Tutorial - 2.

Qu1 what is the time complexity of bellow a how.

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

i = 0, 1, 3, 6 10, let say k times
So gener form would be
$\frac{K(K+1)}{2}$

K^{th} term $n = \frac{K(K+1)}{2} = n$

$K^2 + K = 2n$

$K^2 = n = K = \sqrt{n}$

Time complexity $= O(\sqrt{n})$ m

Ans2 Write Recurrence relation for the recursion function that prints fibonacci series. solve the reccurence relation to get the time complexity of this program and why.

Recurrence relation

int fib(int n){
  int (n<=1). ⟺ O(1) = c.
    return n;
  return fid(n-1)+ fid(n-2) → T(n-1)+T(n-2)
}

Recurrence relation $T(n) = T(n-1) + T(n-2) + c$

Now $T(n-1) \simeq T(n-2)$

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

By Backward substitution

$T(n-1) = 2T(n-1-1)+c \Rightarrow 2T(n-2)+c$

$T(n) = 2[2T(n-2)+c]+c$

$= 4T(n-2)+3c$

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

$2T(n-3)+c$

$T(n) = 4T(n-2) + 3(c)$

$4(2T(n-3) + c) + 3c$

$T(n) = 8T(n-3) + 7c$

generalizing: $2^k T(n-k) + (2^k - 1)c.$

assume $n - k = 0 \Rightarrow n - k.$

$2^n T(0) + (2^n - 1)c$

$= 2^n + (2^n - 1)c$

$2^n + (2^n - 1)c$

$= 2^n$

Time complexity $= O(2^n).$

Space complexity

For fibonacci space required is ~~dl~~ directly ∝ to maximum depth of Recurssion tree, sice maximum depth is di ∝ to number of eleme

Ques 3 write a program which have complexity - n(logn)

(i)
```
for(i=1; i<=n; i++){
    for(j=1; j<=n; j=j*2)
    { sum = sum+i; }
}
```

(ii) n^3
```
for(i=0; i<n; i++){
  { for(j=0; j<n; j++)
      for(k=0; k<n; k++))
          sum = sum+k
  }
}
};
```

(iii) log n(log n)
```
for(i=1; i<n ; i=i*2)
  for(k=1 ; k=n; k=k*2)
      sum = sum+ j;
    }
  }
};
```

Page No.: Date: YOUVA

Ques 4 solve the Recurrence relation $T(n) = T(n/4) + T(n/2) + cn^2$

$T(n/4) = T(n/2)$

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

$a \geq 1$ and $b \geq 1$

By using master's method.

$T(n) = a\,T(n/b) + f(n)$

$c = \log_2 a \Rightarrow 1$

$f(n) > n^c \Rightarrow (n^2 > n^c)$

$T(n) = O(f(n))$

$O(n^2)$

Ques 5 What is the time complexity of following fun().

```
int fun(int n)
    for (int i=1; i<=n; i++) {
        for (int j=1 ; j<n ; j+=i ){
            Some O(1) task }}}
```

Sol

for $i=1 \rightarrow 1+2+3+ \dots (n+1) = n$

for $i=2 \rightarrow 1+3+5+ \dots n \Rightarrow n/2$

for $i=3 \rightarrow 1+4+7+ \dots n \Rightarrow n/3$

$n + \frac{n}{2} + \frac{n}{3} + \dots + 1$

$\Rightarrow n\left(1 + \frac{1}{2} + \frac{1}{3} + \dots \frac{1}{n}\right)$

Now we know

$n\left(1 + \frac{1}{2} + \dots \frac{1}{n}\right) \leq n\left(1 + \frac{1}{2} + \frac{1}{3} \dots \frac{1}{n}\right)$

$n\left(1 + \frac{1}{2} + \frac{1}{3} + \dots \frac{1}{n}\right) \leq n(1 + 0.3 \dots)$

$O(n \log n)$ Ans.

Ques 7 Write an recurrence relation when quick sort repeatedly divides the array to show the recurrence the time complexity the difference in height of both the extreme parts what do your understand by the analysis.

99 to 1 in quick sort.
when pivot is where from front or end always.

$T(n) = T(99/100) + T(n/100) + O(n)$.

$T(n)$

$T\left(\frac{99n}{100}\right)$ $\quad$ $T(n/100)$

$T\left(\frac{(99)^2}{(100)^2} \times n\right)$ $\quad$ $T\left(\frac{99n}{100}\right)$ $\quad$ $T\left(\frac{99n}{(100)^2}\right)$ $\quad$ $T\left(\frac{n}{100^2}\right)$

$$n = \left(\frac{99}{100}\right)^K$$

$$\log n = K \log \frac{99}{100}$$

$$K = \log n \frac{100}{99}$$

Time complexity = $n \log$.

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

(a) $n, n!$ $\log n$, $\log\log n$, root(n), $\log(n!)$, $n\log n$.
$2^n, 2^{2^n}, 4^n, n^2, 100$.

$100 < \log(\log n) < \log n < \log^2 n < \sqrt{n} < n < \log n! < n\log n <$
$\log^{2n} < n^2 < 2^n < 4^n < 2^{2^n} < n!$

Page No.:
Date:
YOUVA

Tutorial 3

Ans 1) Write linear search pseudocode to search an element in a sorted array with minimum comparisons.

```
Void linear Search (int A[], int n, int Key){
        int flag=0;
        for(int i=0; i<n; i++)
            if( A[i] == Key){
                flag=1;
                break;
        }}
    if( flag==0)
        cout <<"Not found";
    else
        cout << "found";
```

Ans2. Write pseudo code for iterative ... algorithm has been in

```
Iterative for(i=1 to n-1) {
        t = A[i]; j=i-1;
    while ( j>=0 && A[j]>t) {
            A[j+1] = A[j]
            j--;
        }
        A[j+1]=t;
    }
```

Recursion

```
    void insertionsort (int arr[], int n){
            if(n<=1)
                return;
```

M	T	W	T	F	S	S

Page No.:
Date:
YOUVA

Ques 3 $T(n) = 3T(n/2) + n^2$

$n^{\log_2 3} = n^{1.5}$

$n^{1.5} > n$

insertion sort (arr, n-1)

int last = arr [n-1], j=n-2;

while (j>0 && arr [j] > last)

arr [j+1] = arr [j]

j--;

arr [j+1] = last;

}

Insertion sort is called online sorting a input element per iteration and produces a partial solution without require access to offline algorithm

Ques 3 complexity of all sorting algorithm has been discussed

	Best	Average	Worst
Bubble sort	$O(n)$	$O(n^2)$	$O(n^2)$
Selection sort	$O(n^2)$	$O(n^2)$	$O(n^2)$
Insertion sort	$O(n)$	$O(n^2)$	$O(n^2)$
Count sort	$O(n+k)$	$O(n+k)$	$O(n+k)$
Quick sort	$O(n\log n)$	$O(n\log n)$	$O(n^2)$
Merge sort	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$
Heap sort	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$

Inplace stable instable.

Algorithm	Inplace	Stable	Online
Bubble	✓	✓	✗
Selection	✓	✗	✗
Insertion	✓	✓	✓
Count	✗	✓	✗
Merge	✗	✓	✗
Quick	✓	✗	✗
Heap	✓	✗	✗

Ques 5 Write pseudo code for binary search in linear search

Recursive = int binary (int arr[], int l, int r, int key)

```
{ if(r >= l)
    { int mid = l+(r-l)/2;
      if (arr[mid] == key)
            return mid;
      if (arr[mid] > key)
          return binary(arr, mid-1, key)
      return binary(arr, mid+1, r, key)
    }
  return -1;
}
```

```
Int binary search(int arr, int l, int r, int key)
  { while (l <= r){
        int m = l+(r-l)/2;
      if (arr[m] == key)
            return m;
    if (arr[m] < key)
        l = m+1;
    else
        r = m-1;
    }
    return -1;
  }
```

Ques 7 Find two indices such that A[i]+A[j]=k in minimum

```
void sum (int A[], int k, int n)
  { Sort (A, A+n)
      int i=0  j=n-1
   while (i<j)
      if (A[i] + A[j] == k)
          break;
    else if (A[i] + A[j] > k)
          j--;
   print (i,j);
```

Ans 8 which sorting best for practical uses? Explain
For practical uses, we it would be best for very large data. Further, time complexity of merge sort is same in all cases, that is $O(n(\log n))$.