P5) Sort a given set of n integer elements using merge sort method & compute its time complexity. Run the program for varied values of n > 5000, & record the time taken to sort. Plot a graph of the time taken versus n on graph sheet. The elements can be read from file or can be generated using random number generator. Demonstrate using Java how the divide & conquer method works along with its time complexity analysis: worst case, average case & best case.

```
import java.util.Random;
import java.util.Scanner;
public class lab5{
  public static void main(String[] args){
    int a[] = new int[10000];
Scanner in = new Scanner(System.in);
long start, end;
System.out.println(" -- merge sort --");
System.out.println("Enter number of elements:");
int n = in.nextInt();
```

```
Random rand = new Random();
for (int i=0; i<n; i++)
a[i] = rand.nextInt(10000);
System.out.println("array Elements in be
 sorted are:");
for( int i=0; i<n; i++)
System.out.print(a[i]+" ");
start = System.nanoTime();
mergesort (a,0,n-1);
end = System.nanoTime();
System.out.println("\n Sorted Elements are:");
for (int i=0; i<n; i++)
System.out.print(a[i] +" ");
System.out.println("\n\n Time taken to sort
 is :"+(end-start)+" ns");
System.out.println("***\n\n");
}

static void mergesort (int a[], int low, int high)
{
 int mid;
 if (low<high)
 {
    mid=(low+high)/2;
    mergesort (a,low,mid);
    mergesort (a,mid+1,high);
```

```
    merge(a,low, mid,high);
  }
}

Static void merge(int a[], int low, int mid, int high)
{
 int i,j,h,k;
 int temp[] = new int[100000];
 i=low; k=low; j=mid+1;
 while((i<=mid) && (j<=high))
 {
  if( a[i] < a[j])
  {
  temp[k] =a[i];
   i++; k++;
  }
  else
  {
  temp[k] = a[j];
   j++; k++;
  }
 }
while(i<=mid)
{
  temp[k] = a[i];
   i++; k++;
}
```

```
while (j<=high)
{
  temp[k] = a[j];
  j++; k++;
}
for( k=low; k<=high ; k++)
a[k] = temp[k];
}
}
```

Output:

```
-- merge sort --
Enter the number of Elements :9
Array Elements to be sorted are:
159 210 926 342 886 12 379 423 72
The sorted array:
12 72 159 210 342 379 423 886 926
The time taken to sort the array is : 140000ns
* * *
```

Tracing:

i=	0	1	2	3	4	5	6	7	8
	159	210	926	342	886	12	379	423	72

mergesort(a, 0, 8)

low=0 high=8 0<8 mid=4

mergesort(a,0,4) | mergesort(a,5,8) merge(a,0,4,8)

low=0; high=4; 0<4; mid=2 | l=5; h=8; 5<8; mid=6

mergesort(a,0,2) ms(a,3,4) | ms(a,5,6) ms(a,7,8)

merge(a,0,2,4) | merge(a,5,6,8)

l=0; high=2; 0<2; mid=1 | l=3; h=4; 3<4 mid=3 | l=5, h=6, 5<6 mid=5 | l=7; h=8; 7<8 mid=7

ms(a,0,1) ms(a,2,2) ms(a,3,3) ms(a,4,4) ms(a,5,5) ms(a,6,6) ms(a,7,7) merge(a,7,7,8)

2≠2 3≠3 4≠4 merge(a,5,5,6)

merge(a,0,1,2) merge(a,3,3,4)

l=0; high=1; 0<1

mid=0

ms(a,0,0) ms(a,1,1)

merge(a,0,0,1)

0<0

1<1

1) merge(a,0,0,1) a: | 159 | 210 | ...

low=0 mid=0 high=1 i j

i=0 j=1 k=0

temp: | 159 | 210 |

0 1

↳ i<=0 && j<=1

0<=0 && 1<=1 ⇒ true.

if a[0]<a[1]

159<210 ⇒ true

temp[0] = a[0] ⇒ temp[0] = 159;

• k++; i++; k=1 & i=1

k=1; i=1; j=1

↳ while(i<=0 && 1<=1) ⇒ fail

↳ while(1<=0) ⇒ fail

↳ while(1<=1) ⇒ true

temp[1] = a[1] ⇒ temp[1] = 210;

j++; k++ ⇒ j=2; k=2

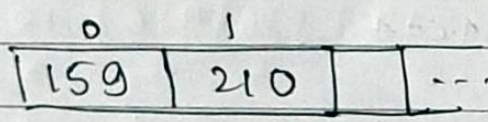
while(2<=1) ⇒ fail

• For(k=0; k<=1; k++)

k=0 ⇒ a[0] = temp[0] ⇒ a[0] = 159 k++

k=1 ⇒ a[1] = temp[1] ⇒ a[1] = 210 k++

k=2 2<=1



ii) merge (a, 0, 1, 2) a: 159 | 210 | 926 | ...

0 1 2

i↑ j↑

i=0; j=2; k=0

while(0<=1 && 2<=2)

↳ a[0] < a[2]

159 < 926 ⇒ temp[0] = 159

i++; k++ ⇒ i=1, k=1

wh(1<=1 && 2<=2)

↳ a[1] < a[2]

210 < 926 ⇒ temp[1] = 210 i=2, k=2

2<=1

↳ while (j<=high)

while(2<=2)

↳ temp[2] = a[2] = 926

temp: | 159 | 210 | 926 | = a

0 1 2

copy temp to a from 0 to 2

iii) merge(a,3,3,4)

i=3 , k=3 , j=4

a | | | | 342 | 886 | ...

0 1 2 3 4

i↑ j↑

Wh(3<=3 && 4<=4)

↳ a[3] < a[4]

342 < 886 ⇒ temp[3] = a[3] = 342

i++; k++ ⇒ i=4, k=4

↳ 4<=3 && 4<=4

↳ 4<=3

↳(4<=4)

↳ [illegible] = a[4] = 886

j++; k++ ⇒ j=5 , k=5

5<=4

temp: | 342 | 886 |

3 4

copy temp to a from 3 4

a: | 159 | 210 | 926 | 342 | 886 | ...

0 1 2 3 4

159 210 342 886

iv) merge(a,0,2,4)

a	159	210	926	342	886
	0	1	2	3	4
	i↑		i↑	j↑	j↑

i=0; k=0; j=3

↳ 0<=2 && 3<=4

a[0]<a[3] **k=0**

159 < 342 ⇒ temp[0] = 159 i++, k++

↳ 1<=2 && 3<=4

a[1]<a[3] **k=1**

210 < 342 ⇒ temp[1] = 210 i++, k++

↳ 2<=2 && 3<=4

a[2]<a[3] **k=2**

926 < 342 ⇒ temp[2] = 342, j++, k++

↳ 2<=2 && 4<=4

a[2]<a[4] **k=3**

926 < 886 ⇒ temp[3] = 886, j++, k++

↳ 2<=2 && 5!=4 **k=4**

↳ 2<=2

temp[4] = a[2] = 926. ⇒ i++ k++

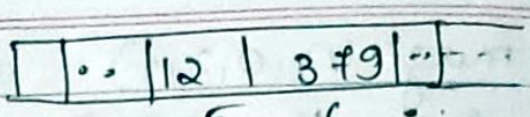
2!=2 **k=5**

temp:

159	210	342	886	926
0	1	2	3	4

copy temp to a from 0 to 4.

v) merge(a, 5, 5, 6) a:



p=5 k=5 q=6 (i↑ j↑)

↳ 5<=5 && 6<=6

a[5] < a[6]

12 < 379 ⇒ temp[5] = 12 ⇒ p++, k++

p = 6, k = 6

↳ 6 <= 5

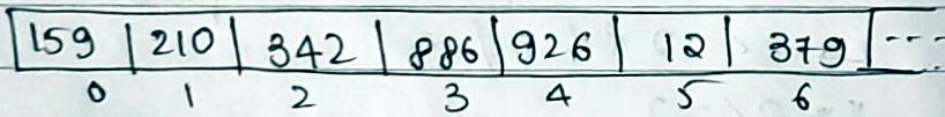
↳ 6 <= 6

temp[6] = a[6] = 379

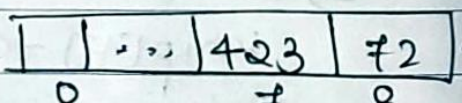
temp:

Copy temp to a from 5 to 6.

a:



vi) merge(a, 7, 7, 8) a:



i↑ j↑

p=7, k=7, q=8

↳ 7<=7 & 8<=8

a[7] < a[8]

423 ≮ 72 ⇒ temp[7] = a[8] ⇒ k++, j++

k = 8 q = 9

↳ 7 <= 7 & 9 <= 8

↳ 7 <= 7

temp[8] = a[7] = 423

temp:

72	423
7	8

Copy temp to a from 7 to 8

a:

159	210	342	886	926	12	379	72	423
0	1	2	3	4	5	6	7	8

vii) merge(a, 5, 6, 8) a

12	379	72	423
5	6	7	8

p=5, j=7, k=5

↳ 5<=6 && 7<=8

a[5] < a[7]

12 < 72 ⇒ temp[5] = 12 ⇒ i++, k++

↳ 6<=6 && 7<=8 k=6 & p=6

a[6] < a[7]

379 < 72 ⇒ temp[6] = a[7] = 72 ⇒ j++, k++

↳ 6<=6 && 8<=8

a[6] < a[8]

379 < 423 ⇒ temp[7] = a[6] = 379 ⇒ j++, i++

↳ 7 !<= 6

↳ 8<=8

temp[8] = a[8] = 423

temp:

12	72	379	423
5	6	7	8

Copy temp to a from 5 to 8

a:

159	210	342	886	926	12	72	379	423
0	1	2	3	4	5	6	7	8

viii) merge(a,0,4,8)

i=0 j=5 k=0

↳ 0<=4 && 5<=8

a[0]<a[5] -

159 < 12 ⇒ temp[0]=a[5]=12 ⇒ k++, j++

↳ 0<=4 && 6<=8

a[0]<a[6]

159 < 72 ⇒ temp[1]=a[6]=72 ⇒ k++, j++

↳ 0<=4, && 7<=8

a[0]<a[7]

159 < 379 ⇒ temp[2]=a[0]=159 ⇒ k++, i++

↳ 1<=4 && 7<=8

a[1]<a[7]

210 < 379 ⇒ temp[3]=a[1]=210 ⇒ k++, i++

↳ 2<=4 && 7<=8

a[2]<a[7]

342 < 379 ⇒ temp[4]=a[2]=342 ⇒ k++, i++

↳ 3<=4 && 7<=8

a[3]<a[7]

886 < 379 ⇒ temp[5]=a[7]=379 ⇒ k++, j++

↳ 3<=4 && 8<=8

a[3]<a[8]

886 < 423 ⇒ temp[6]=a[8]=423 ⇒ k++, j++

↳ 3<=4 && 9<=8

↳ 3<=4

temp[7]=a[3]=886 ⇒ k++, j++

↳ i<c=4

temp[8] = a[4] = 926 → i++, k++

↳ 5<4=4

'temp':

12	72	159	210	342	379	423	886	926
0	1	2	3	4	5	6	7	8

Copy temp to a from 0 to 8