Faculty : Dr. S. Prasanna Devi.

PART B

1. What is an Algorithm?
2. Write the algorithm for GCD calculation?
3. What is algorithm design Technique?
4. Differentiate time and Space efficiency?
5. Design an algorithm to compute the area and Circumference of a circle
6. List the important problem types.
7. How will you measure input size of algorithms
8. Define best, worst and average case efficiency?
9. Define big oh(O),Big omega(Ω) and big theta(Θ) notations
10. List the basic efficiency classes
11. Define recurrence relation?
12. What is non recursion relation?
13. Define nonrecursive algorithm?
14. What does this algorithm compute? How many times is the basic operation executed?
15. Write an algorithm using recursive function to find the sum of n numbers.
16. List the factors which affects the running time of the algorithm.
17. What is meant by substitute methods?
18. Write the general plan for analyzing Time efficiency of recursive algorithm.

PART C

1. Discuss in detail about fundamentals of algorithmic problem solving?
2. Explain the necessary steps for analyzing the efficiency of recursive algorithms
3. Explain the general framework for analyzing the efficiency of algorithm.
4. Write the asymptotic notations used for best case ,average case and worst case analysis of algorithms and Write an algorithm for finding maximum element of an array perform best , worst and average case complexity with appropriate order notations.

5. Explain the method of solving recurrence equations with suitable example.
6. Explain the method of solving Non recursive equations with suitable examples .
7. i)Describe the basic efficiency classes in detail. ii) Write an algorithm for Fibonacci numbers generation and compute the following a) How many times is the basic operation executed b) What is the efficiency class of this algorithm.
8. Solve the following recurrence relations

a) $x(n)=x(n-1) + 5$ for $n > 1$ $x(1)=0$

b) $x(n)=3x(n-1)$ for $n > 1$ $x(1)=4$

c) $x(n)=x(n-1)+n$ for $n > 0$ $x(0)=0$

d) $x(n)=x(n/2)+n$ for $n > 1$ $x(1)=1$ (solve for n=2k)

e) $x(n)=x(n/3)+1$ for $n >1$ $x(1)=1$ (solve for n=3k)

1. Consider the following recursion algorithm

```
Min1(A[0 -------n-1])
 If n=1
 return A[0]
 Else
temp = Min1(A[0…….n-2])
If temp <= A[n-1]
return temp
Else Return A[n-1]
```

a) What does this algorithm compute?

b) Setup a recurrence relation for the algorithms basic operation count and solve it.

9. Calculation of time/space complexity for renowned algorithms – Insertion sort, Bubble sort, Merge sort, Binary search.

PROBLEMS SOLVED IN CLASS

PROBLEMS IN SOLVING RECURRENCE RELATIONS

1. **Solve by recurrence tree T(n)=T(n-1)+n**

```
Void recursion test(int n)
{
if(n>0)
{
for( i=0; i<n; i++)
{
print(n);
}
recursion test (n-1);
}
```

2. **Solve T(n)=T(n-1)+n by backward recursion.**

3. **Find complexity using recursion tree T(n)=T(n-1)+log n.**
   Solve using fwd recursion.

```
Void recursion (n)
{
if(n>0)
{
for( i=1; i<n; i*2)
{
print(i)
}
recursion (n-1)
}
}
```

4. **Solve by recursion tree T(n)=T(n-1)+n^2.**

5. **Solve by recursion tree T(n)=T(n-2)+1.**

6. **Solve by recursion tree T(n)+T(n-100)+n.**

7. **Solve by recursion tree T(n)=2T(n-1)+1.**

8. **Solve by recursion tree T(n)=T(n-1)+n^3.**

FINDING TIME/SPACE COMPLEXITY

1. Algorithm swap (a ,b)

```
{
}
```

2. Algorithm sum (A ,X)

```
{
   }
3. Algorithm sum (A ,B ,n )
   {
   }
4. Algorithm multiply (A ,B ,X )
   {
   }
5. For ( i=1 ; i<n ; i=i*2 )
   {
   }
6. For ( i=n ; i>=1 ; i=i/2 )
   {
   }
7. For ( i=0 ; i*i<n ; i++ )
   {
   }
8. For ( i=0 ; i<n ; i++ )
   {
   }
   For ( j=0 ; j<n ; j++ )
   {
   }
9. For ( i=1 ; i<n ; i=i*2 )
   {
   P++;
   }
   For ( j=1 ; j<p ;j=j*2 )
   {
   }
```

10.
```
For ( i=0 ; i<n ; i++ )
{
For ( j=0 ; j<n ; j++ )
{
}
}
```

11.
```
i=0;
while (i<n)
{
stmt ;
i++;
}
```

12.
```
a=1;
while (a<b)
{
Strut;
a=a*2;
}
```

SOLVE ASYMPTOTIC NOTATIONS

1. Given f(n) = 2n+3 . Show f(n)=o(g(n)).
2. Compare functions :
   a) $f(n) = n^2 \log n$

      $g(n) = n(\log n)^{10}$
   b) Say true/false.
      1. $(n+k)^n = O(n^n)$.
      2. $2^{2n} = O(2^n)$.
      3. $n^{\log n} = O(2^n)$.
      4. $5n^2-6n = O(n^2)$.
      5. $n! = O(n^n)$.

6. $2n_2^{2n}+n\ \log(n) = \Theta(n^2 2^n)$.
7. $33n^3+4n^2 = \Omega(n^3)$.
8. $f(n)=2n+3=O(n)$.

   $f(n)=2n+3=O(n^2)$.

   $f(n)=\Omega(g(n))$.

   $f(n)=\Theta(g(n))$.
9. $10n^3+15n^4+100n^2 2^n=O(100n^2 2^n)$.
10. $n^{1.001}+n\ \log\ n = \Theta(n^{1.001})$.