You have to do all the 6 problems in this section of the exam.
Partial credit cannot be given unless all work is shown and is readable.

Be complete, yet concise, and above all be neat. Do Your rough work on the last page.
1. Do the following problems using summations. All quantities are integers.
[2 + 2 + 3 + 6 pts]

a) What would be the value of final after the following segment has been executed?

```
final = 0;
for ( i= 0 ; i <= 50 ; i++)  final += 20;
```

\[ \sum_{i=0}^{50} 20 \]

= 20 \( 50 - 0 + 1 \)

= 20 \( 51 \)

= 1020 (2 POINTS)

AWARD ONLY 1 POINT IF SUMMATION STARTS FROM 1 and ANSWER IS 1000

b) What would be the value of num after the following segment has been executed?

```
total = 0;
k  = 20;
um = 0;
for ( i= 1 ; i < 5 ; i++) {
    for ( j = 1; j <= 30 ;  j++)
        num   += 6 * k;
    total += 3*j – 6 ;
}
```

\[ \sum_{i=1}^{4} \sum_{j=1}^{30} 6k \]

\[ \sum_{i=1}^{4} 180k \]

= 720 k

= 720 (20)

= 14400 (3 POINTS)

AWARD ONLY 1 POINT IF FIRST SUMMATION GOES UPTO 5 AND RESULT IS 18000
c) What would be the value of count after the following segment has been executed?

```c
count = 0;
sum = 0;
for ( i = 1; i <= 5; i++) {
    for ( k = 1; k <= 10; k++)
        count += 2 * i;
        sum += 3 * k - 6;
}
```

\[
\text{count} = \sum_{i=1}^{5} \sum_{k=1}^{10} 2i
\]

\[
= \sum_{i=1}^{5} 20i
\]

AWARD 1 POINT IF THIS IS CORRECT

\[
= 20 (5)(6) / 2
\]

\[
= 300
\]

AWARD ANOTHER 2 POINTS IF THIS PART IS CORRECT OTHERWISE

AWARD ZERO
d) Work out an expression for total after the following segment has been executed. 
Note that \( n \) is a even integer ( \( n > 0 \)).

\[
\text{total} = 0;
\]
\[
\text{for} \ ( k= 1 ; \ k < 2 ; \ k++) \ { \ 
\text{for} \ ( i = n/2; \ i <= n ; \ i++) \\
\text{total}+ = 16*i - 12*n 
}
\]

First 'for ' loop goes only once, so need to consider it separately

\[
\sum_{i=n/2}^{n} 16i -12n
\]

\[
\sum_{i=n/2}^{n} 16i - \sum_{i=n/2}^{n} 12n
\]

AWARD 1 POINT IF SPLIT INTO 2 SUMMATIONS IS PROPER

THE FIRST SUMMATION SHOULD BE SPLIT UP AS SHOWN BELOW. ( A COMMON MISTAKE IS TO TAKE IT UPTO \( n/2 \))

\[
16i - 12n + n - n/2 + 1
\]

AWARD 3 POINTS IF SECOND SUMMATION GOES UPTO \( n/2 - 1 \)
SOLUTION PROCEEDS AS FOLLOWS IN THAT CASE

\[
\frac{16n(n+1)}{2} - \frac{16(n/2)(n/2 - 1)}{2} - 12n(n/2 + 1)
\]

\[
= 8n(n+1) -2(n)(n-2) - 6n(n+2)
\]

\[
= 8n^2 + 8n - 2n^2 + 4n - 6n^2 -12n
\]

\[
= 0 \ (\text{correct answer})
\]

AWARD last 2 POINTS ONLY IF RESULT IS ZERO
IF SECOND SUMMATION has been taken upto n/2 **AWARD 1 POINT AT THIS STAGE.**
the solution should **PROCEED AS FOLLOWS:**

\[
\begin{align*}
&= \frac{16n(n + 1)}{2} - \frac{16(n/2)(n/2 + 1)}{2} - 12n(n/2 + 1) \\
&= 8n(n + 1) - 2(n)(n + 2) - 6n(n + 2) \\
&= 8n^2 + 8n - 2n^2 - 4n - 6n^2 - 12n \\
&= -8n \ (\text{answer})
\end{align*}
\]

**AWARD last 2 POINTS ONLY IF RESULT NOW IS -8n**
2. [8 pts] :1 POINT FOR EACH CORRECT ANSWER

a) A very large array contains \( m \) elements sorted in the reverse order (largest to smallest). What would be the time complexity in terms of Big-O, when the following sorting algorithms are applied on this array? Quick sort uses first element as the pivot.

- Merge sort: O(n log n)
- Bubble sort: O(n^2)
- Quick sort: O(n^2)
- Insertion sort: O(n^2)

b) After the array is sorted in proper order using one of the methods mentioned above the following sorting algorithms are applied on the sorted array. Indicate the time complexity in each case.

- Bubble sort: O(n)
- Selection sort: O(n^2)
- Quick sort: O(n^2)
- Merge sort: O(n log n)

3. [5 pts] Indicate the worst case time complexity of following operations in terms of Big-O.

:1 POINT FOR EACH CORRECT ANSWER

a) Searching for a target on a binary tree: O(n)
b) Searching for a target on a binary search tree: O(n)
c) Pre-order traversal of a binary search tree: O(n)
d) Counting leaves of a binary tree: O(n)
e) Finding height of a binary search tree: O(n)
4. [8 pts] Write a recursive function `int countbin ( int D)`, which counts the number of ONES in the binary representation of a given decimal integer `D`. Work out the time complexity of the function in terms of Big-O.

```c
int countbin ( int D)
{
    if (D==0)
        return 0;
    else
        return (n%2)+countbin(D/2);
}
```

(PARTIAL POINTS MAY BE AWARDED)

Recurrence relation:

\[ T(n) = T(n/2) + 3 \]
\[ T(1) = 3 \]

:4 POINTS

:1 POINT

\[ T(n/2) = T(n/4) + 3 \]
\[ T(n) = T(n/4) + 3.2 \]
\[ T(n/4) = T(n/8) + 3 \]
\[ T(n) = T(n/8) + 3.3 \]

Or

\[ T(n) = T(n/2^k) + 3.3 \]

:1 POINT

General case
\[ T(n) = T(n/2^k) + 3.k \]

Let \( n = 2^k \)

Which yields \( k = \log n \)

:1 POINT

\[ T(n) = T(1) + 3 \log n \]

= \( O(\log n) \)

= \( O(\log D) \)
5. [8 pts] a) Draw a binary search tree with the nodes arriving in the order shown below. After every insertion, if the tree gets unbalanced, **redraw** the balanced AVL tree.

50, 80, 30, 20, 70, 60, 55, 10

50
/ \\n/ / \\
30 80
/ / / \\
20 70
/ / / \
60

:2 POINTS

50
/ \\n/ / \\
30 70
/ / / \\
20 60 80
/ / / 
10 55

:2 POINTS

50
/ \\n/ / \\
20 70
/ / / \\
10 30 60 80
/ / \\
55

:1 POINT
b) Draw a binary search tree with the nodes arriving in the order shown below. After every insertion, if the tree gets unbalanced, redraw the balanced AVL tree.

\[
50, \ 30, \ 80, \ 90, \ 60, \ 70
\]

\[
\begin{array}{c}
50 \\
/ \ \\
/ \ \\
30 \ 80 \\
/ \ \\
/ \ \\
60 \ 90 \\
\end{array}
\]

\[
\begin{array}{c}
60 \\
/ \ \\
/ \ \\
50 \ 80 \\
/ \ \\
/ \ \\
30 \ 70 \ 90
\end{array}
\]

: 3 POINTS
(NO PARTIAL POINTS)
Write a function which deletes the largest element on a binary search tree. The nodes of the tree have the following structure:

```c
struct node {
    struct node * left;
    int data;
    struct node * right;
};

struct node * largest( struct node *p)
{
    // TAKE CARE OF NULL TREE : 1 POINT
    if (p==NULL)return NULL;
    // TAKE CARE OF ROOT BEING THE LARGEST NODE : 3 POINTS
    if (p->right==NULL)
        p= p->left;
    // LARGEST NODE MAY HAVE A LEFT SUBTREE : 2 POINTS
    else
    {
        if (p->right->right==NULL)
            p->right= p->right->left;
        else
            // PROPER RECURSION TO FIND THE LARGEST NODE : 2 POINT
            p = largest(p->right);
    }
}
```