**Cryptography**

### -Lab report



Submitted By: Deshant Devkota

Submitted To: Suresh Thapal (Lecturer, Cryptography)

Vedas College, Lalitpur, Nepal

**Tribhuvan University**

**Department of Bsc Csit**

Submitted Date: Saturday, June 5, 2021

Submitted Lab Reports:

1. To implement the Euclidean algorithm to find the GCD.
2. Implement Extended Euclidean Algorithm to find the multiplicative inverse in GF(p).

## LAB 5

Euclidean Algorithm

**1. Objectives:**

In this lab we were to implement the Euclidean Algorithm to find the GCD of two given numbers.

**2.Introduction:**

The Euclidean algorithm is basically a continual repetition of the division algorithm for integers. The point is to repeatedly divide the divisor by the remainder until the remainder is 0. The GCD is the last non-zero remainder in this algorithm.

The Euclidean Algorithm for finding GCD(A,B) is as follows:

* If A = 0 then GCD(A,B)=B, since the GCD(0,B)=B, and we can stop.
* If B = 0 then GCD(A,B)=A, since the GCD(A,0)=A, and we can stop.
* Write A in quotient remainder form (A = B⋅Q + R)
* Find GCD(B,R) using the Euclidean Algorithm since GCD(A,B) = GCD(B,R)

**3.Code:**

#include<iostream>

using namespace std;

int gcd(int a, int b)

{

int r;

while(b!=0)

{

r=a%b;

a=b;

b=r;

}

return a;

}

int main()

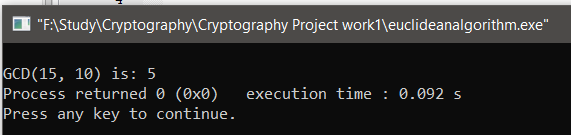
{

int a=15,b=10;

cout<<"\nGCD("<<a<<", "<<b<<") is: "<<gcd(a,b);

}

**4.Output:**



## LAB 6

Extended Euclidean Algorithm

**1. Objectives:**

In this lab we were to implement Extended Euclidean Algorithm to find the multiplicative inverse in GF(p).

**2.Introduction:**

The Extended Euclidean algorithm is an algorithm to compute integers x and y such that: ax + by = gcd(a,b), given a and b. The existence of such integers is guaranteed by Bézout's lemma.

The extended Euclidean algorithm can be viewed as the reciprocal of modular exponentiation.

Extended euclidean can be used to find the multiplicative inverse in GF(p). For gcd(m,b)=1, an extended euclidean algorithm can be used to calculate the multiplicative inverse of b in GF(m). i.e a integer “s” such that: s \* b (mod m) = 1, s is called the multiplicative inverse of b in GF(m).

Extended Euclid(m,b):

1. (A1, A2, A3) <-(1, 0, m); (B1, B2, B3) <- (0, 1, b)

2. if B3 = 0 return A3 = gcd(m, b); no inverse

3. if B3 = 1 return B3 = gcd(m, b); B2 = b¹ mod m

4. Q=Floor(A3/B3).

5. (T1, T2, T3)<- (A1-QB1, A2-QB2, A3-QB3)

6. (A1, A2, A3)<-(B1, B2, B3)

7. (B1, B2, B3)<- (T1, T2, T3)

8. goto 2

**3.Code:**

#include<iostream>

#include<stdlib.h>

using namespace std;

int extendedEuclid(int m, int b)

{

int a1=1, a2=0, a3=m;

int b1=0, b2=1, b3=b;

int t1, t2, t3;

int q;

while(1)

{

q=int(a3/b3);

t1=a1-q\*b1;

t2=a2-q\*b2;

t3=a3-q\*b3;

a1=b1;

a2=b2;

a3=b3;

b1=t1;

b2=t2;

b3=t3;

if(b3==0||b3==1) break;

}

if(b3==0)

{

cout<<"\nInverse Doesn't Exist"<<endl;

exit(1);

}

else

{

return b1;

}

}

int main()

{

int a=550, b=1759;

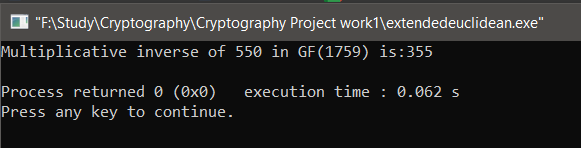
int ans=extendedEuclid(a,b);

cout<<"Multiplicative inverse of "<<a<<" in GF("<<b<<") is:"<<ans<<endl;

return 0;

}

**4.Output:**

****

============================End===============================