Assignment 1 Data Structures & Algorithms. (PRD190001)

Q1] Suppose your calculator only did base 10 algorithms. write an expression to compute log base 2 of 2048 using only log base 10.

→ $\log_A B = \frac{\log_c B}{\log_c A}$; $c > 0$

$\therefore \log_2 2048 = \frac{\log_{10} 2048}{\log_{10} 2}$

Q2] Express the following summation in closed form (an expression that can be directly computed from k)

$3 + 5 + 7 + 9 + \ldots + 2k + 1$

→ $2(1 + 2 + 3 + 4 \ldots) + (1 + 1 + 1 + 1 \ldots)$

$= 2(n(n+1)/2) + n$

$= 2(k(k+1)/2) + k$

$= k(k+1) + k$

Q3] Prove by counter example.
Prove the following statement is false.
$n^3 > 2^n$ for any $n \geq 1$

→ $10^3 = 1000$ $\quad \therefore 10^3 \not> 2^{10}$

$2^{10} = 1024$ $\quad \therefore$ The above statement is false.

Q4] Prove by contradiction.

Prove the following statement is true.

"The square of even numbers is also even"

→ Let us assume that square of even number is odd.

∴ let $2n$ be any even number &

$4n^2 + 1$ be its square.

Thus by squaring the even number $2n$

we get $2n \times 2n = 4n^2$.

$$4n^2 \neq 4n^2 + 1$$

Also $4n^2$ is an even number.

Hence our assumption was incorrect & the square of even numbers is always even.

Q5a) Prove by induction

$$\sum_{i=1}^{n} i^3 = [n^2][(n+1)^2]/4$$

Base case :-

For $i = 1$

$$\sum_{i=1}^{1} 1^3 = 1^2 [(1+1)^2]/4$$

$$= 1$$

This holds true.

Let us assume it holds true for k (until k)

$$\sum_{i=1}^{k} i^3 = (k^2)[(k+1)^2]/4$$

∴ for k+1

$$\sum_{i=1}^{k+1} = i^3 = (k^2)[(k+1)^2]/4 + (k+1)^3$$

$$= \frac{[k^2][(k+1)^2] + 4(k+1)^3}{4}$$

$$= \frac{[k^2+4k+4][k+1]^2}{4}$$

∴ we have proved that

$$\sum_{i=1}^{n} i^3 = [n^2][(n+1)^2]/4 \text{ holds true.}$$

Q5b) $n^2 - n$ is even for any $n \geq 1$

Base case:- for n = 1

$1^2 - 1 = 0$ is true.

Let us assume that this holds true until k

∴ $k^2 - k =$ ~~Even~~ true.

∴ for (k + 1)

$$(k+1)^2 - (k+1) = k + 2k + 1 - k - 1$$

$$= \underline{2k \leftarrow \text{even}}$$

∴ $n^2 - n$ is even for any $n \geq 1$

Q6] Write Java or pseudocode:

a) write a recursive function when passed a value displays

n(n-1)(n-2)(n-3)... 0 ...(n-3)(n-2)(n-1)n

→

```
FUNCTION PRINTORDER(n){
    if n>0 {
        print(n)
        PRINTORDER(n-1)
        print(n)
    }
    else if n==0 {
        print(0)
    }
```

Q6b] write a recursive function that receives an array of integers and a position as parameters and returns the count of odd numbers in array. Let each recursive call consider next integer in array.

→ Answer on next page.

→ FUNCTION COUNTODD (arr, pos) {

```
if pos > arr.length - 1 {
   print (incomect position)
}
else if pos <0 {
   print(incomect position)
}
else if pos == arr.length - 1 {
    if arr[pos] %. 2 == 0 {
        return 0 }
     else return 1
}
else {
   if arr [pos] %. 2 == 0
      return COUNTODD (arr, pos + 1)

   else
      return 1 + countodd (arr, pos + 1)
}
```

Q7) Suppose there exists a generic Java class named Pair with type parameter T that stores two objects with get & set method for each. write the statements necessary to create an object of type pair with string as its type parameter, & use the set method to set the two strings, then the get methods to retreive it for printing. Note: You dont have to write the pair class itself.

→ Pair<String, String> p = new Pair<String, String>;

p.setObject1("hello");

p.setObject2("world");

String p1 = p.getObject1();

String p2 = p.getObject2();