Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Отчёт**

по шестой лабораторной работе

Студент: Пицуха Я. А.

ФИТ 3 курс 4 группа

Минск 2020

1. Ознакомиться с функционалом хотя бы одного (по согласованию с преподавателем) симулятора Энигмы.

Попробовал различные симуляторы и изучил принцип действия машин.

2. Разработать приложение-симулятор шифровальной машины, состоящей из клавиатуры, трех роторов и отражателя. Типы роторов (L-M-R) и отражателя Re следует выбрать из таблиц на рис. 4.5 и 4.6 в соответствии со своим вариантом, представленным в таблице 4.1. Крайний правый столбец этой таблицы показывает, на какое число шагов (букв, i) перемещается соответствующий ротор при зашифровании одного (текущего) символа; число 0 означает перемещение соответствующего ротора на один шаг при условии, что расположенный правее ротор совершит один оборот.

Вариант 11; L-M-R: Beta VIII I; Re: B; Li-Mi-Ri: Dunn 3-1;



5\*4\*3=60 (количество возможных комбинаций роторов)

26\*26\*26=17576 (возможные положения роторов)

26\*26=676 (начальные положения роторов)

26! / (26-2\*10) \*10! \* 210=1,5 \*1014 (коммутация панели)

60\*17576\*676\*1,5\*1014 =1,07\*1023

Вывод: в ходе лабораторной работы мы разработали шифровальную машину Энигма. Также установили, что криптостойкость машины составляет 1,07\*1023 комбинаций.

Листинги программы:

rotor = [  
 ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i',  
 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r',  
 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'],  
 ['l', 'e', 'y', 'j', 'v', 'c', 'n', 'i', 'x',  
 'w', 'p', 'b', 'q', 'm', 'd', 'r', 't', 'a',  
 'k', 'z', 'g', 'f', 'u', 'h', 'o', 's'],  
 ['f', 'k', 'q', 'h', 't', 'l', 'x', 'o', 'c',  
 'b', 'j', 's', 'p', 'd', 'z', 'r', 'a', 'm',  
 'e', 'w', 'n', 'i', 'u', 'y', 'g', 'v'],  
 ['e', 'k', 'm', 'f', 'l', 'g', 'd', 'q', 'v',  
 'z', 'n', 't', 'o', 'w', 'y', 'h', 'x', 'u',  
 's', 'p', 'a', 'i', 'b', 'r', 'c', 'j']  
]  
  
reflector = {  
 'a': 'e', 'b': 'n', 'c': 'k', 'd': 'q',  
 'f': 'u', 'g': 'y', 'h': 'w', 'i': 'j',  
 'l': 'o', 'm': 'p', 'r': 'x', 's': 'z', 't': 'v'  
}  
  
rotorShifts = [3, 1, 3]  
# [11] Beta VIII I B Dunn 3-1-3  
  
  
def shiftRotor(n):  
 temp = []  
 shCount = rotorShifts[n-1]  
 for i in range(0, len(rotor[n])):  
 if i - shCount < 0:  
 temp.append(rotor[n][len(rotor[n]) + (i-shCount)])  
 else:  
 temp.append(rotor[n][i-shCount])  
 for i in range(0, len(rotor[n])):  
 rotor[n][i] = temp[i]  
  
  
def Enigma(m):  
 m = m.lower()  
 encrypted\_m = []  
  
 for e in m:  
 encrypted\_m.append(e)  
  
 for k in range(0, len(m)):  
 for n in range(len(rotor) - 1, 0, -1):  
 encrypted\_m[k] = rotor[n][rotor[0].index(encrypted\_m[k])]  
  
 for e in reflector:  
 if encrypted\_m[k] == reflector[e]:  
 encrypted\_m[k] = e  
 elif encrypted\_m[k] == e:  
 encrypted\_m[k] = reflector[e]  
  
 for n in range(1, len(rotor)):  
 encrypted\_m[k] = rotor[n][rotor[0].index(encrypted\_m[k])]  
  
 for i in range(1, len(rotor)):  
 shiftRotor(i)  
 print("".join(encrypted\_m))  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 message = str(input())  
 Enigma(message)