Q1 Commands
5 Points

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

climb, read, enter, read

Q2 Cryptosystem
5 Points

What cryptosystem was used in this level?

Substitution cipher (with altered whitespaces)

Q3 Analysis
25 Points

What tools and observations were used to figure our the cryptosystem? (Explain in less than 100 words)

Tools Used: Wrote a CPP Code.

We were uncertain whether the cryptosystem used was a substitution cipher or any other cryptosystem based on seeing the cipher text, therefore we first conducted a frequency analysis to check whether it was a substitution cipher.
We wrote a program to find all the monogram, bigram and trigram frequencies in the given ciphertext. Seeing the percentage occurrence of monograms and bigrams, the percentage distribution was strongly matching with actual English dialects. Thus we conclude the cryptosystem used is a substitution cipher. There is an exclamation mark(!) between one of the words of ciphertext. This means either of the two things:- 1) an alphabet has been substituted with an exclamation mark, or 2) the whitespaces between the actual words have been altered during conversion to ciphertext. Also, there is no 'full stop' punctuation mark at the end of ciphertext. This shows that the last two words in ciphertext are actually not the last words of plaintext(their position has been changed). Before doing frequency analysis, we have removed punctuation marks,digits and whitespaces because to find the frequency of each letter, these symbols should not be included in the length of the string. Also, whitespace removal is mandatory here in counting bigrams and trigrams because whitespaces are not at their correct places(If we don't remove whitespace then some bigrams and trigrams will be lost). Note that because of removing whitespaces some other pairs will also be counted as bigrams which shouldn't be but the count of such bigrams will be less and we are going to substitute the top few most frequent n-grams only.

In the given ciphertext, the most frequent occurring monogram, bigram and trigrams are c, fi and fic respectively. Then we used the fact that in english language, the most occuring monogram, bigram and trigram are e, th, the respecitively. On substituting monogram 'c' with 'e', 'f' with 't' , 'i' with 'h' , the bigram and trigram frequencies are satisfied in cipher and english language. Hence, it is highly likely that 'f' is 't' , 'i' is 'h' and 'c' is 'e'.

We got the following bigram frequencies in decreasing order : fi, ic, cm, ck. We already know what 'f' , 'i' and 'c' are. We need to find what should 'cm' and 'ck' translate to. In english, we know the most frequent bigrams starting with 'e' are 'en' , 'er' and 'es'. So we assume the possibilitites of 'm' to be {'n','r','s'} and we start by substituting 'm' with 'n'. We don't find any word emerging right now but we observed that after replacing 'm' with 'n' we get words like 'thene' and 'tenest'. It is suggesting that 'm' should be replaced with 'r' rather than 'n' and in doing so we will get words like 'there' and 'terest'. Moreover, two alphabets before 'terest' should be 'in' so that the word becomes 'interest'. This suggests the following replacements:- m-->r , o-->i , q-->n.

Now, we get a word interestinA. This suggests 'A' should be 'g' (capital letter denotes the word is not yet replaced). So, a-->g.

Now, we get a word nGthing. This suggests 'G' should be 'o'.

Then, we get a word interestingthHnthisone. This suggests 'H' should be 'a'.

Then, the word thereisnothingoEinterestinthe suggests 'E' should be 'f'.

Then, the word sVNstitVtionPiJher should be substitutioncipher. So, 'V'-->'u' , 'N'-->'b', 'P'-->'c', 'J'-->'p'.

Then, the word substitutioncipherinLhichYigitshaSebeenshifteYbR2pUaces should be substitutioncipherinwhichdigitshavebeenshiftedby2places. So, 'L'-->'w', 'Y'-->'d', 'S'-->'v', 'R'-->'y', 'U'-->'l'
Now, all words are clearly visible and the remaining replacements are 'D'-->'m' and 'X'-->'q'.

The password after shifting digits by 2 places in 'iRqy9U1qdgt' is 'iRqy1U5qdgt' but it is not

Assignment 1

GRADED

GROUP
Sharanya Saha
Ayush Sahni
Manjyot Singh Nanra
View or edit group

TOTAL POINTS
50 / 50 pts

QUESTION 1	Commands	R **5** / 5 pts
QUESTION 2	Cryptosystem	**5** / 5 pts
QUESTION 3	Analysis	R **25** / 25 pts
QUESTION 4	Mapping	**10** / 10 pts
QUESTION 5	Password	**5** / 5 pts
QUESTION 6	Codes	**0** / 0 pts
QUESTION 7	Team Name	**0** / 0 pts

getting accepted. So this must mean that '2' itself is encrypted and we need to shift it by 2 places. But it isn't mentioned whether digits have been shifted forward or backwards. Suppose originally digits have been shifted forward by x places. If we consider encrypted message as "digits are shifted forwards by 2 places" it would give us equation (x+x) mod 10 = 2. Possible values of x are x=1,6. But neither of the values gave the correct password. This means that the case where "digits are shifted backwards by 2 places" must be the solution. Analysing this case, we know that shifting digits backwards by 2 places and shifting digits forward by 8 places means the same thing. Thus our equation will be (x+x) mod 10 = 8. Possible values are x=4,9. Out of these using value x=4, the password got accepted. Thus the correct decryption is that "Digits have been shifted forward by 4 places".
The final password is 'iRqy3U5qdgt'.

Q4 Mapping
10 Points

What is the plaintext space and ciphertext space?
What is the mapping between the elements of plaintext space and the elements of ciphertext space? (Explain in less than 100 words)

plaintext space: {[A-Z],[a-z],[0-9],'!',',',.'}
ciphertext space: {[A-Z],[a-z],[0-9],'!',',',.'}
Plaintext space and ciphertext space includes all alphabets, digits and punctuation marks('!',',',.'). Digits are not substituted with alphabets and alphabets are not substituted with digits. Digits are only mapped to digits. Uppercase alphabets are only mapped to uppercase alphabets and lowercase alphabets are only mapped to lowercase alphabets.

Note that plaintext and ciphertext space also contains punctuation marks but they are not substituted with other symbols and are mapped to themselves.
MAPPING:
a->g, c->e, d->m, e->f, f->t, g->o, h->a, i->h, j->p, k->s, l->w, m->r, n->b, o->i, p->c, q->n, r->y, s->v, u->l, v->u, x->q, y->d, 9->3, 1->5
Some digits and alphabets are not mentioned because they are not present in the plaintext.

Q5 Password
5 Points

What is the final command used to clear this level?

iRqy3U5qdgt

Q6 Codes
0 Points

Upload any code that you have used to solve this level

escape_cave.cpp Download

```cpp
#include<bits/stdc++.h>
#include <iostream>
using namespace std;
bool sortbyfreq(const pair<char,float> &a,
              const pair<char,float> &b)
{
    return (a.second > b.second);
}
bool mysort(pair<string, float>& a,
         pair<string, float>& b)
{
    return a.second > b.second;
}
int main(){
    string
ciphertext="omkfpihdncmgeficphsckHkrgvphqkccficmcokqgfioqageoqfcmckfoqficpihdncmKgdcgef
    transform(ciphertext.begin(), ciphertext.end(), ciphertext.begin(), ::tolower);

    //MONOGRAMS
    cout<<"MONOGRAMS"<<endl;
    vector<pair<char,float>> frequency;
    pair<char,float> p;
    p.first='a';
    p.second=0;
    frequency.push_back(p);
    for(int i=1;i<26;i++){
        char ch='a'+i;
        pair<char,float> pt;
        pt.first=ch;
        pt.second=0;
        frequency.push_back(pt);
    }


    for(int i=0;i<ciphertext.length();i++){
        frequency[ciphertext[i]-97].second+=1;;
    }
    for(int i=0;i<26;i++){
        frequency[i].second=frequency[i].second*100/ciphertext.length();
    }
```

```
    }
     sort(frequency.begin(), frequency.end(), sortbyfreq);

    //display monograms with their frequency in decreasing order
    for(int i=0;i<26;i++){
        cout<<frequency[i].first<<" "<<frequency[i].second<<endl;
    }
    cout<<endl;
    //BIGRAMS
    cout<<"BIGRAMS"<<endl;
    map<string,float> bigram_freq;

    for(int i=0;i<26;i++){
        char s1='a'+i;

        for(int j=0;j<26;j++){
            char s2='a'+j;
            string s3="";
            s3+=s1;
            s3+=s2;
            bigram_freq[s3]=0;
        }

}
  for(int i=0;i<ciphertext.length()-1;i++){
        {
            string temp2="";
            temp2+=ciphertext[i];
            temp2+=ciphertext[i+1];

            bigram_freq[temp2]++;
        }
    }
    for(int i=0;i<26;i++){
        char s1='a'+i;

        for(int j=0;j<26;j++){
            char s2='a'+j;
            string s3="";
            s3+=s1;
            s3+=s2;
            bigram_freq[s3]=(bigram_freq[s3]*100)/(bigram_freq.size() -1);
        }

    }

    vector<pair<string, float>> vec;

        // copy key-value pairs from the map to the vector
  map<string, float> :: iterator it2;
  for (it2=bigram_freq.begin(); it2!=bigram_freq.end(); it2++)
  {
    vec.push_back(make_pair(it2->first, it2->second));
  }

        // display bigram frequency in descending order
  sort(vec.begin(), vec.end(), mysort);
  for(int i=0;i<vec.size();i++)
  cout<<vec[i].first<<" "<<vec[i].second<<endl;

 cout<<endl;
    //TRIGRAMS
    cout<<"TRIGRAMS"<<endl;
    map<string,float> trigram_freq;

    for(int i=0;i<26;i++){
        char s1='a'+i;

        for(int j=0;j<26;j++){
            char s2='a'+j;
            for(int k=0;k<26;k++){
                char s3='a'+k;
                string s4="";
                s4+=s1;
                s4+=s2;
                s4+=s3;
                trigram_freq[s4]=0;
            }
     }

    }

    for(int i=0;i<ciphertext.length()-2;i++){
        {
            string temp2="";
            temp2+=ciphertext[i];
            temp2+=ciphertext[i+1];
            temp2+=ciphertext[i+2];

            trigram_freq[temp2]++;
        }
    }

    for(int i=0;i<26;i++){
        char s1='a'+i;

 for(int j=0;j<26;j++){
            char s2='a'+j;
            for(int k=0;k<26;k++){
                char s3='a'+k;
                string s4="";
                s4+=s1;
```

```
                s4+=s2;
                s4+=s3;
                trigram_freq[s4]=(trigram_freq[s4]*100)/(trigram_freq.size()-2);
            }
        }

    }
    vector<pair<string, float>> vec2;

 for (it2=trigram_freq.begin(); it2!=trigram_freq.end(); it2++)
  {
    vec2.push_back(make_pair(it2->first, it2->second));
  }

        // display trigram frequency in descending order
  sort(vec2.begin(), vec2.end(), mysort);
  for(int i=0;i<vec.size();i++)
  cout<<vec2[i].first<<" "<<vec2[i].second<<endl;


  string cipher_text =
"omkfpihdncmgeficphsck.Hkrgvphqkcc,ficmcokqgfioqageoqfcmckfoqficpihdncm.Kgdcgeficuhfcmp

        transform(cipher_text.begin(), cipher_text.end(), cipher_text.begin(),
::tolower);
        //cout<<cipher_text;

        map<char, char> replace;

    // insert elements in random order
    replace.insert(pair<char, char>('a', 'g'));
    replace.insert(pair<char, char>('c', 'e'));
    replace.insert(pair<char, char>('d', 'm'));
    replace.insert(pair<char, char>('e', 'f'));
    replace.insert(pair<char, char>('f', 't'));
    replace.insert(pair<char, char>('g', 'o'));
    replace.insert(pair<char, char>('h', 'a'));
    replace.insert(pair<char, char>('i', 'h'));
    replace.insert(pair<char, char>('j', 'p'));
    replace.insert(pair<char, char>('k', 's'));
    replace.insert(pair<char, char>('l', 'w'));
    replace.insert(pair<char, char>('m', 'r'));
    replace.insert(pair<char, char>('n', 'b'));
    replace.insert(pair<char, char>('o', 'i'));
    replace.insert(pair<char, char>('p', 'c'));
    replace.insert(pair<char, char>('q', 'n'));
    replace.insert(pair<char, char>('s', 'v'));
    replace.insert(pair<char, char>('u', 'l'));
    replace.insert(pair<char, char>('x', 'q'));
    replace.insert(pair<char, char>('y', 'd'));
    replace.insert(pair<char, char>('r', 'y'));

    for(int i=0;i<cipher_text.length();i++)
    {
        if (replace.find(cipher_text[i]) != replace.end())
            cipher_text[i]=replace[cipher_text[i]];
        }
        cout<<cipher_text;
        return 0;
}
```

Q7 Team Name

0 Points

Pasta_Sandwich