CSN-261 L2 REPORT

Anupam Kumar Singh
18114009
O1

Dated: 6/8/2019

PROBLEM STATEMENT 1:
Objective:
1)Create 3 different c program files in which one encrypts, decrypts and compares according to appropriate input keys(n,a,b).
Algos Discuss:
1)TRANSPOSE.c->Takes appropriate n, a, b and divides the input file in section of n characters each and encrypts them individually according to the formula
encrypt[j]=input[(a*i+b)mod n].
2)INVERSE TRANSPOSE.c->Takes appropriate n, a, b and divides the output file file in section of n characters each and decrypts them individually according to the formula
encrypt[j]=decrypt[(a*i+b)mod n].
3)COMPARE.c->Just compares input and decrypted output character by charachter and tells if there is any error in decryption.

```
as1ngh@TELLOWART:~/CSN261/L2/Q1$ time ./transpose
ENTER n:5
ENTER a:3
ENTER b:2
lHleo
W,o r
!l-d-

real    0m1.761s
user    0m0.003s
sys     0m0.000s
```

```
as1ngh@TELLOWART:~/CSN261/L2/Q1$ time ./inverseTranspose
ENTER n:5
ENTER a:3
ENTER b:2
Hello
, Wor
ld!


real    0m1.607s
user    0m0.001s
sys     0m0.003s
```

```
as1ngh@TELLOWART:~/CSN261/L2/Q1$ time ./compare
H H
e e
l l
l l
o o
, ,

W W
o o
r r
l l
d d
! !
OK!
real    0m0.003s
user    0m0.001s
sys     0m0.002s
```

PROBLEM STATEMENT 2:
Objective:
1)Create a quatre and convert the input 2D array into maximal array and print the quatree in form of 3 integer in which first is node value, second is bit value, third is its level in quatre.

Algos Discuss:
1)First we have to add appropriate padding into input array so that it forms a 2D array of some 2^n type.
2)The above task can be performed very easy by writing dfs code with recursion and assign the value to the node when that node contains all value 0 or 1(any one of these).

```
as1ngh@TELLOWART:~/CSN261/L2/Q2$ time ./Q2
(1, 0, 2)
(2, 0, 3)
(3, 0, 3)
(4, 1, 3)
(5, 1, 3)
(6, 0, 3)
(7, 0, 4)
(8, 1, 4)
(9, 1, 4)
(10, 1, 4)
(11, 0, 3)
(12, 1, 3)
(13, 1, 3)
(14, 1, 3)
(15, 1, 4)
(16, 1, 4)
(17, 1, 4)
(18, 0, 4)
(19, 0, 3)
1       1       1       1       2       2       3       3
1       1       1       1       2       2       3       3
1       1       1       1       4       4       5       5
1       1       1       1       4       4       5       5
6       6       7       8       13      13      14      14
6       6       9       10      13      13      14      14
11      11      12      12      15      16      19      19
11      11      12      12      17      18      19      19

real    0m0.003s
user    0m0.003s
sys     0m0.000s
```