| Serial No: |
| --- |
| **Sessional Exam-I** |
| **Total Time: 1 Hour** |
| **Total Marks: 60** |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Signature of Invigilator |

| **CS-2009: Design & Analysis of Algorithms** |
| --- |
| Monday, 25th September, 2023 |
| **Course Instructors** |
| Dr. Ramoza Ahsan, Laraib Afzaal |

| **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  Student Name Roll No. Course Section Student Signature |
| --- |

## DO NOT OPEN THE QUESTION BOOK OR START UNTIL INSTRUCTED.

**Instructions:**

1. Attempt on question paper. Attempt all of them. Read the question carefully, understand the question, and then attempt it.
2. No additional sheet will be provided for rough work. Use the back of the last page for rough work.
3. If you need more space, write on the back side of the paper and clearly mark question and part number etc.
4. After asked to commence the exam, please verify that you have **seven (7)** different printed pages including this title page. There are total of **4** questions.
5. Calculator sharing is strictly prohibited.
6. Use permanent ink pens only. Any part done using soft pencil will not be marked and cannot be claimed for rechecking.

|  | **Q-1** | **Q-2** | **Q-3** | **Q-4** | **Total** |
| --- | --- | --- | --- | --- | --- |
| **Marks Obtained** |  |  |  |  |  |
| **Total**  **Marks** | **20** | **20** | **10** | **10** | **60** |

**Question 1 MCQs [20 Marks]**

**Question 1-a: Complexity of following snippet of code is**

c=0

for(i=0;i<n;i++)

for(j=m,j>0;j-=2)

c++

1. O(n2)
2. O(nm)
3. O(m2)
4. O(n+m)

**Question 1-b: Complexity of following snippet of code is**

c=0

for(i=0;i<n;i++)

for(j=m,j>1;j/=2)

c++

1. O(n2)
2. O(nm)
3. O(m2)
4. O(nlogm)

**Question 1-c: Notation that denotes upper bound of an algorithm is**

1. O
2. Ω
3. Θ
4. None of the above

**Question 1-d:** **O(nb) and O(bn) have the same order of growth**

1. True
2. False

**Question 1-e:** 2n2= Ω (n)

1. True
2. False

**Question 1-f: If f(n) = O(g(n)) and g(n) = O(h(n)), then f(n) = O(h(n))**

1. True
2. False

**Question 1-g: Which sorting techniques have best case complexity as O(n)**

1. Bubble Sort
2. Insertion Sort
3. Selection Sort
4. All of the above
5. Both i and ii
6. Both i and iii
7. Both ii and iii
8. iv

**Question 1-h: In the worst case, selection sort makes O(n2) swaps**

1. True
2. False

**Question 1-i: Which sorting technique is preferred in cases where writing to memory is significantly more expensive than reading**

1. Bubble Sort
2. Insertion Sort
3. Selection Sort
4. All of the above

**Question 1-j: For the following snippet of the code, what will be its recurrence equation**

public int g(int n) {

if (n == 1)

return 2;

else

return 3 \* g(n / 3) + g( n / 5) + 5;

}

1. T(n) { c n==1  
    T(n/3) + T(n/5)+b n>1 }
2. T(n) { c n==1   
    2T(n/3) +b n>1 }
3. T(n) { cn n==1  
    T(n/3) + T(n/5)+b n>1
4. None of the above

**Question 2 Short Questions [20 Marks]**

**Question 2 a-c: For the following code snippets, write down the complexity function and then express it in the form of O notation**

|  | **Code** | **Analysis** | **O()** |
| --- | --- | --- | --- |
| 2-a: | for i=1 to n{  a++  for j=1 to 2\*i{  k=j  while (k>=0){  k=k-1} | **8n3+4n2+2n+1** | **O(n3)** |
| 2-b | for (i=n/2; i<=n; i++){  for(j=1; j<=i; j++){  for(k=1; k<=m; k=k\*2{  cout<<”This is easy”}  }  } | **3n/2+3n2+3logn+1** | **O(n2logm)** |
| 2-c | for(i=n; i>1; i=i/2)  sum+=i  for(j=1; j<m; j+=2)  sum+=j  for(k=1;k<=8;k++)  sum+=k | **4logn+4m+35** | **O(logn+m)** |

**Question 2-d: Find the upper O and tight bound Θ of the function f(n) (Also prove it mathematically)**

f(n)=3n4+6n2+2

for tight bound

c1g(n)<=f(n)<=c2g(n)

3n4<=3n4+6n2+2<=3n4+6n4+2n4

3n4<=3n4+6n2+2<=11n4

C1=3, c2=11, n0=1, for n>=1 equation holds

| n | 3n4+6n2+2 | 11n4 |
| --- | --- | --- |
| 1 | 11 | 11 |
| 2 | 74 | 176 |

**O(n4)**

**Θ(n4)**

**Question 2-e: Find the Recurrence relation of the following code.**

long power(long x, long n){

if(n==0)

return 1;

else

return x\*power(x,n-1)

}

T(n)= { b n==0,

T(n-1) + c, n>0

}

**Question 3 Problem Solving [10 Marks]**

**You have 81 quarters (coins) and a balance. You know that 80 quarters have the same weight, and one weighs less than the others. Give an algorithm (in pseudocode) to identify the light quarter which uses the balance only 4 times in the worst case.**

***Hint: Think Divide and Conquer strategy***

Procedure Find-Light-Quarter()

**For** i=1 to 4

//divide in 3 groups

Put 81/3i in two pans of balance

**If** left(pan)=right(pan)

**then** Discard quarters on these left and right pan

**else if** Left(pan) is lighter

**then** Discard the right pan quarters and quarters not on balance

**else** Discard the left pan and quarters not on balance

end procedure

Working Example: Highlighted one shows the group selected for next iteration

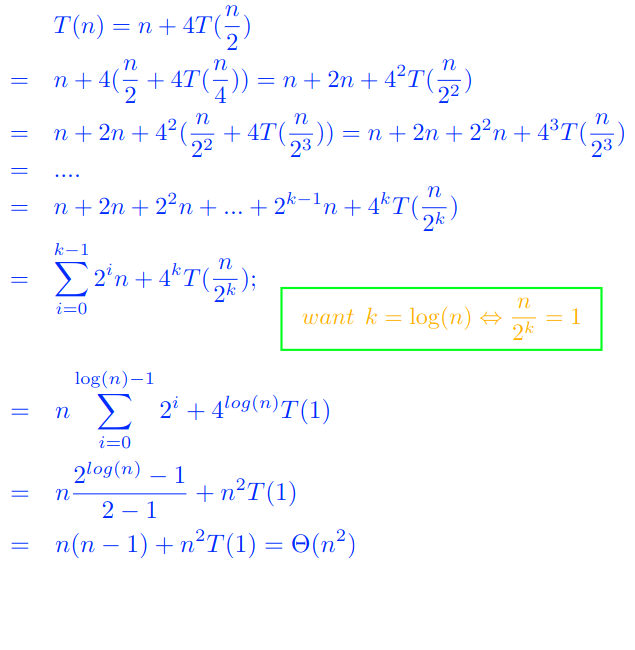
| No. of times Balance Used | Left Pan | Right Pan | Not on Balance |
| --- | --- | --- | --- |
| 1 | 27 | 27 | 27 |
| 2 | 9 | 9 | 9 |
| 3 | 3 | 3 | 3 |
| 4 | 1 | 1 | 1 |

In end left with one 1 quarter on each pan, if they are equal they are discarded, quarter not on balance is the lighter one. Otherwise the quarter on lighter side is the result as other two are discarded

**Question 4 Recurrence Equation [10 Marks]**

**Solve the following recurrence relation using Iteration method.**

**T(n) = 4T(n/2) + n , T(1)=1**

****