OPEN ENDED PROBLEM

PROBLEM STATEMENT: Implement two cylinder rotor machine in C/C++. First cylinder is faster than second one. Internal mapping of cylinder must be random. It encrypts as many as strings until user opt to exit. Also it prints states of cylinder before and after encrypting a string, and ciphertext in between this two states of cylinder.

INTRODUCTION:

Electric rotor machines were mechanical devices that allowed to use encryption algorithms that were much more complex than ciphers, which were used manually. They were developed in the middle of the second decade of the 20th century. They became one of the most important cryptographic solutions in the world for the next tens of years.

Usage

Electro-mechanical machines fitted with movable rotors are used to produce long random keystreams, thus allowing to encrypt messages by using complicated polyalphabetic substitution ciphers.

Description

The main idea that lies behind rotor machines is relatively simple. One can imagine a simple device, similar to a typewriter, with a number of keys used to input text. The number of keys may differ, however usually there are 26 to 32 characters.

Simple substitution cipher

Each keystroke produces an output character, depending of the internal construction of the machine. In the simplest case, if only wires are used (without any rotors), each input key will be mapped to one specific output character.

For example, if someone pressed K in the keyboard, the machine would always produce C.
As a result, the machine would encrypt the messages by using a simple substitution cipher.

Adding a rotor

Having a simple substitution machine, one can imagine adding an additional internal rotor with an internal wiring. The rotor will rotate with a gear, each time after a keystroke. As a result, after pressing the same letter twice, it will be encoded differently due to different internal wiring.

For example, if someone pressed KK in the keyboard, the machine would produce CB (because the wiring changed after the first keystroke, due to the rotor movement).
The internal wiring of the rotor should be kept secret, however we may expect that over time the enemy will discover its design. It will make it easier for them to break the cipher but it won't compromise the security altogether.

To decode a ciphertext the receiver would need a machine with the same rotor. Adding the rotor caused the encryption to become a stronger polyalphabetic substitution cipher.

Make it difficult

To improve the security, one could add more rotors. The output of one rotor would be connected to the input of the second rotor. Similarly, the second rotor output would be connected to the third one, and so on. The strength of the encryption depends on several factors:

- the number of rotors inside the machine.
- the size of each rotor.
- the number of rotor types (with different internal wirings).

Each rotor would contain a different internal wiring. The substitution performed by each rotor should be unknown for the enemy. To make cryptanalysis more difficult and to ensure that the wiring inside each rotor changes with different frequency, the discs should rotate with different speeds.

Additionally, depending on the design of the machine, some additional features may be added to the machine, to ensure that the produced substitution is as random as possible (for example, an additional fixed substitution that does not depends on the rotors).

Some rotor machines (most notably Enigma) were designed to be symmetrical. That means that encrypting the same message twice (with the same settings), would produce the original message.

CRYPTOGRAPHIC ROTOR MACHINE

Hebern Rotor Machine

Hebern rotor machines were one of the first cryptographic machines, designed for encrypting messages automatically. It was first produced in 1917 in the USA.

Lorenz Rotor Machine

Lorenz cryptographic rotor machines were used by the Germans during World War II for encrypting strategic communication between major cities in occupied Europe.

Enigma

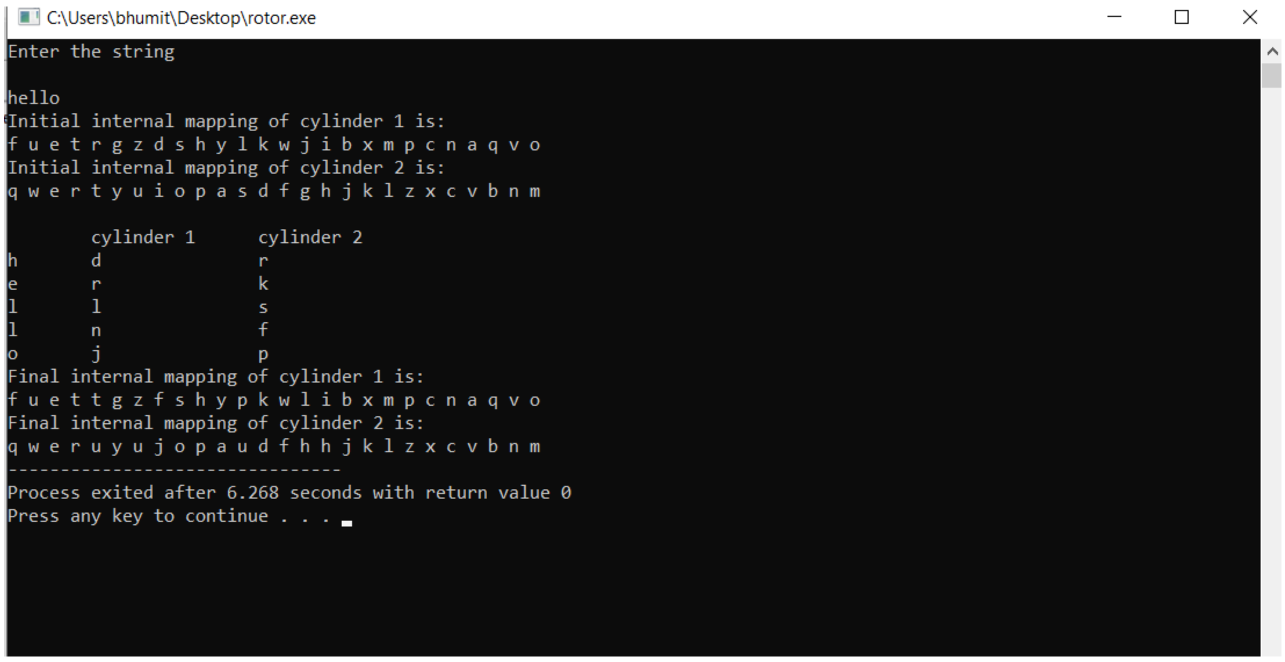
Enigma was one of the most famous and most popular cryptographic rotor machines, used mainly by the Germans during World War II.

SOURCE CODE:

```
#include<conio.h>
#include<stdio.h>
char  rotor1[26]={'f','u','e','t','r','g','z','d','s','h','y','l','k','w','j','i','b','x','m','p','c','n','a','q','v','o'};
char  rotor2[26]={'q','w','e','r','t','y','u','i','o','p','a','s','d','f','g','h','j','k','l','z','x','c','v','b','n','m'};

int  main()
{
char input;
int i,j;
printf("Enter the string \n\n"); scanf("%c",&input);
printf("Initial internal mapping of cylinder 1 is:\n");
for(i=0;i<26;i++)
{
printf("%c ",rotor1[i]);
}
printf("\nInitial internal mapping of cylinder 2 is:\n");
```

```
for(i=0;i<26;i++)
{
printf("%c ",rotor2[i]);
}

printf("\n\n\tcylinder 1\tcylinder 2\n");
while(input!=' ')
{
i=input-97; printf("%c",input); printf("\t%c\t\t",rotor1[i]); j=rotor1[i]-97; printf("%c\n",rotor2[j]);

{
rotor1[i]=((rotor1[i]-97+2)%26)+'a';
rotor2[i]=((rotor2[i]-97+1)%26)+'a';
}

scanf("%c",&input);
}

printf("Final internal mapping of cylinder 1 is:\n");
for(i=0;i<26;i++)
{
printf("%c ",rotor1[i]);
}
printf("\nFinal internal mapping of cylinder 2 is:\n");
for(i=0;i<26;i++)
{
printf("%c ",rotor2[i]);
}
return 0;
```

OUTPUT :



C:\Users\bhumit\Desktop\rotor.exe

```
Enter the string

hello
Initial internal mapping of cylinder 1 is:
f u e t r g z d s h y l k w j i b x m p c n a q v o
Initial internal mapping of cylinder 2 is:
q w e r t y u i o p a s d f g h j k l z x c v b n m

        cylinder 1      cylinder 2
h       d               r
e       r               k
l       l               s
l       n               f
o       j               p
Final internal mapping of cylinder 1 is:
f u e t t g z f s h y p k w l i b x m p c n a q v o
Final internal mapping of cylinder 2 is:
q w e r u y u j o p a u d f h h j k l z x c v b n m
--------------------------------
Process exited after 6.268 seconds with return value 0
Press any key to continue . . .
```

REFERENCE:

- http://www.crypto-it.net/eng/simple/rotor-machines.html
- William Stalllings