

TUTORIAL / PRACTICAL NO.

INDEX

S.NO	PRACTICALS	DATE	SIGN.	REMARKS
1.	Write program to implement Caesar cipher technique.			
2.	Write program to implement polyalphabetic cipher.			
3.	Write program to implement Vigenere cipher.			
4.	Write program to implement Rail Fence technique.			
5.	Write program to implement simple Columnar technique			
6.	Write program to implement RSA algorithm			
7.	Write program to implement Euclidean algorithm			
8.	Write steps to ensure security of web browser			

TUTORIAL / PRACTICAL NO. 1

Objective: Write a program to implement caesar cipher technique.

Program:

```
#include <iostream.h>
#include <string.h>
using namespace std;
int main()
{
    cout << "Enter the message : ";
    cin.getline(msg, 100);
    int i, j, len, ch, key;
    cout << "Enter key : ";
    cin >> key;
    len = strlen(msg);
    cout << "Enter your choice \n 1. Encryption
\n 2. Decryption \n";
    cin >> ch;
    if (ch == 1)
        char c;
        for (int i = 0; msg[i] != '\0'; ++i)
        {
            c = msg[i];
            if (c >= 'a' && c <= 'z')
            {
                c = c + key;
                c = c - 'z' + 'a' - 1;
            }
            msg[i] = c;
        }
}
```

else if (c >= 'A' && c <= 'Z')

```
{   c = c + key;
    if (c > 'Z')
```

for decryption OUTPUT

Enter the message : how ARE you

Enter key : 3

Enter your choice

1. Encryption

2. Decryption

1

Encrypted message : RQZ DUH bzx

for decryption

Enter the message : RQZ DUH bzx

Enter Key : 3

Enter your choice

1. Encryption

2. Decryption

2

Decrypted message : how ARE you

TUTORIAL / PRACTICAL NO.

```
{   c = c - 'z' + 'A' - 1;
    }   msg[i] = ch;
}
cout << "Encrypted message : " << msg << endl;
} else if (ch == 2)
{
    char c;
    for (int i=0; msg[i] != '\0'; ++i)
    {
        c = msg[i];
        if (c >= 'a' && c <= 'z')
        {
            c = c - key;
            if (c < 'a')
            {
                c = c + 'z' - 'a' + 1;
            }
            msg[i] = ch;
        }
        else if (c >= 'A' && c <= 'Z')
        {
            c = c - key;
            if (c < 'A')
            {
                c = c + 'Z' - 'A' + 1;
            }
            msg[i] = c;
        }
    }
    cout << "Decrypted message : " << msg
    << endl;
}
```

OUTPUT

Enter the plain text
todayismybirthday

Enter key
november

new key is : novembernovember

Cipher Text : gcycK jwdlpdvf iherl
plain Text : todayismybirthday

TUTORIAL / PRACTICAL NO. 2

Objective :- Write a program to implement polyalphabetic cipher

Program :-

```
#include <iostream.h>
#include <string.h>
using namespace std;

int main()
{
    string pt, ct, key, vt;
    int i, j;
    cout << "Enter plain text" << endl;
    cin >> pt;
    cout << "Enter key" << endl;
    cin >> key;

    j = 0;
    for (i = strlen(key); i < strlen(pt); i++)
    {
        if (j == strlen(key))
        {
            j = 0;
            key[i] = key[j];
            j++;
        }
        cout << "new key is :" << key;
    }

    for (i = 0; i < strlen(pt); i++)
    {
        ct[i] = ((pt[i] - 97) + (key[i] - 97)) % 26;
    }

    cout << "cipher text :" << ct << endl;

    for (i = 0; i < strlen(ct); i++)
    {
        if (ct[i] < key[i])
        {
            vt[i] = 26 + ((ct[i] - 97) - (key[i] - 97)) + 97;
        }
        else
            vt[i] = ((ct[i] - 97) - (key[i] - 97)) % 26 + 97;
    }

    cout << "plain text is :" << vt;
}

return 0;
```

OUTPUT

Plain Text : HELLO

Key : MONEY

Cipher Text : TSYPM

Message : HELLO

TUTORIAL / PRACTICAL NO. 3

Objective: Write a program to implement Vernam Cipher.

Program:

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

string Encryption(string text, string key)
{
    string cipherText = "";
    int cipher[Key.length()];
    for (int i = 0; i < Key.length(); i++)
    {
        cipher[i] = text.at(i) - 'A' + key.at(i) - 'A';
    }
    for (int i = 0; i < key.length(); i++)
    {
        if (cipher[i] > 25)
        {
            cipher[i] = cipher[i] - 26;
        }
    }
    for (int i = 0; i < key.length(); i++)
    {
        int x = cipher[i] + 'A';
        cipherText += (char)x;
    }
    cout << cipherText;
}

static string Decryption(string s, string key)
{
    string plaintext = "";
    int plain[Key.length()];
    for (int i = 0; i < key.length(); i++)
    {
        if (plain[i] < 0)
        {
            plain[i] = plain[i] + 26;
        }
    }
    for (int i = 0, i < key.length(); i++)
    
```

TUTORIAL / PRACTICAL NO.

```
{ int x = plaintext[i] + 'A';
  plaintext += (char)x; }

return plaintext;
}

int main()
{
    string plaintext = "Hello";
    string key = "MONEY";
    for (int i = 0; i < plaintext.length(); i++)
    {
        plaintext[i] = toupper(plaintext[i]);
    }
    for (int i = 0; i < key.length(); i++)
    {
        key[i] = toupper(key[i]);
    }
    string encryptedText = Encryption(plaintext, key);
    cout << "Cipher Text" << encryptedText << endl;
    cout << "Plain Text" << Decryption(encryptedText,
key);

return 0;
}
```

TUTORIAL / PRACTICAL NO. 4

❖ Objective: Write a program to implement Rail Fence technique.

❖ Program:

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

string encryptRailFence(string text, int key)
{
    char rail[key][text.length()];
    for (int i = 0; i < key; i++)
        for (int j = 0; j < text.length(); j++)
            rail[i][j] = '\n';
    bool dir-down = false;
    int row = 0, col = 0;
    for (int i = 0; i < text.length(); i++)
    {
        if (row == 0 || row == key - 1)
            dir-down = !dir-down;
        rail[row][col++] = text[i];
        dir-down ? row++ : row--;
    }
    string result;
    for (int i = 0; i < key; i++)
        for (int j = 0; j < text.length(); j++)
            if (rail[i][j] != '\n')
                result.push_back(rail[i][j]);
    return result;
}

string decryptRailFence(string cipher, int key)
{
    char rail[key][cipher.length()];
    for (int i = 0; i < key; i++)
        for (int j = 0; j < cipher.length(); j++)
            rail[i][j] = '\n';
    bool dir-down;
}
```

TUTORIAL / PRACTICAL NO.

```
int row = 0, col = 0;
for( int i=0 ; i < cipher.length() ; i++ )
{
    if( row == 0 )
        dir_down = true ;
    if( row == key - 1 )
        dir_down = false ;
    rail[row][col++] = '*';
    dir_down ? row++ : row-- ;
}

int index = 0;
for( int i=0 ; i < key ; i++ )
    for( int j=0 ; j < cipher.length() ; j++ )
        if( rail[i][j] == '*' && index < cipher.length() )
            rail[i][j] = cipher[index++];
string result;
row = 0, col = 0;
for( int i=0 ; i < cipher.length() ; i++ )
{
    if( row == 0 )
        dir_down = true ;
    if( row == key - 1 )
        dir_down = false ;
    if( rail[row][col] != '*' )
        result.push_back( rail[row][col++]);
    dir_down ? row++ : row-- ;
}
return result;
}

int main()
{
    cout << encryptRailfence("attack at once", 2)
        << endl;
}
```

OUTPUT

Plain Text :

attack at once

defend the east wall

Cipher Text :

atc toctaka ne

dnhanweedtees aff tl

TUTORIAL / PRACTICAL NO.

```
cout << encryptRailFence ("defend the east wall",  
3) << endl;  
  
cout << decryptRailFence ("etc to take me", 2)  
<< endl;  
  
cout << decryptRailFence ("dnhanee dees alf  
ml", 3) << endl;  
  
vector<int> v; //  
{  
    v.push_back(1);  
    v.push_back(2);  
    v.push_back(3);  
    v.push_back(4);  
    v.push_back(5);  
    v.push_back(6);  
    v.push_back(7);  
    v.push_back(8);  
    v.push_back(9);  
    v.push_back(10);  
    v.push_back(11);  
    v.push_back(12);  
    v.push_back(13);  
    v.push_back(14);  
    v.push_back(15);  
    v.push_back(16);  
    v.push_back(17);  
    v.push_back(18);  
    v.push_back(19);  
    v.push_back(20);  
    v.push_back(21);  
    v.push_back(22);  
    v.push_back(23);  
    v.push_back(24);  
    v.push_back(25);  
    v.push_back(26);  
    v.push_back(27);  
    v.push_back(28);  
    v.push_back(29);  
    v.push_back(30);  
    v.push_back(31);  
    v.push_back(32);  
    v.push_back(33);  
    v.push_back(34);  
    v.push_back(35);  
    v.push_back(36);  
    v.push_back(37);  
    v.push_back(38);  
    v.push_back(39);  
    v.push_back(40);  
    v.push_back(41);  
    v.push_back(42);  
    v.push_back(43);  
    v.push_back(44);  
    v.push_back(45);  
    v.push_back(46);  
    v.push_back(47);  
    v.push_back(48);  
    v.push_back(49);  
    v.push_back(50);  
    v.push_back(51);  
    v.push_back(52);  
    v.push_back(53);  
    v.push_back(54);  
    v.push_back(55);  
    v.push_back(56);  
    v.push_back(57);  
    v.push_back(58);  
    v.push_back(59);  
    v.push_back(60);  
    v.push_back(61);  
    v.push_back(62);  
    v.push_back(63);  
    v.push_back(64);  
    v.push_back(65);  
    v.push_back(66);  
    v.push_back(67);  
    v.push_back(68);  
    v.push_back(69);  
    v.push_back(70);  
    v.push_back(71);  
    v.push_back(72);  
    v.push_back(73);  
    v.push_back(74);  
    v.push_back(75);  
    v.push_back(76);  
    v.push_back(77);  
    v.push_back(78);  
    v.push_back(79);  
    v.push_back(80);  
    v.push_back(81);  
    v.push_back(82);  
    v.push_back(83);  
    v.push_back(84);  
    v.push_back(85);  
    v.push_back(86);  
    v.push_back(87);  
    v.push_back(88);  
    v.push_back(89);  
    v.push_back(90);  
    v.push_back(91);  
    v.push_back(92);  
    v.push_back(93);  
    v.push_back(94);  
    v.push_back(95);  
    v.push_back(96);  
    v.push_back(97);  
    v.push_back(98);  
    v.push_back(99);  
    v.push_back(100);  
}
```

TUTORIAL / PRACTICAL NO. 5

❖ Objective: Write program to implement simple columnar transposition technique.

❖ Program : #include <bits/stdc++.h>

using namespace std;

❖ Stealing Encryption (int no_rows, int len_key, int len_mg, string mg, int col_val[])

{ int x=0;

char enc_mat[no_rows+1][len_key];

for (int i=0; i<no_rows+1; i++)

{ for (int j=0; j<len_key; j++)

{ if (x>= len_mg)

{ enc_mat[i][j] = '-';

}

else {

enc_mat[i][j] = mg[x];

}

x++; }

int t=1;

string cipher = " ";

while (t <= len_key)

{ for (int i=0; i<len_key; i++)

{ int R = col_val[i];

if (R == t)

{ for (int j=0; j<no_rows+1; j++)

{ cipher += enc_mat[j][i];

}

t++;

}

}

return cipher;

OUTPUT

Encrypted Message: comwomomeretaho

Decrypted Message: come home tomorrow

TUTORIAL/PRACTICAL NO.

```
❖ string Decryption (int no-rows, int len-key)
❖ { char dec-mat [no-rows+1] [len-key];
  int x=0, t=1;
  while ( t <= len-key )
    { for (int i=0; i<len-key; i++)
        { int R = col-val [i];
          if (R == t)
            { for (int j=0; j<no-rows+1; j++)
                { dec-mat [j][i] = cipher [x];
                  x++;
                } t++; } } }

❖ storing message = "";
❖ for (int i=0; i< rows+1; i++)
  { for (int j=0; j< len-key; j++)
    { if (dec-mat[i][j] == '-')
      { dec-mat [i][j] = ' ';
      }
      message += dec-mat[i][j];
    } } return message;
  }

❖ int main()
{ storing meg = "come home tomorrow";
  storing key = "HAEK";
  int len-key = key.length();
  int len-meg = meg.length();
  int val = 1, count = 0, ind;
  int colval [len-key];
  }
```

TUTORIAL / PRACTICAL NO.

```
while (count < len-key)
{
    int min = 999;
    for (int i=0; i<len-key; i++)
    {
        if ((min > int(key[i])) && col-val[i]==0)
        {
            min = int(key[i]);
            index = i;
        }
    }
    int no-rows = len-msg / len-key;
    string cipher-text = "";
    cipher-text = Encryption (no-rows, lenkey,
    len-msg, msg, col-val);
    cout << "Encrypted message : " << cipher-text
    << endl;
    string original-msg = "";
    original-msg = Decryption (no-rows, len-key,
    cipher-text, col-val);
    cout << "Decrypted message " << original-msg;
}
return 0;
}
```

TUTORIAL/PRACTICAL NO. 6

❖ Objective: Write a program to implement simple RSA algorithm with small numbers.

❖ Program:

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

int gcd (int a, int b)
{
    int temp;
    while (1)
    {
        temp = a % b;
        if (temp == 0)
            return b;
        a = b;
        b = temp;
    }
}

int main()
{
    double p = 3;
    double q = 7;
    double n = p * q;
    double e = 2;
    double phi = (p - 1) * (q - 1);
    while (e < phi)
    {
        if (gcd(phi) == 1)
            break;
        else
            e++;
    }

    int k = 2;
    double d = (1 + (k * phi)) / e;
    double msg = 12;
    cout << "Message data" << msg;
```

OUTPUT

Message data = 12.000000

Encrypted data = 3.000000

Original Message Sent = 12.000000

TUTORIAL / PRACTICAL NO.

```
double c = pow(mrg, e);
c = fmod(c, n);
cout << "Encrypted data :" << c;
double m = pow(c, d);
m = fmod(m, n);
cout << "Original message sent" << m;
between 0;
}
```

OUTPUT

$$\text{GCD}(10, 15) = 5$$

$$\text{GCD}(35, 10) = 5$$

$$\text{GCD}(31, 2) = 1$$

UTORIAL / PRACTICAL NO. 7

Objective : Write a program to implement
Euclidean algorithm.

Program :

```
#include <stdio.h>
#include <bits/stdc++.h>
using namespace std;
int gcd (int a, int b)
{
    if (a == 0)
        return b;
    return gcd(b % a, a);
}

int main ()
{
    int a = 10, b = 15;
    cout << "GCD (" << a << ", " << b << ") = " <<
    gcd (a, b) << endl;
    a = 35, b = 40;
    cout << "GCD (" << a << ", " << b << ") = " <<
    gcd (a, b) << endl;
    a = 31, b = 2;
    cout << "GCD (" << a << ", " << b << ") = " <<
    gcd (a, b) << endl;
    return 0;
}
```

TUTORIAL / PRACTICAL NO. 8

❖ Objective: Write various steps to ensure security of any web browser (Firefox , Google chrome)

Firefox ⇒

STEPS

- ❖ • Auto - install updates → For both Mac and PC - go to Firefox menu > Preferences (Mac) options (PC) General tab > Firefox Update section . Select " Automatically install updates ".
- ❖ • Block unwanted pop-ups → For both Mac and PC - go to Firefox menu > Preferences (Mac) options (PC) > Privacy + Security > Permission section . Check " Warn you when website try to install add-ons ";
- ❖ • Don't save passwords → For both Mac & PC - go to Firefox menu > Browser Privacy section . Uncheck the " Ask to save login & passwords for websites " box .
- ❖ • Cookies + Site Data → Select " custom " and set cookies to block " Third - party trackers ". Also place checks to block " Cryptominers + Fingerprinters "

Google Chrome ⇒

STEPS

- ❖ • Auto - download updates → To make sure that you're protected by latest security updates , google chrome automatically updates whenever it detects new version is available

TUTORIAL/PRACTICAL NO.

- ❖ Block unwanted plugins → Go to settings > Unsandboxed plugins access & turn on "Ask when a site wants to use a plugin to access your computer".
- ❖ Don't save passwords → Go to settings > Passwords & turn off "Offer to save passwords".
- ❖ Handle cookies → Turn on "Allow site to save and read cookie data (recommended)" and "Block - Third party cookies".
- ❖ Camera Access → Go to Settings > Advanced > Content settings > Camera & turn on "Ask before accessing (recommended)".