Information Security
Lab File
ITD03

Submitted by : Shiv Kumar
IT-2
5th Semester
2016UIT2563

Index

Practical 1
Shift Cipher

```
#include <bits/stdc++.h>

using namespace std;

int main()
{
	int T;
	cout<<"Please enter the no. of test cases:";
	cin>>T;
	int K=2;
	while(T--)
	{
		string input,output;
		cout<<"Message: ";
		cin>>input;

		///ENCRYPTION
		cout<<"Encrypted Data: ";
		for(int i=0;i<input.size();i++)
			output+=(input[i]-'a'+K)%26+'a';
		cout<<output<<endl;

		///DECRYPTION
		cout<<"Descrypted Data: ";
		input.clear();
		for(int i=0;i<output.size();i++)
			input+=(output[i]-'a'-2+26)%26+'a';
		cout<<input<<endl;
	}
}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ shift_cipher.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Please enter the no. of test cases:2
Message: abcd
Encrypted Data: cdef
Descrypted Data: abcd
Message: yzab
Encrypted Data: abcd
Descrypted Data: yzab
```

Practical 2
Multiplicative cipher

```
#include <bits/stdc++.h>

using namespace std;

int inverse(int N)
{
	int r1=26,r2=N;
	int t1=0,t2=1;
	while(r2>0)
	{
		int q=r1/r2;
		int r=r1-q*r2;
		r1=r2;
		r2=r;
		int t=t1-q*t2;
		t1=t2;
		t2=t;
	}
	return t1;
}

int main()
{
	int T;
	cout<<"Please enter the no. of testcases:";
	cin>>T;
	int K=3;
	int inv=inverse(K);

	while(T--)
	{
		string input,output;
		cout<<"Message: ";
		cin>>input;

		///ENCRYPTION
		cout<<"Encrypted Data: ";
		for(int i=0;i<input.size();i++)
			output+=((input[i]-'a')*K)%26+'a';
		cout<<output<<endl;

		///DECRYPTION
		cout<<"Descrypted Data: ";
		input.clear();
		for(int i=0;i<output.size();i++)
			input+=((output[i]-'a')*inv)%26+'a';
		cout<<input<<endl;
```

```
    }
}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ multi_cipher.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Please enter the no. of testcases:2
Message: abcd
Encrypted Data: adgj
Descrypted Data: abcd
Message: yzab
Encrypted Data: uxad
Descrypted Data: yzab
```

Practical 3
Affine cipher

```
#include <bits/stdc++.h>
using namespace std;


int inverse(int N)
{
	int r1=26,r2=N;
	int t1=0,t2=1;
	while(r2>0)
	{
		int q=r1/r2;
		int r=r1-q*r2;
		r1=r2;
		r2=r;
		int t=t1-q*t2;
		t1=t2;
		t2=t;
	}
	return t1;
}
int main()
{
	int T;
	cout<<"Please enter the no. of testcases:";
	cin>>T;
	int K1=3,K2=10;
	int inv=inverse(K1);

	while(T--)
	{
		string input,output;
		cout<<"Message: ";
		cin>>input;

		///ENCRYPTION
		cout<<"Encrypted Data: ";
		for(int i=0;i<input.size();i++)
			output+=(((input[i]-'a')*K1)%26 + K2)%26+'a';
		cout<<output<<endl;

		///DECRYPTION
		cout<<"Descrypted Data: ";
		input.clear();
		for(int i=0;i<output.size();i++)
			input+=(((output[i]-'a'-K2+26)%26)*inv)%26+'a';
		cout<<input<<endl;
	}
```

}

```
Shivs-MacBook-Air:infosec championballer$ g++ affine_cipher.cpp -o run
Shivs-MacBook-Air:infosec championballer$ ./run
Please enter the no. of testcases:2
Message: abcd
Encrypted Data: knqt
Descrypted Data: abcd
Message: yzab
Encrypted Data: ehkn
Descrypted Data: yzab
```

Practical 4
Playfair cipher

```
#include <bits/stdc++.h>

using namespace std;


int inverse(int N)
{
	int r1=26,r2=N;
	int t1=0,t2=1;
	while(r2>0)
	{
		int q=r1/r2;
		int r=r1-q*r2;
		r1=r2;
		r2=r;
		int t=t1-q*t2;
		t1=t2;
		t2=t;
	}
	return t1;
}
int main()
{
	int T;
	cout<<"Please enter the no. of testcases:";
	cin>>T;

	while(T--)
	{
		string input,output,key;
		cout<<"Message: ";
		cin>>input;
		cout<<"Key: ";
		cin>>key;


		///ENCRYPTION
		cout<<"Encrypted Data: ";
		int plain[input.size()];
		for(int i=0;i<input.size();i++)
			plain[i]=input[i]-'a';
		int keyarr[input.size()];
		for(int i=0;i<input.size();i++)
			keyarr[i]=key[i%key.size()]-'a';

		int cipher[input.size()];
		for(int i=0;i<input.size();i++)
```

```
            cipher[i]=(plain[i]+keyarr[i])%26;

        for(int i=0;i<input.size();i++)
            output+=cipher[i]+'a';

        cout<<output<<endl;

        ///DECRYPTION
        cout<<"Descrypted Data: ";
        input.clear();
        for(int i=0;i<output.size();i++)
            cipher[i]=output[i]-'a';
        for(int i=0;i<output.size();i++)
            keyarr[i]=key[i%key.size()]-'a';
        for(int i=0;i<output.size();i++)
            plain[i]=(cipher[i]-keyarr[i]+26)%26;
        for(int i=0;i<output.size();i++)
            input+=plain[i]+'a';

        cout<<input<<endl;
    }
}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ playfair.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Please enter the no. of testcases:1
Message: hidethegoldinthetreeslump
Key: playfairexample
Encrypted Data: wtdcyhmxsiduceltercjstlqm
Descrypted Data: hidethegoldinthetreeslump
```

Practical 5
Hill Cipher

```
#include <bits/stdc++.h>
using namespace std;

void multiply(char a[10][10],char key[10][10],int N,int M)
{
    char c[10][10];
    for(int i=0;i<N;i++)
        for(int j=0;j<M;j++)
        {
            int sum=0;
            for(int k=0;k<M;k++)
                sum=(sum+(((a[i][k]-'a')%26)*((key[k][j]-'a')%26))%26)%26;
            c[i][j]=sum+'a';
        }
    for(int i=0;i<N;i++)
        for(int j=0;j<M;j++)
            a[i][j]=c[i][j];
}

// 9 7 11 13 4 7 5 6 2 21 14 9 3 23 21 8
void hillcipher()
{
    string s;
    cout<<"Enter String: ";
    cin>>s;
    cout<<"Enter M for M X M key matrix: ";
    int M;
    cin>>M;
    char key[10][10];
    cout<<"Enter the key matrix:";
    for(int i=0;i<M;i++)
        for(int j=0;j<M;j++)
            cin>>key[i][j];
    char a[10][10];
    int row=ceil(double(s.size())/M);
    for(int i=0,index=0;i<row;i++)
        for(int j=0;j<M;j++,index++)
            if(index<s.size())
                a[i][j]=s[index];
            else
                a[i][j]='a';

    multiply(a,key,row,M);
    string output="";
    for(int i=0,index=0;i<row;i++)
        for(int j=0;j<M;j++,index++)
            if(index<s.size())
```

```
            output+=a[i][j];
    cout<<"Encrpyted String: "<<output<<endl;
}
int main()
{
    hillcipher();
}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ hill.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Enter String: hello
Enter M for M X M key matrix: 4
Enter the key matrix:i g k m d g e f b u n i c w u h
Encrpyted String: xihji
```

Practical 6
Vignere Cipher

```
#include<bits/stdc++.h>

using namespace std;

int main()
{
	int t;
	cout<<"Please enter the no. of testcases:";
	cin>>t;

	while(t--)
	{
		string plain;
		cout<<"Enter plain text:";
		cin>>plain;

		string key;
		cout<<"Enter key:";
		cin>>key;
		string encode = plain;

		for(int i=0,j=0;i<plain.length();i++)
		{
			encode[i]=(encode[i]-97+key[j]-97)%26+97;
			j = (j+1)%key.length();

		}

		cout<<"Encoded string is:"<<encode<<endl;

		string decode = encode;
		for(int i=0,j=0;i<plain.length();i++)
		{
			decode[i] = (((decode[i]-97)+26-(key[j]-97))%26)+97;
			j =  (j+1)%key.length();
		}
```

```
            cout<<"Decoded string is:"<<decode<<endl;
        }
}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ vignere_cipher.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Please enter the no. of testcases:2
Enter plain text:helloshiv
Enter key:vig
Encoded string is:cmrgwycqb
Decoded string is:helloshiv
Enter plain text:helloboy
Enter key:vig
Encoded string is:cmrgwhjg
Decoded string is:helloboy
```

Practical 7
Rail fence-row and column transformation

```
#include <bits/stdc++.h>
using namespace std;

void railfence_row()
{
    string s;
    int k;
    cout<<"Enter string: ";
    cin>>s;
    cout<<"KEY(2 for keyless): ";
    cin>>k;
    char a[k][s.size()];
    for(int i=0;i<k;i++)
        for(int j=0;j<s.size();j++)
            a[i][j]=0;
    int up=0,down=k-1,i=0,j=0,index=0;
    bool flag=true;
    while(index<s.size())
    {
        a[i][j]=s[index];
        index++;
        if(flag && i==down)
            flag=false;
        else if(!flag && i==up)
            flag=true;
        if(flag)
            i++;
        else
            i--;
        j++;
    }
    string output;
    for(int i=0;i<k;i++)
        for(int j=0;j<s.size();j++)
            if(a[i][j]!=0)
                output+=a[i][j];
    cout<<"ENCRYPTED String: "<<output<<endl;

    s.clear();

    for(int i=0;i<k;i++)
        for(int j=0;j<output.size();j++)
            a[i][j]=0;
    up=0;
    down=k-1;
    i=0;
    j=0;
```

```
    index=0;

    flag=true;
    while(index<output.size())
    {
        a[i][j]='*';
        index++;
        if(flag && i==down)
            flag=false;
        else if(!flag && i==up)
            flag=true;
        if(flag)
            i++;
        else
            i--;
        j++;
    }

    index=0;
    for(int i=0;i<k;i++)
        for(int j=0;j<output.size();j++)
            if(a[i][j]=='*')
            {
                a[i][j]=output[index];
                index++;
            }

    up=0;
    down=k-1;
    i=0;
    j=0;
    index=0;

    flag=true;
    while(index<output.size())
    {
        s+=a[i][j];
        index++;
        if(flag && i==down)
            flag=false;
        else if(!flag && i==up)
            flag=true;
        if(flag)
            i++;
        else
            i--;
        j++;
    }

    cout<<"DECRYTED String: "<<s<<endl;
}
```

```
void railfence_column()
{
    string s;
    int k;
    cout<<"Enter string: ";
    cin>>s;
    cout<<"KEY(2 for keyless): ";
    cin>>k;
    char a[s.size()][k];
    for(int i=0;i<s.size();i++)
        for(int j=0;j<k;j++)
            a[i][j]=0;
    int up=0,down=k-1,i=0,j=0,index=0;
    bool flag=true;
    while(index<s.size())
    {
        a[j][i]=s[index];
        index++;
        if(flag && i==down)
            flag=false;
        else if(!flag && i==up)
            flag=true;
        if(flag)
            i++;
        else
            i--;
        j++;
    }
    string output;
    for(int i=0;i<k;i++)
        for(int j=0;j<s.size();j++)
            if(a[j][i]!=0)
                output+=a[j][i];
    cout<<"ENCRYPTED String: "<<output<<endl;

    s.clear();

    for(int i=0;i<output.size();i++)
        for(int j=0;j<k;j++)
            a[i][j]=0;
    up=0;
    down=k-1;
    i=0;
    j=0;
    index=0;

    flag=true;
    while(index<output.size())
    {
        a[j][i]='*';
        index++;
        if(flag && i==down)
```

```
            flag=false;
        else if(!flag && i==up)
            flag=true;
        if(flag)
            i++;
        else
            i--;
        j++;
    }

    index=0;
    for(int i=0;i<k;i++)
        for(int j=0;j<output.size();j++)
            if(a[j][i]=='*')
            {
                a[j][i]=output[index];
                index++;
            }

    up=0;
    down=k-1;
    i=0;
    j=0;
    index=0;

    flag=true;
    while(index<output.size())
    {
        s+=a[j][i];
        index++;
        if(flag && i==down)
            flag=false;
        else if(!flag && i==up)
            flag=true;
        if(flag)
            i++;
        else
            i--;
        j++;
    }
    cout<<"DECRYTED String: "<<s<<endl;
}

int main()
{
    railfence_column();
}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ railfence.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Enter string: helloshiv
KEY(2 for keyless): 2
ENCRYPTED String: hlohvelsi
DECRYTED String: helloshiv
```

Practical 8
DES

```
#include <bits/stdc++.h>
using namespace std;

void initial_permutation(string &input)
{
	string temp;
	int
p[64]={58,50,42,34,26,18,10,2,60,52,44,36,28,20,12,4,62,54,46,38,30,22,14,6,64,56,48,40
,
32,24,16,8,57,49,41,33,25,17,9,1,59,51,43,35,27,19,11,3,61,53,45,37,29,21,13,5,63,55,47
,39,31,23,15,7};
	for(int i=0;i<64;i++)
		temp+=input[p[i]-1];
	input=temp;
}
void final_permutation(string &input)
{
	int
p[64]={58,50,42,34,26,18,10,2,60,52,44,36,28,20,12,4,62,54,46,38,30,22,14,6,64,56,48,40
,
32,24,16,8,57,49,41,33,25,17,9,1,59,51,43,35,27,19,11,3,61,53,45,37,29,21,13,5,63,55,47
,39,31,23,15,7};
	int invp[64];
	for(int i=0;i<64;i++)
		invp[p[i]-1]=i+1;

	string temp;
	for(int i=0;i<64;i++)
		temp+=input[invp[i]-1];
	input=temp;
}
string key_generation(string cipher,int *numberofshift,int round)
{
	///SHIFT
	for(int i=0;i<numberofshift[round];i++)
	{
		char temp=cipher[0];
		for(int j=0;j<27;j++)
			cipher[j]=cipher[j+1];
		cipher[27]=temp;

		temp=cipher[28];
		for(int j=28;j<55;j++)
			cipher[j]=cipher[j+1];
		cipher[55]=temp;
	}
```

```
	///COMPRESSION P-BOX
	int
p[48]={14,17,11,24,1,5,3,28,15,6,21,10,23,19,12,4,26,8,16,7,27,20,13,2,41,52,31,37,47,55
,30,40,51,45,33,48,44,49,39,56,34,53,46,42,50,36,29,32};
	string x;
	for(int i=0;i<48;i++)
		x+=cipher[p[i]-1];
	return x;
}
string DESfunc(string k,string s)
{
	///EXPANSION
	string temp;
	for(int i=0;i<32;)
	{
		temp+=s[(i-1+32)%32];
		for(int j=0;j<4;j++,i++)
			temp+=s[i];
		temp+=s[i%32];
	}

	///KEY MIXER/WHITENER
	for(int i=0;i<32;i++)
		if(temp[i]==k[i])
			temp[i]='0';
		else
			temp[i]='1';

	///COMPRESSION S-BOX(Assuming S1=S2...=S8)
	int sbox[4]
[16]={14,4,13,1,2,15,11,8,3,10,6,12,5,9,0,7,0,15,7,4,14,2,13,10,3,6,12,11,9,5,3,8,4,1,14,8,
13,6,2,11,15,12,9,7,3,10,5,0,15,12,8,2,4,9,1,7,5,11,3,14,10,0,6,13};
	string final;
	for(int i=0;i<48;i+=6)
	{
		string x;
		for(int j=i;j<i+6;j++)
			x+=temp[j];
		int row=(x[5]-'0')*2+x[0]-'0';
		int col=(x[4]-'0')*8+(x[3]-'0')*4+(x[2]-'0')*2+(x[1]-'0');
		int num=sbox[row][col];
		x.clear();
		for(int j=0;j<4;j++,num/=2)
			x+=num%2+'0';
		for(int j=3;j>=0;j--)
			final+=x[j];
	}

	///STRAIGHT P-BOX
	int
p[32]={16,7,20,21,29,12,28,17,1,15,23,26,5,18,31,10,2,8,24,14,32,27,3,9,19,13,30,6,22,11
,4,25};
```

```
	temp.clear();
	for(int i=0;i<32;i++)
		temp+=final[p[i]-1];
	return temp;
}
int main()
{
	///ENCRYPTION
	int numberofshift[16];
	for(int i=0;i<16;i++)
		if(i==0 || i==1 || i==8 || i==15)
			numberofshift[i]=1;
		else
			numberofshift[i]=2;
	for(int i=1;i<16;i++)
		numberofshift[i]+=numberofshift[i-1];

	cout<<endl<<"ENCRYPTION: "<<endl<<endl;
	string input,cipherkey;
	cout<<"Enter Input String(64 bits): ";
	cin>>input;
	cout<<"Enter Cipher Key(56 bits): ";
	cin>>cipherkey;
	cout<<"INPUT: "<<input<<endl;
	initial_permutation(input);
	cout<<"INITIAL PERMUTATION: "<<input<<endl;

	for(int i=0;i<16;i++)
	{
		string key=key_generation(cipherkey,numberofshift,i);
		string L,R;
		for(int i=0;i<32;i++)
			L+=input[i];
		for(int i=32;i<64;i++)
			R+=input[i];

		string temp=DESfunc(key,R);

		for(int i=0;i<32;i++)
			if(temp[i]==L[i])
				L[i]='0';
			else
				L[i]='1';

		if(i!=15)
			swap(L,R);
		for(int i=0;i<32;i++)
			input[i]=L[i];
		for(int i=32;i<64;i++)
			input[i]=R[i-32];

		cout<<"After Round #"<<i+1<<": "<<input<<endl;
```

```
	}
	final_permutation(input);
	cout<<"FINAL PERMUTATION: "<<input<<endl;


	///DECRYPTION
	cout<<endl<<"DECRYPTION: "<<endl<<endl;
	initial_permutation(input);
	cout<<"INITIAL PERMUTATION: "<<input<<endl;
	for(int i=0;i<16;i++)
	{
		string key=key_generation(cipherkey,numberofshift,15-i);
		string L,R;
		for(int i=0;i<32;i++)
			L+=input[i];
		for(int i=32;i<64;i++)
			R+=input[i];

		if(i!=0)
			swap(L,R);

		string temp=DESfunc(key,R);

		for(int i=0;i<32;i++)
			if(temp[i]==L[i])
				L[i]='0';
			else
				L[i]='1';

		for(int i=0;i<32;i++)
			input[i]=L[i];
		for(int i=32;i<64;i++)
			input[i]=R[i-32];

		cout<<"After Round #"<<i+1<<": "<<input<<endl;
	}
	final_permutation(input);
	cout<<"FINAL PERMUTATION: "<<input<<endl;
}
```

Practical 9
RSA

```
#include <bits/stdc++.h>

using namespace std;

long long gcd(long long a,long long b)
{
	if(a==0 && b==0)return 0;
	if(b>a)return gcd(b,a);
	if(b==0)return a;

	return gcd(b,a%b);
}
int findInverse(int N,int M)
{
	N=N%M;
	int t1=0,t2=1,r1=M,r2=N;
	while(r2)
	{
		int q=r1/r2,r=r1%r2;
		r1=r2;
		r2=r;
		int t=t1-q*t2;
		t1=t2;
		t2=t;
	}
	return (t1+M)%M;
}
int modularExpo(int X,int N,int M)
{
	if(N==0)
		return 1;
	if(N%2==0)
		return modularExpo((X*X)%M,N/2,M);
	return (X*modularExpo(X,N-1,M))%M;
}
int main()
{
	int P;
	cout<<"Enter Plain text: ";
	cin>>P;
	///KEY GENERATION-->

	///1. CHOOSE TWO PRIMES p and q
	int p=7,q=11;
	///2. n=pq
	int n=p*q;
```

```
	///3. choose e such that gcd(e,Q(n))=1.
	int e=2;
	int Qn=(p-1)*(q-1);
	while(gcd(e,Qn)!=1 && e<Qn)
		e++;
	///4. d=inverse(e) mod Qn
	int d=findInverse(e,Qn);
	///5.Public Key=e, Private key=d

	///ENCRYPTION-->
	int C=modularExpo(P%n,e,n);
	cout<<"Cipher Text: "<<C<<endl;

	///DECRYPTION-->
	P=modularExpo(C%n,d,n);
	cout<<"Plain Text: "<<P<<endl;



}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ rsa.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Enter Plain text: 17
Cipher Text: 52
Plain Text: 17
```

Practical 10
Elgamal

```
#include <bits/stdc++.h>
using namespace std;

int findInverse(int N,int M)
{
	N=N%M;
	int t1=0,t2=1,r1=M,r2=N;
	while(r2)
	{
		int q=r1/r2,r=r1%r2;
		r1=r2;
		r2=r;
		int t=t1-q*t2;
		t1=t2;
		t2=t;
	}
	return (t1+M)%M;
}
int modularExpo(int X,int N,int M)
{
	if(N==0)
		return 1;
	if(N%2==0)
		return modularExpo((X*X)%M,N/2,M);
	return (X*modularExpo(X,N-1,M))%M;
}
int main()
{
	int P;
	cout<<"Plain Text: ";
	cin>>P;
	///KEY GENERATION-->
	///1. Take a big prime number p
	int p=11;
	///2. Decide a d such that 1<=d<=p-2
	int d=3;
	///3. Find e1 such that e1 is primitive root of p
	///Primitive root means if x^n = y, n>=1 AND n<p, then y should cover all numbers
1 to p-1
	int e1=2;
	while(e1<p)
	{
		bool h[p];
		for(int i=1;i<p;i++)
			h[i]=0;
		for(int i=e1,j=1;j<p;j++,i=(i*e1)%p)
			h[i]=1;
```

```
            bool flag=true;
            for(int i=1;i<p;i++)
                if(!h[i])
                {
                    flag=false;
                    break;
                }
            if(flag)
                break;
            e1++;
        }
        ///4. Find e2=(e1^d)%p
        int e2=modularExpo(e1%p,d,p);
        ///5. Public Key: e1,e2,p Private Key: d,p


        ///ENCRYPTION-->
        ///Find a random number r
        int r=4;
        int C1=modularExpo(e1%p,r,p);
        int C2=(P*modularExpo(e2%p,r,p))%p;
        cout<<"Cipher Text: "<<C1<<" and "<<C2<<endl;

        ///DECRYPTION-->
        P=(C2*findInverse(modularExpo(C1,d,p),p))%p;
        cout<<"Plain Text: "<<P<<endl;

}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ elgamal.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Plain Text: 17
Cipher Text: 5 and 2
Plain Text: 6
```

Program 11
Rabin

```
#include <bits/stdc++.h>
using namespace std;

int findInverse(int N,int M)
{
	N=N%M;
	int t1=0,t2=1,r1=M,r2=N;
	while(r2)
	{
		int q=r1/r2,r=r1%r2;
		r1=r2;
		r2=r;
		int t=t1-q*t2;
		t1=t2;
		t2=t;
	}
	return (t1+M)%M;
}
int modularExpo(int X,int N,int M)
{
	if(N==0)
		return 1;
	if(N%2==0)
		return modularExpo((X*X)%M,N/2,M);
	return (X*modularExpo(X,N-1,M))%M;
}
void chinese_remainder(int a1,int a2,int m1,int m2)
{
	int M=m1*m2;
	int M1=M/m1,M2=M/m2;
	int invM1=findInverse(M1,m1);
	int invM2=findInverse(M2,m2);

	int x=(a1*M1*invM1 + a2*M2*invM2)%M;
	cout<<x<<" ";
}
int main()
{
	int P;
	cout<<"Plain Text: ";
	cin>>P;
	///KEY GENERATION-->
	///1. Select two large numbers p and q.
	int p=23,q=7;
	///2. n=pq
	int n=p*q;
	///3. Public Key: n, Private Key: p,q
```

```
        ///ENCRYPTION-->
        int C=(P*P)%n;
        cout<<"Cipher Text: "<<C<<endl;

        ///DECRYPTION-->
        int a1=modularExpo(C%p,(p+1)/4,p);
        int a2=p-a1;
        int b1=modularExpo(C%q,(q+1)/4,q);
        int b2=q-b1;

        chinese_remainder(a1,b1,p,q);
        chinese_remainder(a1,b2,p,q);
        chinese_remainder(a2,b1,p,q);
        chinese_remainder(a2,b2,p,q);
        cout<<"are the possible Plain Text"<<endl;
}
```

```
[Shivs-MacBook-Air:infosec championballer$ g++ rabin.cpp -o run
[Shivs-MacBook-Air:infosec championballer$ ./run
Plain Text: 17
Cipher Text: 128
144 52 109 17 are the possible Plain Text
```