Mon Tue Wed Thu Fri Sat

Date: __/__/__

2019-CE-04(Saba)

Lab2

Question no. 1:

a) Corrrect.
b) Corrrect
c) Incorrect. Because more time is required to convert base b to decimal within Karatsuba algorithm. But multiplication decreases.

Question no 2:

(a) Algorithm

```
def DistinctUsers(A):
    users = [ ]
    for i in range (0, len(A)):
        a=[K for K in range(A[i][0], A[i][1]+1)]
        fo j in range (i+1, len(A)):
            b=[l for l in range(A[j][0], A[j][1]+1)]
            if (a[len(a)-1] in b) or
                (b[len(b)-1] in a):
                users.append(i+1, j+1)
    return users
```

Date: __/__/__

Mon Tue Wed Thu Fri Sat

Description:

Input: A List of tuple. Each tuple has a entring and leaving time.

Output: A List of tuples of distinct users.

User id = index of tuple + 1.
Each tuple compare with every other in a list and checks if they are distinct. If they are distinct we append their user id in the form of tuple to users list.

Running time:

As it is nested loop, so its running time is $O(n^2)$.

Date: ___/___/___

2019-CE-041SAR

Question no.3:

(a) Pseudocode

```
1. def IstrustWorthy(t1, t2)
      // Input: a tuple of two toads
      // Output: What one toad says
                 about others and return
                 true if it is trustworthy.

      num = 0
1. def toad-to-toadComparison(toadsList)
2.    for i in range(0, lengt(todslist):
3.        for j in range(1, length(toadslist)):
4.            if IstrustWorthy(toadlist[i], toadslist[j]):
5.                num += 1
6.    return num
```

Explanation:

Function toad-to-toadComparison compares each toad with every other toad and checks if it is trustworthy using IstrustWorthy function which returns true if t1 is trustworthy and finally it incremented num.

2019-CE-04(SABA)

(b)

- We have exactly n/2 comparisons.
- As there are more trustworthy toads, the output has at least one toad.
- We compares two consecutive toads if both are trustworthy, we randomly choose one from them and if they are different, we removes both.

```
def even_toads(toadslist)
    if len(toadslist) == 2:
        del(toadlist[1])
    for i in range(0, length-1, 2)
        if toadslist[i] and toadlist[i+1] are
            both trustworthy
            del(toadslist[1])
        else
            del(toadslist[i:i+2])
```

(C)

Procedure:

If we have only one toad in a list, we return the same toad. Otherwise we check if toad list is even or odd if it is odd we remove any one of the toad and work on the rest of list as in case of even.

(d) **Procedure:**

Input: List of toads
Output: Single trustworthy toad.

List is dividing after each iteration using algorithms described in part (b) and (c) and we obtain only one trustworthy toad at last.

(e)

Base case: When length(toads) = 1. The only one toad in a list always trustworthy according to conditions.

Inductive step and conclusion: If program works for n-K. Then it should work for n=K+1 becaus even-toads and odd toads divide the list after each iteration of while loop.

(f)

Running time: As there are only one loop iterates through list of toads. So, it is O(n).

Date: __/__/__

2019-CE-04(SABA)

(g)

We can find all trust worthy toads by using toad-to-toad comparison and IsTrustworthy functions. as in part a.