



Documento anónimo

Sobrecarga-de-Operadores-Matriz.pdf

Sobrecarga de Operadores Matriz



1º Metodología de la Programación



Grado en Ingeniería Informática



Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada

Maths
informática

Tan lejos como nunca, tan cerca
como siempre #QuédateEnCasa

CURSOS INTENSIVOS en JUNIO para todas las asignaturas de
Ingeniería Informática

615 29 80 22 91 399 45 49 C/Andrés Mellado, 88 duplicado

academia.maths www.mathsinformatica.com academia@mathsinformatica.com



```

#include <iostream>

using namespace std;

class Matriz{

private:

    int **matriz;
    int filas;
    int columnas;

public:

    Matriz(){

        filas = 0;
        columnas = 0;

        matriz = new int *[0];
        matriz[0] = new int [0];

    }

    Matriz(int **matriz, int filas, int columnas){

        this->filas = filas;
        this->columnas = columnas;
        this->matriz = matriz;

    }

    Matriz(const Matriz &matr){

        filas = matr.filas;
        columnas = matr.columnas;
        matriz = matr.matriz;

    }

    Matriz &operator = (const Matriz &m){ //Sobrecarga del operador =

        for(int i = 0; i < this->filas; i++){

            delete [] this->matriz[i];

        }

        delete [] this->matriz;

        this->filas = m.filas;
        this->columnas = m.columnas;
    }

```

```

matriz = new int *[this->filas];

for(int i = 0; i < this->filas; i++){

    matriz[i] = new int [this->columnas];

}

for(int i = 0; i < this->filas; i++){

    for(int j = 0; j < this->columnas; j++){

        this->matriz[i][j] = m.matriz[i][j];

    }

}

return *this;

}

friend bool operator == (const Matriz &mat1, const Matriz &mat2){ //Sobrecarga del operador
==

    bool resultado = true;

    if(mat1.filas != mat2.filas || mat1.columnas != mat2.columnas){

        resultado = false;

    }

    for(int i = 0; i < mat1.filas; i++){

        for(int j = 0; j < mat2.filas; j++){

            if(mat1.matriz[i][j] != mat2.matriz[i][j]){

                resultado = false;

            }

        }

    }

    return resultado;

}

```

Matriz operator + (const Matriz &mat2){ //Sobrecarga del operador +

Matriz resultado;

if(filas == mat2.filas && columnas == mat2.columnas){

resultado.filas = filas;

resultado.columnas = columnas;

resultado.matriz = new int *[filas];

for(int i = 0; i < filas; i++){

resultado.matriz[i] = new int [columnas];

}

for(int i = 0; i < filas; i++){

for(int j = 0; j < columnas; j++){

resultado.matriz[i][j] = matriz[i][j] + mat2.matriz[i][j];

}

}

}

return resultado;

}

Matriz operator - (const Matriz &mat2){ //Sobrecarga del operador -

Matriz resultado;

if(filas == mat2.filas && columnas == mat2.columnas){

resultado.filas = filas;

resultado.columnas = columnas;

resultado.matriz = new int *[filas];

for(int i = 0; i < filas; i++){

resultado.matriz[i] = new int [columnas];

}

for(int i = 0; i < filas; i++){

for(int j = 0; j < columnas; j++){



```
        resultado.matriz[i][j] = matriz[i][j] - mat2.matriz[i][j];
    }
}

return resultado;
}

void operator += (const Matriz &mat2){ //Sobrecarga del operador +=
    if(filas == mat2.filas && columnas == mat2.columnas){
        for(int i = 0; i < filas; i++){
            for(int j = 0; j < columnas; j++){
                matriz[i][j] = matriz[i][j] + mat2.matriz[i][j];
            }
        }
    }
}

void operator -= (const Matriz &mat2){ //Sobrecarga del operador -=
    if(filas == mat2.filas && columnas == mat2.columnas){
        for(int i = 0; i < filas; i++){
            for(int j = 0; j < columnas; j++){
                matriz[i][j] = matriz[i][j] - mat2.matriz[i][j];
            }
        }
    }
}
```

```
Matriz operator * (const Matriz &mat2){ //Sobrecarga del operador *
```

```
Matriz resultado;
int suma = 0;

if(columnas == mat2.filas){

    resultado.filas = filas;
    resultado.columnas = mat2.columnas;
    resultado.matriz = new int *[filas];

    for(int i = 0; i < filas; i++){

        resultado.matriz[i] = new int [mat2.columnas];

    }

    for(int i = 0; i < resultado.filas; i++){

        for(int j = 0; j < resultado.columnas; j++){

            for(int k = 0; k < columnas; k++){

                suma = suma + matriz[i][k] * mat2.matriz[k][j];

            }

            resultado.matriz[i][j] = suma;
            suma = 0;

        }

    }

}

return resultado;

}
```

```
friend ostream &operator << (ostream &os, const Matriz& matr){ //Sobrecarga del operador <<
```

```
for(int i = 0; i < matr.filas; i++){

    for(int j = 0; j < matr.columnas; j++){

        os << matr.matriz[i][j] << " ";

    }

    os << endl << endl;
```

```

}

return os;

}

friend istream &operator >> (istream &in, Matriz &matr){ //Sobrecarga del operador >>

    int val = 0;

    cout << "Introduzca el valor de sus filas y columnas: ";
    in >> matr.filas >> matr.columnas;

    matr.matriz = new int *[matr.filas];

    for(int i = 0; i < matr.filas; i++){

        matr.matriz[i] = new int [matr.columnas];

    }

    for(int i = 0; i < matr.filas; i++){

        for(int j = 0; j < matr.columnas; j++){

            cout << endl << "Introduzca el valor de la posicion [" << i << "][" << j << "]: ";
            in >> val;
            matr.matriz[i][j] = val;

        }

    }

    return in;
}

~Matriz(){

    for(int i = 0; i < filas; i++){

        delete [] matriz[i];

    }

    delete [] matriz;
    filas = 0;
    columnas = 0;

}

};

```

```

int main(){

    Matriz m1;
    Matriz m2;
    Matriz m3;

    cout << "Introduzca los datos de su matriz 1: " << endl;
    cin >> m1;

    cout << "\n";
    cout << "Introduzca los datos de su matriz 2: " << endl;
    cin >> m2;

    m3 = m1;

    cout << endl << endl;

    cout << "MATRIZ 1: " << endl << endl;
    cout << m1;

    cout << endl;

    cout << "MATRIZ 2: " << endl << endl;
    cout << m2;

    cout << "MATRIZ 3: " << endl << endl;
    cout << m3;

    if((m1 == m2) == true){

        cout << endl << "SI son iguales m1 y m2" << endl;

    }else{

        cout << endl << "NO son iguales m1 y m2" << endl;

    }

    if((m1 == m3) == true){

        cout << endl << "SI son iguales m1 y m3" << endl;

    }else{

        cout << endl << "NO son iguales m1 y m3" << endl;

    }

    cout << endl << "SUMA MATRIZ 1 + MATRIZ 2: " << endl << endl;

```




```
m3 = m1 + m2;
cout << m3;

cout << endl << "RESTA MATRIZ 1 - MATRIZ 2: " << endl << endl;

m3 = m1 - m2;
cout << m3;

cout << endl << "RESTA MATRIZ 2 - MATRIZ 1: " << endl << endl;

m3 = m2 - m1;
cout << m3;

cout << endl << "MULTIPLICACION MATRIZ 1 * MATRIZ 2: " << endl << endl;

m3 = m1 * m2;
cout << m3;

cout << endl << "MULTIPLICACION MATRIZ 2 * MATRIZ 1: " << endl << endl;

m3 = m2 * m1;
cout << m3;

cout << endl << "MATRIZ1 = MATRIZ1 + MATRIZ2: " << endl << endl;

m1 += m2;
cout << m1;

cout << endl << "MATRIZ1 NUEVA = MATRIZ1 NUEVA - MATRIZ2: " << endl << endl;

m1 -= m2;
cout << m1;

}
```