Name-Abhinav Rawat
Class-BTech CST-SPL-1(44)
2017446

Date.
Page No.

Design and analysis of algorithm.

Tutorial 2

Ques1 What is time complexity of below code?

```
void fun(int n)
{
 int j=1, i=0;
 while(i<n)
 {
  i=i+j;
  j++;
 }
}
```

j	i
1	0
1	1
2	3
3	6
4	10
5	15

$S = 0+1+2+6+10+15+ \ldots\ldots\ldots T_k$ — (1)

also $S = 0+1+3+6 \ldots\ldots T_{k-1} + T_k$ — (2)

from 1-2

$\Rightarrow 0 = 1+2+3+4+\ldots\ldots k - T_k$

$T_k = 1+2+3+4+\ldots +k$

$T_k = \frac{1}{2} k(k+1)$

Abhinav

Date.
Page No.

for k iteration

$1+2+3+6+\ldots\ldots k<n$

$\Rightarrow \frac{k(k+1)}{2}<n$

$\frac{k^2+k}{2}<n$

S.B.S

$\sqrt{\frac{k^2}{2}+\frac{k}{2}}<\sqrt{n}$

$k \simeq O(\sqrt{n})$

$T(n)<O(\sqrt{n})$

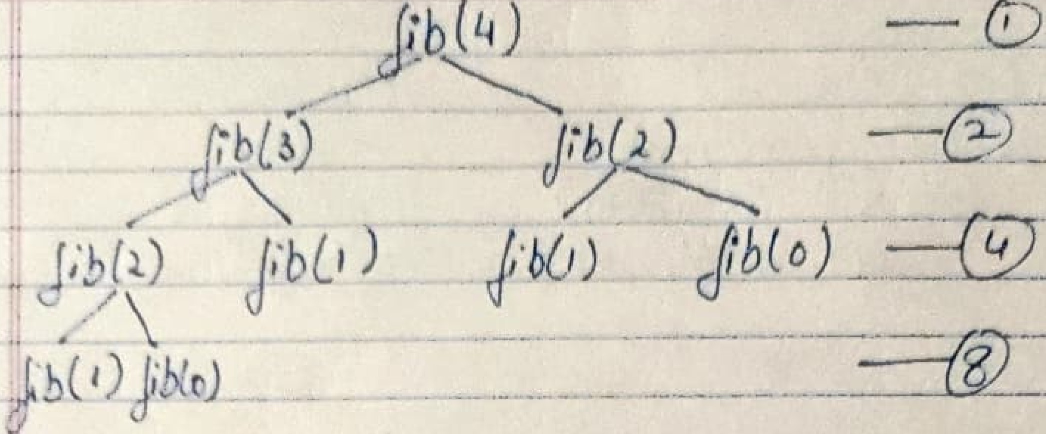
Que2 Write recurrence relation for recursive fn that prints fibonacci series. Solve the recurrence relation to get time complexity of the program. What will be the space complexity of this program and why?

0 1 1 2 3 5 . . . . n

```
int fib(int n)
{
  if(n<=1)
     return n;                    — O(1)
  return fib(n-1)+fib(n-2) — T(n-1)+T(n-2)
}
```

$T(n)=T(n-1)+T(n-2)+1$

Ahuiran



$T(n) = 1 + 2 + 4 + 8 + \ldots n$

$S(n) = \dfrac{a(r^{terms} - 1)}{r - 1}$

$= \dfrac{1(2^{n+1} - 1)}{1}$

$\Rightarrow T(n) = [2^n \cdot 2 - 1]$

$T(n) = O(2^n)$

Space complexity - $O(1)$ as recursive implementation doesn't store any values and calculates every value from scratch so as complexity of call is $O(1)$

$\therefore$ total space complexity $= O(1)$

Ans 3 Programs which have complexity:-

1. $n(\log n)$

```
for(i=1; i<=n; i=i*2)        // log n times
    for(j=1; j<=n; j++)      // n times
        int s=1;
```

$\Rightarrow O(n \log n)$

Alwin

ii) n^3

```
for(i=0; i<=n; i++)                    //n times
    for(j=0; j<=n; j++)                //n times
        for(k=0; k<=n; k++)            //n times
            cout << "N power 3 time complexity";
```

$\Rightarrow O(n^3)$

iii) log(log n)

```
for(int i=2; i<n; i=pow(i,10))
{
    cout << "Log Log n"<<endl;
}
```

$\Rightarrow \log(\log(n))$

Que 4 $T(n) = T(n/4) + T(n/2) + cn^2$

by neglecting lower order term $T(n/4)$

$T(n) = T(n/2) + cn^2$

$a = 1$, $b = 2$

$\Rightarrow c = \log_2 1$

$= 0$

$n^c = n^0 = 1 < cn^2$

$\therefore \Rightarrow T(n) = \Theta(n^2)$

Abhinav

Date.
Page No.

Ques 5 int fn(int n)

```
{
  for(int i=1; i<=n; i++)
    for(int j=1; j<n; j+=i)
      cout<<'Hi';
}
```

for $i=1$ $\quad j=1+2+3+4+5+6 \ldots\ldots n$

for $i=2$ $\quad j=1,3,5,7,\ldots\ldots n$

for $i=3$ $\quad j=1,4,7,\ldots\ldots n.$

$$T(n) = n+\frac{n}{2}+\frac{n}{3}+\frac{n}{4}+\ldots\ldots 1$$

$$= n\left(1+\frac{1}{2}+\frac{1}{4}+\ldots\ldots\frac{1}{n}\right)$$

$$= n\int_{1}^{n} 1/n$$

$$= O(n.\log n)$$

Ques 6 Time complexity :-> for(int i=2; i<=n; i=pow(i,k))
{ }
where k is a constant

First iteration $i=2$

Second " $i=2^k$

Third " $i=(2^k)^k = 2^{k^2}$

|

|

n^{th} iteration $i=2^{k^i}=n$

Ahinav

$n = 2^{k^i}$

applying log on both sides.

$\log n = \log_2 2^{k^i}$

$\log n = k^i \log_2 2$

$\log n = k^i$

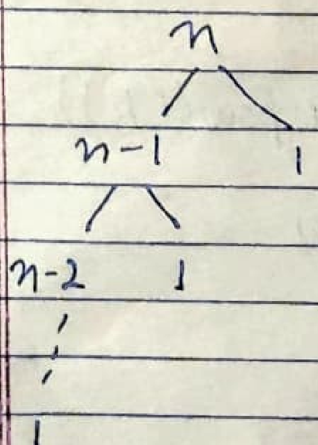
Taking log with base k.

$\log_k \log n = i$

$T(n) = \log_k \log(n)$

Sol 7 99% & 1% array part

$\therefore T(n) = T(n-1) + O(1)$



$(n) = (T(n-1) + T(n-2) + \ldots T(1) + O(1)) \times n$

$= n \times n$

$\therefore T(n) = O(n^2)$

lowest height = 2

highest height = n

$\because$ diff $= n-2$ $\quad n>1$

the given algo provides linear result

Ques 8 Arrange the following in increasing order of given growth rate.

Ans 8 a) $100 < \log.\log n < \log n < (\log n)^2 < \sqrt{n} < n < n\log n < \log(n!) < n^2 < 2^n < 4^n < 2^{2n}$

b) $1 < \log.\log n < \sqrt{\log n} < \log n < \log 2n < 2\log n < n < n\log n < 2n < m < \log(n!) < n^2 < n! < 2^{2n}$

c) $96 < \log_8 n < \log 2n < 5n < n\log_6 n < 6n\log_2 n < \log(n!) < 8n^2 < 7n^3 < n! < 8^{2n}$