1. WAP to implement linear search algorithm repeat for different value of N, number of element in list to be searched and plot A graph of time taken versus N.

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <time.h>
int linear(int a[], int n, int key)
{
int i;
for (i=0; i<n; i++)
{
if (a[i] == key)
{
return (i);
}
}
return (-1);
}
void main()
{
int *a, i, n, key, pos;
clock_t start, end;
```

```
clrscr();
printf("enter the size of an array\n");
scanf("%d", &n);
a = (int*) calloc (n, sizeof(int));
printf("elements are:\n");
for (i=0; i<n; i++)
{
a[i] = rand();
printf("%d\t", a[i]);
}
printf("\n enter key of search");
scanf("%d", &key);
start = clock();
delay(116);
pos = linear(a, n, key);
end = clock();
if (pos == -1)
{
printf("key not found");
}
else
{
printf("%d is at pos %d\n", key, pos+1);
}
printf("time taken=%f\n", (end-start/CLK_TCK));
getch();}
```

output:

enter the size of an array.

5

elements are:

346 30 130 175 217.

enter key of search.

346.

346 is at pos 1.

time taken = 10.375.

2. WAP to implement binary search algorithm repeat for different values of N, number of element in list to be searched and plot A graph of time taken versus N,

→
```
#include<stdio.h>
#include<conio.h>
#include<time.h>
void main()
{
int *a, flag, i, n, item, result, j, temp;
clock_t start, end;
clrscr();
printf("enter the size of the array");
scanf("%d", &n);
a=(int*) calloc (n, sizeof(int));
printf("elements are:");
for (i=0; i<n; i++)
{
a[i]=rand();
printf("%d\t", a[i]);
}
for (i=0; i<n; i++)
for (j=i+1; j<n; j++)
{
```

```
if (a[i] > a[j])
{
temp = a[i];
a[i] = a[j];
a[j] = temp;
}
}
printf (" enter the item to be searched.");
scanf (" %d ", &item);
start = clock ();
delay (1000);
flag = bsearch (a, item, 0, n-1);
end = clock ();
if (flag == -1)
{
printf (" the item %d is not found ", item);
}
else
{
printf (" the item %d found in pos %d", item,
flag + 2);
printf (" time taken : %f ", ((end - start)/(CLK_TCK)));
}
getch ();
}
```

```
int bsearch (int a[], int item, int first,
int last).
{
int middle;
if (first > last)
return (-1);
else
{
middle = (first + last) / 2;
if (item < a[middle])
return (bsearch (a, item, first, middle - 1));
else if (item > a[middle])
return (bsearch (a, item, middle + 1, last));
else
return (middle);
}
}
```

output:

enter the size of the array 5

elements are : 346 130 10982 1090 11656

enter the item to be searched 346

the item 346 found in pos 3

time taken : 0.104890

3. WAP to solve towers of hanoi problem and execute it for different number of disks.

→
```
#include<stdio.h>
#include<conio.h>
#include<process.h>
void tower(int n, char a, char b, char c)
{
if(n==1)
{
printf("move disc %d from %c to %c \n",
n, a, c);
return;
}
tower(n-1, a, b, c);
printf("move disc %d from %c to %c \n", n, a, c);
tower(n-1, b, c, a);
}
void main()
{
int n;
clrscr();
printf("enter the number of disc \n");
scanf("%d", &n);
tower(n, 'a', 'b', 'c');
```

```
getch();
}
```

output:

```
enter the number of disc
3
move disc 1 from a to c
move disc 2 from a to c
move disc 1 from b to a
move disc 3 from a to c
move disc 1 from b to a
move disc 2 from b to a
move disc 1 from c to b.
```

4. WAP to sort given set of no. using selection sort algorithm. Repeat for different value of N, number of element in list to be searched and plot A graph of time taken vs N. the element can be read from a file or can be generated using random number generator.

->
```
#include<stdio.h>
void main()
{
int a[100], num, min, i, j, temp;
clrscr();
printf("\n please enter the total elements:");
scanf("%d", &num);
printf("\n please enter the array elements");
for(i=0; i<num; i++)
scanf("%d", &a[i]);
for(i=0; i<num-1; i++)
{
min=i;
for(j=i+1; j<num; j++)
{
if(a[min]>a[j])
{
min=j;
```

```
}
}
if (min != i)
{
temp = a[i];
a[i] = a[min];
a[min] = temp;
}
}
printf("\n result:");
for (i=0; i<num; i++)
{
printf("%d\n", a[i]);
}
printf("\n");
getch();
}
```

output:

please enter the total elements : 5

please enter the array elements 5

4

3

2

2

result : 2 2 3 4 5

5. WAP to find value of A using brute force based algorithm and divide and conquer based algorithm.

→.
```
#include<stdio.h>
int power(int n1, int n2);
void main()
{
int base, a, result;
clrscr();
printf("enter the base number");
scanf("%d", &base);
printf("enter power number(positive number):");
scanf("%d", &a);
result = power(base, a);
printf("%d ^ %d = %d", base, a, result);
getch();
}
int power(int base, int a)
{
if (a!=0)
return(base * power(base, a-1));
else
return 1;
}
```

Output:

enter the base number : 2

enter the power number (posidive number): 5

2^5 = 32.