CSc22000 - Algorithms
Spring 2021 Midterm Exam
Instructor: Ahmet C. Yuksel
12 April'20
Time Limit: 1hr 15 minutes + 15 minutes to Scan/Upload/Send

Name (Print): ______________________

This exam contains 6 pages (including this cover page) and 10 problems. Check to see if any pages are missing. Enter all requested information on the top of this page, and put your initials on the top of every page, in case the pages become separated.

You may *not* use your books, notes, or any calculator on this exam.

You are required to show your work on each problem on this exam. The following rules apply:

- **Organize your work**, in a reasonably neat and coherent way, in the space provided.
  Work scattered all over the page without a clear ordering will receive very little credit.
- **Mysterious or unsupported answers will not receive full credit**. A correct answer, unsupported by calculations, explanation, or algebraic work will receive no credit; an incorrect answer supported by substantially correct calculations and explanations might still receive partial credit.
  Abbreviations, shortcuts and the handwriting that the instructor cannot read will receive no credit.
- If you need more space, use the back of the pages; clearly indicate when you have done this.

Do not write in the table to the right.

Problem	Points	Score
1	8	
2	16	
3	8	
4	9	
5	10	
6	10	
7	10	
8	6	
9	15	
10	8	
Total:	100	

1. Compare the following functions based on their growth rate. Use the proof and the proper notations to classify them.
   (a) (2 points) $g(x) = log_e^3 x$ vs $f(x) = \sqrt{\ln^2 x}$
   (b) (2 points) $g(x) = log_2(x!)$ vs $f(x) = \ln(1 + x^4)^4$
   (c) (2 points) $g(x) = 3x^2 + x^2 \log_2 x$ vs $f(x) = 2000x + log_2(x!)$
   (d) (2 points) $g(x) = 4^{log_2 x}$ vs $f(x) = 2^x$

2. (16 points) Solve the following recurrence relation:
   $W(n) = \Theta(n) + 4W(n/16) + 4W(n/16) + 8W(n/16)$
   $T(0) = 0$
   $T(1) = 1$
   After obtaining your solution, use Master Theorem to verify (prove) your solution. You may use only the homogeneous/inhomogeneous recurrence solution. Other methods will not be accepted.

3. (8 points) What is the purpose of the code below, also write its running time by using the proper notation.

```
My_Function(X)
    for i=2 to X.length
        key=X[i]
        current=i-1
        while current > 0 && X[current] < key
            X[current+1]=X[current]
            current=current-1
        X[current+1]=key
```

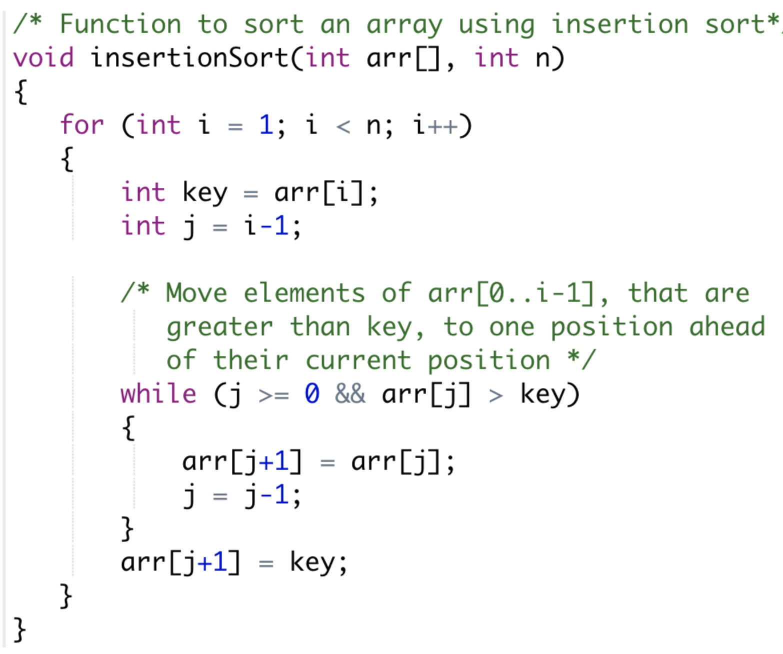
4. (9 points) Analyse the running time of the following code. Use the proper notation.

```
for (int i=1; i<n; i=i*e)
{
    for (int i=2; i<n; i=i*i)
    {
        if (i<999999)
        {
            cout << i << "\n"
        }
    }
}
```

5. (10 points) Use Quicksort algorithm that picks a random element as the pivot to sort the following array. Implement your algorithm that uses the same mechanism that we used in class. Show all the steps for at least one half of the array. Also write the worst case, best case time complexity of the Quicksort algorithm. Is Quicksort stable and in-place?

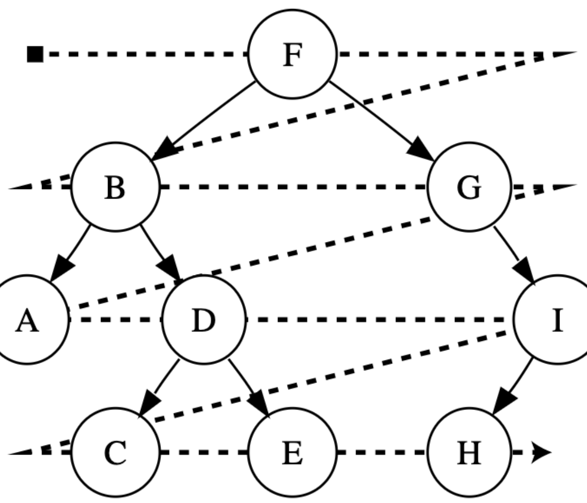
   A=[-3,-7,-9,-11,-10,-2,-3,-4,-6,-12]

6. (10 points) How much time does the following insertion sort code below take to sort an array of n elements in the following form: arr[] = 2, 1, 4, 3, 6, 5,. . . .i, i-1, ...n, n-1 ?



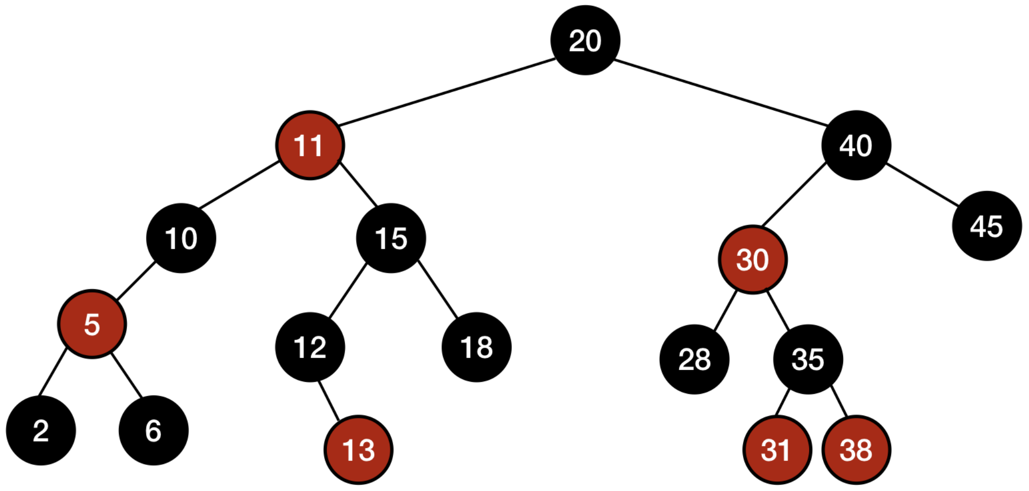
```
/* Function to sort an array using insertion sort*/
void insertionSort(int arr[], int n)
{
   for (int i = 1; i < n; i++)
   {
       int key = arr[i];
       int j = i-1;

       /* Move elements of arr[0..i-1], that are
          greater than key, to one position ahead
          of their current position */
       while (j >= 0 && arr[j] > key)
       {
           arr[j+1] = arr[j];
           j = j-1;
       }
       arr[j+1] = key;
   }
}
```

7. (10 points) Consider the BST below. Write a proper pseudocode runs in $\Theta(n)$ to list all its elements based on the height it is stored at. Sample output: [F,B,G,A,D,I,C,E,H]



8. (6 points) Draw a non-empty B-Tree where the degree is 4 and the height is 3. Place some keys to the nodes. Your tree must contain the biggest node as well as the smallest node in size.

9. The tree below is a RBT.



   (a) (3 points) Insert 38.
   (b) (3 points) Remove 20.
   (c) (3 points) Remove 40 from the the resulting tree.
   (d) (3 points) Remove 45 from the resulting tree.
   (e) (3 points) Remove 18 from the the resulting tree.

   Make sure to mention the case(s) that you meet in each operation.

   You must mention the insertion and deletion case(s) that you meet in each operation to get points.

10. (8 points) Draw a recursion tree that finds the running time of

$$T(n) = T(3n/4) + T(n/5) + \Theta(n^2)$$