**WARSHALL’S CODE:**

public const int INF = 11111;

private static void Print(int[,] matrix, int transitive)

{

Console.WriteLine("transitive closure is:");

for (int i = 0; i < transitive; ++i)

{

for (int j = 0; j < transitive; ++j)

{

if (matrix[i, j] == INF)

Console.Write("INF".PadLeft(7));

else

Console.Write(matrix[i, j].ToString().PadLeft(7));

}

}

}

public static void FloydWarshall(int[,] closure, int transitive)

{

int[,] matrix = new int[transitive, transitive];

for (int i = 0; i < transitive; ++i)

for (int j = 0; j < transitive; ++j)

matrix[i, j] = closure[i, j];

for (int k = 0; k < transitive; ++k)

{

for (int i = 0; i < transitive; ++i)

{

for (int j = 0; j < transitive; ++j)

{

if (matrix[i, k] + matrix[k, j] < matrix[i, j])

matrix[i, j] = matrix[i, k] + matrix[k, j];

}

}

}

Print(matrix, transitive);

Console.ReadKey();

}

}

}

**EUCLIDEAN CODE :**

class gcdd

{

public int gcd(int a, int b)

{

if (b == 0)

{

return a;

}

else

{

return gcd(b, a % b);

}

}

static void Main(string[] args)

{

int a;

int b;

Console.Write("Enter first number: ");

a = int.Parse(Console.ReadLine());

Console.Write("Enter second number: ");

b = int.Parse(Console.ReadLine());

gcdd obj = new gcdd();

int g = obj.gcd(a, b);

Console.WriteLine("GCD of {0} and {1} is equal to {2}", g);

Console.ReadKey();

}

**FAST MODULAR EXPONENTIATION CODE :**

static int expo(int B, int K, int M)

{

int res = 1;

B = B % M;

while (K > 0)

{

if ((K & 1) == 1)

res = (res \* B) % M;

K = K >> 1;

B = (B \* B) % M;

}

return res;

}

public static void Main()

{

int B = 2; //base

int K = 7; //expo

int M = 15; // mod

Console.WriteLine("exponential is " + expo(B, K, M));

Console.ReadKey();

}