Q1 Team Name
0 Points

INFINITY

Q2 Commands
10 Points

List the commands used in the game to reach the ciphertext.

To get the morse code from starting screen - 'go'
To get back to starting screen from morse code - 'back'
To reach the ciphertext from starting screen- 'read'

Q3 CryptoSystem
10 Points

What cryptosystem was used in this level?

Play Fair Encryption Algorithm

Q4 Analysis
20 Points

What tools and observations were used to figure out the cryptosystem? (Explain in less than 300 words)

Tools - C++
1) We first used the 'go' command, then we got a paragraph containing the sequence of '.' and '-'. From prior knowledge, we figured out it is a morse code.
2) After translating the morse code on the boulder, we got the word 'CRYPTANALYSIS'.
3) The whole paragraph was written in small English alphabets except in the last line word 'PLAY FAIR' was written in capital letters. After googling the word, it turned out to be an encryption algorithm.
4) We then used the 'back' command and to read the 'patterns written on the boulder', we used the 'read' command. This way we found the ciphertext.
5) Upon reading about the Play Fair algorithm, we learned about how encryption and decryption are done and the key required for the same.
6) This led us to the conclusion that the word 'CRYPTANALYSIS' is used as the key in the algorithm.
References :
(a) https://www.geeksforgeeks.org/playfair-cipher-with-examples/
(b) https://en.wikipedia.org/wiki/Playfair_cipher

Q5 Decryption Algorithm
15 Points

Briefly describe the decryption algorithm used. Also mention the plaintext you deciphered. (Use less than 350 words)

PlayFair Cipher
1) PlayFair is a digraph substitution cipher.
2) First 5*5 key matrix is created containing 25 unique English alphabets and one alphabet (usually 'J') is omitted from the table (as the matrix can hold only 25 alphabets). If the plaintext contains 'J', then it is replaced by 'I'.
3) The initial alphabets in the key matrix are the unique alphabets of the key ('CRYPTANALYSIS') in the order in which they appear, followed by the remaining letters of the English alphabet in order, filled from left to right row-wise in the matrix.
4) Final 5*5 Matrix would look like this

C R Y P T
A N L S I
B D E F G
H K M O Q
U V W X Z

5) Before pairing the letters in the ciphertext, we removed all the spaces and punctuation marks.
6) For decryption, we consider pairs of two letters then we apply one of the below rules -
a) If both letters of the pair are in the same row, then take the letter to the left of each one (for the first letter we take the last letter of the row).
b) If both letters of the pair are in the same column, then take the letter to the above of each one (for the first letter we take the last letter of the column).
c) If both the letters are from the different row and column then form a rectangle with both letters as diagonally opposite corners and we take the letter on the horizontally opposite corner of the rectangle as plain text for that letter.
7) While encrypting, pair cannot be made with the same letter. So, we have to break the letter in single and add an 'X' to the previous letter.
e.g. - hello - 'he' 'lx' 'lo'.
So, while decrypting, we have to remove extra 'X' from the plain text whenever two same letters are separated by 'X'.
8) We then inserted the required spaces and punctuation marks at their original places.
9) We used C++ code and got the final decrypted text(plaintext) as follows -
BE WARY OF THE NEXT CHAMBER, THERE IS VERY LITTLE JOY THERE. SPEAK OUT THE PASSWORD "ABRA_CA_DABRA" TO GO THROUGH. MAY YOU HAVE THE STRENGTH FOR THE NEXT CHAMBER. TO FIND THE EXIT YOU FIRST WILL NEED TO UTTER MAGIC WORDS THERE.

Q6 Password
10 Points

What was the final command used to clear this level?

ABRA_CA_DABRA

Q7 Code
0 Points

Upload any code that you have used to solve this level

CS641_A2_INF.cpp — Download

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

int main()
{
    string s = "";
    while (1)
    {
        char c;
        cin >> c;
        if (c == ';')
            break;
        if (c == ',' || c == '.' || c == '"' || c == '_')
            continue;
        s += c;
    }
    char a[5][5] = {{'C', 'R', 'Y', 'P', 'T'}, {'A', 'N', 'L', 'S', 'I'}, {'B', 'D', 'E', 'F', 'G'}, {'H', 'K', 'M', 'O', 'Q'}, {'U', 'V', 'W', 'X', 'Z'}};
    map<char, pair<int, int>> m;
    for (int i = 0; i < 5; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            m[a[i][j]] = {i, j};
        }
    }
    string ans;
    int n = s.size();
    for (int i = 0; i < n; i += 2)
    {
        char c1 = s[i];
        char c2 = s[i + 1];
        pair<int, int> p1 = m[c1];
        pair<int, int> p2 = m[c2];
        if (p1.second == p2.second)
        {
            ans += a[(p1.first + 4) % 5][p1.second];
            ans += a[(p2.first + 4) % 5][p2.second];
        }
        else if (p1.first == p2.first)
        {
            ans += a[p1.first][(p1.second + 4) % 5];
            ans += a[p2.first][(p2.second + 4) % 5];
        }
        else
        {
            ans += a[p1.first][p2.second];
            ans += a[p2.first][p1.second];
        }
    }
    cout << s << endl;
    cout << ans << endl;
}
```