

**Practica Lenguage Ensamblador**

**Andres Felipe Cadena Velez**

**Erik Alexander Gonzalez Cardona**

**Prof. Hugo de Jesus Mesa Yepes**

**Periodo 2019-1**

**Medellin**

**Desarrollo de la práctica**

Inicialmente se busca una manera de visualizar las diferentes operaciones definidas en el método de gauss para invertir matrices, por esto surgen unas operaciones bases que aislamos en el archivo (ayuda.cpp), con base en estos se empezó el desarrollo del proyecto el cual nos dio ideas de como operar entre filas en ensamblador.

Las matrices se crean dentro del arreglo matrix que contiene tanto a la

matriz a invertir como la matriz identidad (se agrega dinámicamente) y se opera sobre ese arreglo para guardar posteriormente en el mismo la matriz inversa sobre-escribiendo la identidad inicial. También se hace uso de dos filas auxiliares (aux y aux1) y la fila independientes que tiene el valor de los términos independientes. Tanto la matriz(matrix) como aux, aux1 e independientes tienen el tamaño máximo para matrices 5x5 como predefinido cumpliendo los requisitos de la práctica pero este valor puede ser aumentado para resolver matrices mucho más grandes(NxN) modificando las constantes MAX\_COLUMNAS, MAX\_FILAS, MAX\_TIRA(tamaño completo de matrix) respectivamente.

Básicamente en el programa primero se verifica que cada elemento en la diagonal sea un uno(1) ,luego se hace cero(0) el resto de los valores de la columna hacia abajo y después se hace cero(0) el resto de los valores hacia arriba del uno(1) y esto genera la matriz inversa. luego esta matriz inversa se multiplica por el vector independientes y así obtenemos las soluciones.

En caso de que en algún elemento de la diagonal sea cero(0) después de hacer los intercambios si estos fueran posibles entonces la matriz que se desea invertir no tiene solución.

**Algoritmos**

A continuación, se explicará el funcionamiento de los algoritmos principales para operar entre filas.

**Suma:** Suma los valores de una fila (aux) con los valores de una fila (aux1) y guarda el resultado de la suma en la fila (aux).Esto se usó en el código principal pero en vez de trabajar con aux y aux1, se usaron las filas directamente de la matrix.

**Multiplicación:** Multiplica cada valor de una fila (aux) con una constante “c\_mul” y guarda el valor en la fila (aux). Esto se usó en el código pudiendo ser aux la fila superior o la fila inferior, para luego ésta ser sumada en alguna otra fila dentro de matrix.

**Intercambio de filas:** Intercambia los valores de dos filas (Superior e Inferior) de la matriz (Matrix). “pointery” (índice para fila superior) la cual además será la que tiene la posición más cercana al inicio del arreglo, y “p\_auxy” (índice fila inferior) la cual siempre estará debajo de pointery. Siendo el tope 0 y N lo más bajo para asignar a pointery o p\_auxy respectivamente.

**Reemplazar valores:** Reemplaza los valores de una fila en la matriz (Matrix) por los de otra fila (aux).

Esto se usó en el código para reemplazar algún aux que multiplicó constantes para hacer ceros o unos en alguna fila de la matriz (matrix).

**Herramientas**

Para facilitar el desarrollo de la práctica y el buen trabaja grupo, se requirió el uso de las siguientes herramientas, las cuales fueron sugeridas por los integrantes de este.

1. Lenguaje C++.
2. Compilador de Open Watcom V2.
3. Editor de texto Atom(Para codificar sobre el mismo código de forma simultánea).
4. Repositorio en Github (Para no correr riesgo de perder los cambios hechos).

**Nota:** *El Open Watcom debe descargarse desde la siguiente URL:* [*https://github.com/open-watcom/open-watcom-v2/releases*](https://github.com/open-watcom/open-watcom-v2/releases) *, se despliega “Assets” para elegir la arquitectura de su pc.*