# http://alimentos2014.epn.edu.ec/logo_epn_android.jpgESCUELA POLITÉCNICA NACIONL

**ESFOT**

* **Curso:** Gr1
* **Tarea:**  Deber
* **Nombres y apellidos:** Erick Omar Vargas Albán
* **Fecha de asignación:** 05/12/2018 **Fecha de entrega:** 12 /12/2018
* **Tarea:** Trabajo por actuaciones
* Implementar el algoritmo de búsqueda binaria para letras en cualquier lenguaje de programación
* **Resolución**

1. **Implementación**

# include <iostream>

using namespace::std;

// FUNCION MOVER\_TORRES

void Mover\_Torres(int N, int Origen, int Auxiliar, int Destino){

//cuando existe un disco o caso limite

if ( 1 == N ){ //condicion si esque solo existe un disco

cout << "Mueve el disco " << Origen << " a " << Destino << endl; // directo se mueve el disco de origen a destino

}

//cuando existe mas de un disco

if ( N > 1 ){ //condicion cuando es mas de un disco

Mover\_Torres(N - 1, Origen, Destino, Auxiliar); // declaro las variables origen destino y auxiliar como las torres y reduzco N-1 porque debe existir un disco base que vaya directo a la torre 3

cout <<"Mueve el disco " << N << " de " << Origen << " a " << Destino << endl; //impresión de los resultados que se darán al mover los discos

Mover\_Torres( N - 1, Auxiliar, Origen, Destino); //cambio de posicion cuando ya el disco n esta en su posición es decir el disco mas grande ya esta en la torre 3

}

}

//Funcion Principal

int main(){

int Discos; //variable para ingresar la cantidad de discos

cout << "Torres de Hanoi"<< endl;

cout << "Introduzca el numero de discos que quiere mover de Torre 1 a Torre 3 "<<endl;;

cin >> Discos; //Introducir el numero de discos

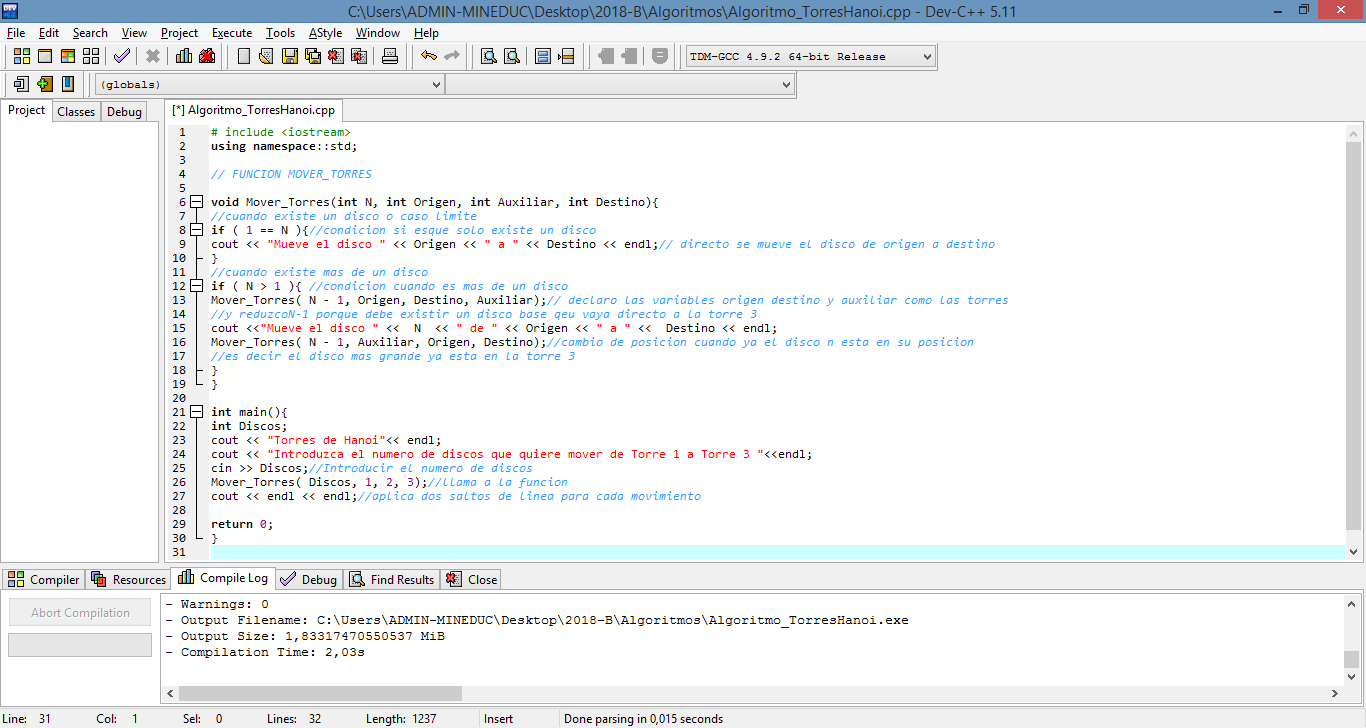
Mover\_Torres( Discos, 1, 2, 3); //llama a la funcion

cout << endl << endl; //aplica dos saltos de linea para cada movimiento

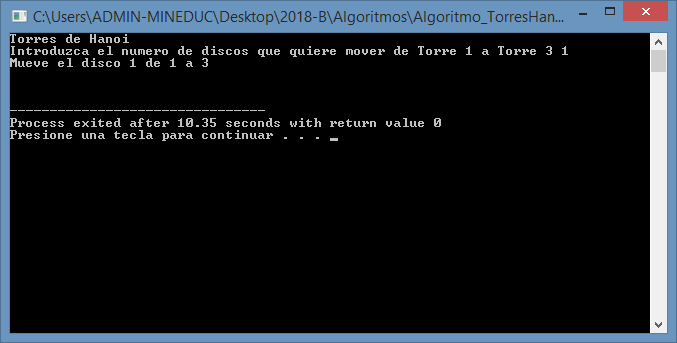
return 0;

}

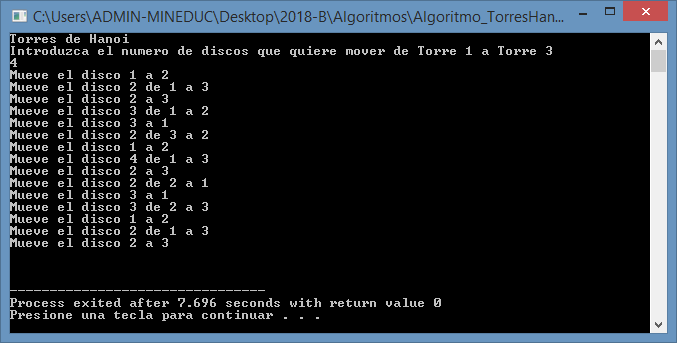
**Resultados**



**1 disco**



**Mas de 1 disco**



1. **Resolución**

* El problema se plantea para explicar la [recursividad](https://es.wikipedia.org/wiki/Recursividad).
* Numeramos los discos desde 1 hasta n,
* Aplicamos el método divide y vencerás es decir resolvemos el problema grande en soluciones más pequeñas o en el caso de las torres pasar los discos de menor tamaño hasta que el disco más grande(n) llegue a su punto de origen
* El número de movimientos mínimo a realizar para resolver el problema de este modo es de 2*n* – 1, siendo *n* el número de discos.
* Existe un cambio en el primer movimiento cuando el disco es par o impar

Existen 3 columnas serán variable

Columan1 =Origen

Columna2 =Auxiliar

Columna3=Final

Ahora como principal condición si existe 1 disco

**SI** Origen==1{

**Mover** disco a destino //como solo existe un disco no hay necesidad de ocupar la variable auxiliar

**Terminar**

}

**Si no** {

**Hanói** ({1,….n-1} {\displaystyle \scriptstyle \{1,\dots ,n-1\}}origen, destino, auxiliar)  //mover todas las fichas menos la más grande (n) a la varilla auxiliar

**Mover** disco n a destino  //mover la ficha grande hasta la varilla final

**Hanói** (auxiliar, origen, destino) //mover todas las fichas restantes, 1...n–1, encima de la ficha grande (n)

}

**Terminar**