Show your work! Give the best result that you can give!

NOTICE TO THE STUDENTS
Read the instructions carefully listed below and sign the box:
1. Textbooks, lecture notes, calculators with extensive memories, and any kind of computers are not permitted in the classrooms: if you have any, leave them on the instructors desk.
2. Cell phones should be totally switched off (not in silent or flight modes) and do not keep them with you: either put them in your bags or leave them on the instructors desk.
3. Permitted material to be kept on your desks are; pencils, sharpeners, erasers (and in case you may need: water and tissues). Pencil boxes are strictly forbidden.
4. Check your desk for any graffiti; the graffiti related to the course will be treated as an attempt to cheat.
5. You are not allowed to talk to other students during the exam whatever the reason may be.
6. Disobeying the above rules will be severely penalized and a disciplinary action will be conducted.
7. Please prepare your IDs (with photos) on your desk for identity check.

(to be signed when exam is finished!)
I certify that this is my own work only.
Name: ......................................
Signature: ..................................
Time: ........................................

1. Suppose you have an algorithms \( A_1, A_2, A_3 \) Input the this problem naturally divided into 4 groups, and You happen to know performance of the algorithm for each groups of input, in terms of running time, of these algorithms over all possible inputs of size \( n \).

<table>
<thead>
<tr>
<th>Group</th>
<th>complexity of ( A_1 )</th>
<th>Complexity of ( A_2 )</th>
<th>Complexity of ( A_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group 1</td>
<td>( 2.5n^5 + 1000 \log n + 10^{100} )</td>
<td>( n \log n + 10^9 )</td>
<td>( n^2 \log n )</td>
</tr>
<tr>
<td>Group 2</td>
<td>( 100n^2 \log n + 0.001 \times n^2 + n \log n + n^3 )</td>
<td>( 10^4n \log n + n^2 \log n )</td>
<td>( 10^8n + 0.3n^2 \log n )</td>
</tr>
<tr>
<td>Group 3</td>
<td>( 10^6n \log n + 0.01n^2 + 5.1n^4 )</td>
<td>( 0.01 \times 2^n )</td>
<td>( n^3 + 200n^2 + 0.5n^2 \log n )</td>
</tr>
<tr>
<td>Group 4</td>
<td>( + 1000n + 10^6n^3 )</td>
<td>( n^4 + 100n \log n )</td>
<td>( 0.001n^2 \log n )</td>
</tr>
</tbody>
</table>

Give performance of \( A_1, A_2, A_3 \) in terms of \( O(\cdot), \Omega(\cdot), \Theta(\cdot) \) notation. 9 pts
2. Solve the following recursion via Master Theorem, if possible. Show your calculations and justify your results. [14 pts]

- \( T(n) = 5T(n/4) + \Theta(n^2) \)

- \( T(n) = 6T(n/2) + \Theta(n^2 \log n) \)

- \( T(n) = 6T(n/3) + \Theta(n) \)

- \( T(n) = 5T(n/2) + \Theta(n \log n) \)

- \( T(n) = 8T(n/2) + \Theta(n^3 (\log n)^3) \)

- \( T(n) = 4T(n/2) + \Theta(n^2 / \log n) \)

- \( T(n) = T(n/3) + \Theta((\log n)^2) \)

3. Given n positive integers \( x_i \) each between 1 and k, sorting can be done with Counting sort in \( O(n+k) \); for simplicity, assume that is n+k. Given n positive integers on a 64 machine we want to use Counting sort. We have the option of representing each \( x_i \) as a \( 2^t \) base number with \( 64/t \) digits (that is each \( x = y_1y_2...y_s \) where \( s=64/t \) and each \( y_j \) is between 0 and \( 2^t \). Compute total cost of radix sorting for \( t=8 \) and \( t=16 \), and identify cheaper one for \( n = 2^{10} \) and \( n = 2^{20} \). [6 pts]
4. Assume \( X \) is an algebraic object whose multiplication is relatively expensive. We want to compute \( X^n \) for \( n=127, 512, 195, 1030 \). How can we compute \( X^n \) with minimum number of multiplications. Write explicitly, number of multiplications; list of multiplications 8 pts

5. Recall that for \( f(n) = f_n \), nth Fibonacci number the matrix \( A \) satisfies with \( f(0) = 0, f(1) = 1 \), and

\[
 f(n + 2) = f(n + 1) + f(n) \quad (**) \quad A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \quad \text{and} \quad A^n = \begin{bmatrix} f(n + 1) & f(n) \\ f(n) & f(n - 1) \end{bmatrix}
\]

Let us assume that multiplication of two 2x2 matrices costs 100 operation whereas \((**)\) costs one operation (addition/subtraction) (similarly \( f(n) = f(n+2) - f(n+1) \)) . 3+3 pts

- Considering \( n = 2^k \), what is smallest \( k \) for which computing \( f(n) \) from \( A^n \) is cheaper than computing via \((**)\) ?
- for \( n=4150 \) what is the cheapest ways of computing \( f(n) \) as a combination of \( A^{2^k} \) and \((**)\) ?
6. **Order Statistics** Given a list of size 11 we want to find a median element of the list using partition algorithm. Take the first element of the list in question as the pivot element, and use the partition algorithm as discussed in class. For each recursive call to partition algorithm, indicate pivot element, (with a circle) the list input to partition algorithm, and order of the element which is sought (e.g. 5th, 7th etc).

<table>
<thead>
<tr>
<th>n</th>
<th>i</th>
<th>list</th>
<th>partition</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>7</td>
<td>3 5 8 20 15 9 14 4 25 40</td>
<td></td>
</tr>
</tbody>
</table>
7. Given the elements \{10, 20, 3, 5, 8, 50, 15, 25, 30, 200, 14, 24, 150, 54\} within an array A. Put these in a heap via Heapify! Show your calculations using trees. Works with levels \[6\ \text{pts}\]
8. Insert into the given heap: 175 then 300 successively [6 pts]
9. apply deletemax twice to given Heap [6 pts]
10. **Hashing - Double Hashing**  
We want to place the following items in a hash table: defined by mod 16: 21, 5, 37, 53, 69, 22, 24, 56

- Let \( h_1(k) = k \mod 16 \) and \( h'_2(k) = k \mod 13 \) and \( h_2(k) = h'_2(k) \) if \( h'_2(k) \) is odd, and \( h_2(k) = h'_2(k) + 1 \) otherwise. Let \( h(k, i) = (h_1(k) + ih_2(k)) \mod 16 \). Using the sequence \( h(k, i) i=0, 1, 2 \), ... place the above numbers a Hash table: **10 pts**

- place the items via linear probing **5 pts**
11. Given the binary search tree $T_o$, each time starting with $T_o$, 5 + 5 pts

a) insert 38

```
20
  /   \
15    40
  /     /
10    17  30  86
    /    /
   12    25  35
```

b) insert 14

```
20
  /   \
15    40
  /     /
10    17  30  86
    /    /
   12    25  35
```
12. Given the binary search tree $T_o$, delete 50 [10 pts]