```
/* Examen Septiembre 2012*/
Class Matriz{
    private:
        int nfilas;
        int *ncolumnas
        double **datos;
        void liberarEspacio();

        friend ostream & operator<<(ostream & output,const Matriz & objeto);
        friend istream & operator>>(istream & flujo,Matriz & objeto);

        public:
        Matriz();
        ~Matriz();
        Matriz(const Matriz & otro);
        const Matriz & operator=(const Matriz & otro);
        void set(int fila, int columna, double valor);
        double get(int fila, int columna);
        Matriz & vFlip();
        void max(int &fila, int&columna);
        void escribir(string nombrearchivo,string comentario);
        void leerArchivo(string nombrearchivo);
}


//**********************************
//*************UTILS.CPP*************
//**********************************
ostream & operator<<(ostream &output, const & Matriz objeto){
    output<<endl;
    for(int i=0;i<nfilas;i++)
        for(int j=0;j<ncolumnas[i];j++)
            output<<objeto.datos[i][j]<<" ";;
return output;
}
istream & operator>>(istream & flujo,Matriz & objeto){
    for(int i=0;i<nfilas;i++)
        for(int j=0;j<ncolumnas[i];j++)
            flujo>>objeto.datos[i][j];
return flujo;
}


//**********************************
//*************UTILS.CPP*************
//**********************************


//**********************************
//***********Matriz.CPP**********
//**********************************
void Matriz::reservarEspacio(int nfilas, int *ncolumnas){

    this->nfilas=otro.nfilas;
    this->ncolumnas=new int[nfilas];
    datos=new double*[nfilas];
    for(int i=0;i<nfilas;i++){
        this->ncolumnas[i]=ncolumnas[i];
        this->datos[i]=new double[ncolumnas[i]];
    }
}
void Matriz::copiarDatos(const Matriz & otra){
    for(int i=0;i<nfilas;i++)
        for(int j=0;j<ncolumnas[i];j++)
            this->datos[i][j]=otra.datos[i][j];
}

void Matriz::liberarEspacio(){
    if(datos!=0){
        for(int i=0;i<nfilas;i++)
            delete [] datos[i];

        delete [] ncolumnas;
```

```
        delete [] datos;
        nfilas=0;
    }
}

Matriz::Matriz(){
    nfilas=0;
    ncolumnas=0;
    datos=0;
}

~Matriz(){
    liberarEspacio();
}

Matriz::Matriz(const Matriz & otro){
    reservarEspacio(otro.nfilas,otro.ncolumnas);
    copiarDatos(otro);
}


const Matriz & Matriz::operator=(const Matriz & otro){
    if(this!=otro){
        liberarEspacio();
        reservarEspacio(otro.nfilas,otro.ncolumnas);
        copiarDatos(otro);
    }
    return *this;
}

void Matriz::set(int fila, int columna, double valor){
    datos[fila][columna]=valor;
}

double Matriz::get(int fila, int columna){
    return datos[fila][columna];
}

Matriz & Matriz::vFlip(){
    Matriz * resultado = new Matriz(*this);
    for(int i=0;i<nfilas;i++){
        resultado->ncolumnas[i]=this->ncolumnas[nfilas-i-1];
        resultado->ncolumnas[nfilas-i-1]=this->ncolumnas[i];
        resultado->datos[i]=this->datos[nfilas-i-1];
        resultado->datos[nfilas-i-1]=this->datos[i];
    }
    return *resultado;
}

void max(int &fila, int&columna){
    fila=0;
    columna=0;
    double max=datos[0][0];
    for(int i=0;i<nfilas;i++){
        for(int j=0;j<ncolumnas[i];j++)
            if(datos[i][j]>max){
                fila=i;
                columna=j;
            }
    }
}
void escribir(string nombrearchivo,string comentario){
    ofstream salida(nombrearchivo.c_str(),ios:out|ios:binary);
    if(salida){
        salida<< "MP"<<endl;
        if(comentario.compare("")==0)
            salida<< "#"<<comentario<<endl;
        salida<<nfilas<<endl;
        for(int i=0;i<nfilas;i++){
        for(int j=0;j<ncolumnas[i];j++)
            salida.write((char*)&ncolumnas[i],sizeof(int));
            salida.write((char*)datos[i],sizeof(double)*ncolumnas[i])M
        }
```

```
    }
    salida.close();
}

void leerArchivo(string nombrearchivo){
    ifstream entrada(nombrearchivoc_str(),ios:in|binary);
    char cabecera[100];
    liberarEspacio();
    if(entrada){
        entrada.getline(cabecera);
        //Si el archivo leido empieza con la cadena MP
        if(strcmp(cabecera,"MP"){
            char caracter;
            caracter=entrada.peek()
            if(str_cmp(caracter,"#")){//Si hay comentario, ignoramos la fila
                entrada.getline(cabecera);
            }
            fichero>>nfilas;
            fichero.ignore();

            ncolumnas=new int[nfilas];
            datos = new double*[nfilas];
            for(int i=0;i<nfilas;i++){
                fichero.read((char*)&ncolumnas[i],sizeof(int));
                datos[i]=new double[ncolumnas[i]];
                fichero.read((char*)datos[i],sizeof(double)*ncolumnas[i]);
            }
        }
    fichero.close();
    }
}

//**********************************
//**********Matriz.CPP**********
//**********************************
```