Cpts 223

Written Homework Assignment 2

Yurun Han

1.

a)

Runtime complexity for f() and g()

In g(), function call n time till n=0

so here complexity is O(n)

In f(), For loop call n times so again here complexity is O(n)

b)

```
int h(int n)

{

(n==0)? Return 0: return 1+h(n-1);

}
```

2.

Here first analyze the complexity of f():
  f() calling itself recursively with parameter n/2 each time
  Means it is dividing n by 2 in each iteration

  So, time complexity: O(logn)

No, for g():
  In g(), we have a for loop.
  We are incrementing i by doubling its current time,
  So, for loop runs logn time

  Since in each iteration f() is getting called

  So, time complexity of g() = logn * logn = (logn)^2

3.

Algorithm to find k:

1. Read the value of n
2. Assign a variable k=0
3. Create a Boolean array B of size 10 such that initially all the values of the array contains false
4. Repeat a loop until all the values in the B are true
5. Increment k by 1
6. Multiply n with k and store result in R
7. Now take each digit in R and update value in B corresponding to digit as true
8. End loop
9. Return k

```
#include <iostream>

using namespace std;

int main()

{

    int n,k;

    //Boolean array of size 10

    bool digitExist[10];

    //Initially, fill the boolean array with false

    for (int i = 0; i < 10; i++)

    {

        digitExist[i] = false;

    }

    //Read the value of n

    cout << "Enter n: ";

    cin >> n;

    //Assign the variable k with 0

    k = 0;

    //Loop repets untill all digits 0-9 are found. i.e all boolean array

    //values are true

    while (digitExist[0] == false || digitExist[1] == false || digitExist[2] == false ||

        digitExist[3] == false || digitExist[4] == false || digitExist[5] == false ||
        digitExist[6] == false || digitExist[7] == false || digitExist[8] == false ||
        digitExist[9] == false)
    {
        //increment k. (k=1,2,3,...)
        k++;
        //Multiply n and k. store the result in R
        int R = n*k;
        //Check each digit in R
        while (R != 0)
        {
            int j = R % 10;//Take each digit in R
            //update value in B corresponding to digit as true.
            //That is , if a digit is found, then update its corresponding value in
            //boolean array
            digitExist[j] = true;
            R = R / 10;
        }
    }//end loop
    cout << "k= " << k << endl;
        return 0;
}
```

Time complexity:

The outer while loop runs k times.

Let the length of the R is n or number of digits in R is n. Then the inner while loop runs n times.

Therefore obviously the total running time will be $O(nk)$

4.

a.

whether the number is even or odd.

Pseudo code:-

```
IF num % 2 == 0:
      "EVEN"
ELSE :
       "ODD"
```

Complexity O(1).

b.

Let the list be A and number be n.

```
for i in A:
if i == n:
    "FOUND"
"NOT FOUND"
```

Complexity:- O(n).

c.

Let us consider that the smallest number initially is first element of list.

So,

```
for i in range(1,n):
if a[i] < min:
     min = a[i]
```

min is the smallest number.

Complexity:- O(n)

d.

```
for i in range(0,n):
for j in range(0,n):
    if a[i] != b[j]:
      "NO"
```

"YES"

complexity:- O(n^2)

e.

As the lists are sorted and of same length , then all elements at same index will be equal.

So, pseudo code:-

```
for i in range(0,n):

if a[i] != b[i]:

    "NOT EQUAL"

"EQUAL"
```

Complexity:- O(n)

f.

Complexity:- O(n)

5.

```
#include <iostream>
using namespace std;
bool isAnagram(string str1, string str2)
{
   // if 2 strings are different length return false
   if(str1.length()!=str2.length())
       return false;
// create 2 arrays to store the count of 2 strings
int count1[256] = { 0 };
int count2[256] = { 0 };
int i;

// increase the count of chars by iterating char by char
for (i = 0; i<str1.length(); i++) {
count1[str1[i]]++;
count2[str2[i]]++;
}



// Comparing count arrays
for (i = 0; i < 256; i++)
if (count1[i] != count2[i])
return false;
return true;
}
int main(){
   cout<<isAnagram("eat","tea")<<endl;
   cout<<isAnagram("abc","def")<<endl;

}
```