
void ClearMatrix(matrix &pMatrix);
bool IsDigit(char ch );
bool ReadMatrix(matrix &pMatrix, char mode, string filenm);
int getDiagonal(matrix pMatrix, int value);
//---------------------------------------------------------------------
int main(int argc, char* argv[])
{
string alldo;
do{
	string filenm = "";
	char mode = 0;
	if(argv[ 1 ] != "" && argc >= 2)
	{
		filenm = argv[ 1 ];
		cout << argv[ 1 ] << endl;
		mode = 'F';
	}
	else
	{
		do
		{
			string str;
			cout << "Tomb feltoltese billenyuzetrol - ( B ) \nFeltoltes fajlbol - ( F )\n:= ";
			cin >> str;
			if(str == "B" || str == "b")
			{
				mode = 'B';
			}
			else if(str == "F" || str == "f")
			{
				mode = 'F';
			}
		}while( !( mode == 'F' || mode == 'B' ));
	} matrix t;
	bool tBool = ReadMatrix(t, mode, filenm); // mátix létrehozása, felolvasása
	//kiíratás!
	if(tBool)
		{
		for(int i = 0; i < t.dom; i++)
		{
			for(int j = 0; j < t.dom; j++)
			{
				cout << t.pData[ i ][ j ] << ", " ;
			}
			cout << "\n";
		}
		//számítás
		int max = getDiagonal(t, -(t.dom - 1));
		int diag = -(t.dom - 1);
		int s = 0;
		for(int i = -(t.dom - 1); i < t.dom; i++)
		{
			if( i != 0)
				s = getDiagonal(t, i);
			if(s > max)
			{
				max = s;
				diag = i;
			}
		}
		cout << "A legnagyobb osszegu atlo := " << diag << ", A Max ertek := " << max << endl;
		ClearMatrix( t );
	} // itt majd újra kérem a bevitelt stb..
	cout << "Ujbol, \nakarja? ( I/N ) := ";
	cin >> alldo;
}while( !( alldo == "N" || alldo == "n")); // ez így nem szép.. de
	return 0;
}
//---------------------------------------------------------------------
```

```
int getDiagonal(matrix pMatrix, int value)
{
	int s = 0;

	if(value >= 0)
	{
		for(int k = 0; k < pMatrix.dom - value; k++)
		{
			s += pMatrix.pData[ k ][ k + value ];
		}
	}
	else
	{
		for(int k = 0; k < pMatrix.dom + value; k++)
		{
			s += pMatrix.pData[ k - value ][ k ];
		}
	}
	return s;
}//int getDiagonal(matrix pMatrix, int value)
//---------------------------------------------------------------------
bool ReadMatrix(matrix &pMatrix, char mode, string filenm)
{
	//Definiáljuk és megnyitjuk a fájlt
	if(filenm == "" && mode == 'F')
	{
		cout << "A fajl neve := ";
		cin >> filenm;
	}
	ifstream x(filenm.c_str());
	if(mode == 'B')
	{
		cout << "Matrix merete := ";
		getNum(pMatrix.dom, 1, INT_MAX); // Mátrix méretének meghatározása
	}else if(mode == 'F')
	{
		//Ha hiba van befejezzük a programot
		if (x.fail()){
			cout << "A megadott fajlt nem talalom!";
			//exit(1);
			return false;
		}
		//Beolvassuk/kiírjuk a tömb hosszát
		x >> pMatrix.dom;
	}
	//felolvasás
	int RANGE_MIN = 0;
	int RANGE_MAX = 100;
	bool tBool = GetMatrix( pMatrix );
	if(!tBool)
	{
		cout << "Hiba a matrix letrehozesanal!" << endl;
		return false;
	}
	for(int i = 0; i < pMatrix.dom; i++)
	{
		for(int j = 0; j < pMatrix.dom; j++)
		{
			cout << "Adja meg a(z) " << i << ", " << j << " helyiertekeket := ";
			if(mode == 'B')
			{
				getNum( pMatrix.pData[ i ][ j ], INT_MIN, INT_MAX);
			}else if(mode == 'F')
			{
				x >> pMatrix.pData[ i ][ j ];
				cout << pMatrix.pData[ i ][ j ] << "\n";
			}

		}
	}
	return true;
}//bool ReadMatrix(matrix &pMatrix, char mode)
//---------------------------------------------------------------------
bool GetMatrix(matrix &pMatrix/*, int tmpX, int tmpY*/)
{
	if( pMatrix.dom == 0 && pMatrix.dom == INT_MAX) //hiba a mátrix létrehozásánál
	{
		return false;
	}
	pMatrix.pData = new int*[ pMatrix.dom ];
    for(int i = 0; i < pMatrix.dom; i++)
    {
		pMatrix.pData[ i ] = new int[ pMatrix.dom ];
    }
//	pMatrix.domX = tmpX;
//	pMatrix.domY = tmpY;
	return true;
}//bool GetMatrix(matrix &pMatrix)
//---------------------------------------------------------------------
bool IsDigit(char ch)
{
	return ((ch >= '0' && ch <= '9') || ch == '-');
}//bool IsDigit(char ch)
//---------------------------------------------------------------------
```

```
void ClearMatrix(matrix &pMatrix)
{
    for(int i = 0; i < pMatrix.dom; i++)
    {
            delete[] pMatrix.pData[ i ];
    }
      delete[] pMatrix.pData;
      pMatrix.dom = 0;
      //pMatrix.hib = 0;
}//void ClearMatrix(matrix &pMatrix)
//---------------------------------------------------------------------
void getNum(int &n, int lob, int hib) //lob az alsó határ
{
      bool error;
      do
      {
            string str;
            n = 0;
            error = false;
            cin >> str;
            for(int h = 0; h < (signed)str.length(); h++)
            {
                  if(!IsDigit(str[ h ]))
                  {
                        error = true;
                        cout << "Hiba, Szamot adjon meg!\n"  << endl;
                        break;
                  }
            }
            if(!error)
            {
                  n = atoi(str.c_str());
            }
            if((n < lob || n >= hib) || error) //Nem engedjük meg a hib-ig menni
és a lob-nál nagyobbnak kell lennie!
            {
                  cout << "Hibas a bevitt ertek\n" << lob << " -tol, " << hib <<
" -ig. Adja meg a szamot." << endl;
                  error = true;
            }
      }while( !(lob <= n) || error);
}//int getNum()
//---------------------------------------
```