S.No: 1	Exp. Name: ***Write a C program to Sort the given elements in Ascending order using Bubble Sort***	**Date: 2023-02-27**

Aim:

Write a program to `sort` (`Ascending order`) the given elements using `bubble sort technique`.

At the time of execution, the program should print the message on the console as:

```
Enter value of n :
```

For example, if the user gives the **input** as:

```
Enter value of n : 3
```

Next, the program should print the messages one by one on the console as:

```
Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :
```

if the user gives the **input** as:

```
Enter element for a[0] : 22
Enter element for a[1] : 33
Enter element for a[2] : 12
```

then the program should **print** the result as:

```
Before sorting the elements in the array are
Value of a[0] = 22
Value of a[1] = 33
Value of a[2] = 12
After sorting the elements in the array are
Value of a[0] = 12
Value of a[1] = 22
Value of a[2] = 33
```

Note: Do use the **printf()** function with a **newline** character (`\n`).

Source Code:

Program504.c

```
#include<stdio.h>
int main()
{
        int n,ns,ne,temp,arr[100];
        printf("Enter value of n : ");
        scanf("%d",&n);
        for (int i=0;i<n;i++){
                printf("Enter element for a[%d] : ",i);
                scanf("%d",&arr[i]);
        }
        printf("Before sorting the elements in the array are\n");
        for(int i=0;i<n;i++){
                printf("Value of a[%d] = %d\n",i,arr[i]);
        }
        printf("After sorting the elements in the array are\n");
        for(int i=0;i<n-1;i++){
                for(int j=0;j<n-1-i;j++){
                        if(arr[j]>arr[j+1]){
                                temp=arr[j];
                                arr[j]=arr[j+1];
                                arr[j+1]=temp;









                        }
                }
        }
        for(int i=0;i<n;i++){
                printf("Value of a[%d] = %d\n",i,arr[i]);
        }

}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
5
Enter element for a[0] :
2
Enter element for a[1] :
7
Enter element for a[2] :
6
Enter element for a[3] :
4
Enter element for a[4] :
1

Before sorting the elements in the array are
Value of a[0] = 2
Value of a[1] = 7
Value of a[2] = 6
Value of a[3] = 4
Value of a[4] = 1
After sorting the elements in the array are
Value of a[0] = 1
Value of a[1] = 2
Value of a[2] = 4
Value of a[3] = 6
Value of a[4] = 7

Test Case - 2
User Output
Enter value of n :
4
Enter element for a[0] :
28
Enter element for a[1] :
34
Enter element for a[2] :
26
Enter element for a[3] :
29
Before sorting the elements in the array are
Value of a[0] = 28
Value of a[1] = 34
Value of a[2] = 26
Value of a[3] = 29
After sorting the elements in the array are
Value of a[0] = 26
Value of a[1] = 28
Value of a[2] = 29
Value of a[3] = 34

Test Case - 3
User Output
Enter value of n :
8
Enter element for a[0] :
7
Enter element for a[1] :
3
Enter element for a[2] :
9
Enter element for a[3] :
2

Enter element for a[4] :
5
Enter element for a[5] :
4
Enter element for a[6] :
6
Enter element for a[7] :
1
Before sorting the elements in the array are
Value of a[0] = 7
Value of a[1] = 3
Value of a[2] = 9
Value of a[3] = 2
Value of a[4] = 5
Value of a[5] = 4
Value of a[6] = 6
Value of a[7] = 1
After sorting the elements in the array are
Value of a[0] = 1
Value of a[1] = 2
Value of a[2] = 3
Value of a[3] = 4
Value of a[4] = 5
Value of a[5] = 6
Value of a[6] = 7
Value of a[7] = 9

Test Case - 4
User Output
Enter value of n :
4
Enter element for a[0] :
-23
Enter element for a[1] :
-14
Enter element for a[2] :
-56
Enter element for a[3] :
-35
Before sorting the elements in the array are
Value of a[0] = -23
Value of a[1] = -14
Value of a[2] = -56
Value of a[3] = -35
After sorting the elements in the array are
Value of a[0] = -56
Value of a[1] = -35
Value of a[2] = -23
Value of a[3] = -14

Test Case - 5
User Output
Enter value of n :
5
Enter element for a[0] :
28
Enter element for a[1] :
45
Enter element for a[2] :
-1
Enter element for a[3] :
-5
Enter element for a[4] :
2
Before sorting the elements in the array are
Value of a[0] = 28
Value of a[1] = 45
Value of a[2] = -1
Value of a[3] = -5
Value of a[4] = 2
After sorting the elements in the array are
Value of a[0] = -5
Value of a[1] = -1
Value of a[2] = 2
Value of a[3] = 28
Value of a[4] = 45

S.No: 2	Exp. Name: ***Write a C program to Search an element using Binary Search process***	**Date: 2023-03-01**

Aim:

Write a program to `search` a key element in the given array of elements using `binary search`.

At the time of execution, the program should print the message on the console as:

```
Enter value of n :
```

For example, if the user gives the **input** as:

```
Enter value of n : 3
```

Next, the program should print the messages one by one on the console as:

```
Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :
```

if the user gives the **input** as:

```
Enter element for a[0] : 89
Enter element for a[1] : 33
Enter element for a[2] : 56
```

Next, the program should print the message on the console as:

```
Enter key element :
```

if the user gives the **input** as:

```
Enter key element : 56
```

then the program should **print** the result as:

```
After sorting the elements in the array are
Value of a[0] = 33
Value of a[1] = 56
Value of a[2] = 89
The key element 56 is found at the position 1
```

Similarly if the key element is given as **25** for the above one dimensional array elements then the program should print the output as "**The Key element 25 is not found in the array**".

Note: Do use the **printf()** function with a **newline** character (`\n`) at the end.

Source Code:

```
Program510.c
```

```
#include<stdio.h>
int binarysearch(int arr[],int n,int key){
        int low = 0,high = n-1;
        while(low<=high){
                int mid = (low+high)/2;
                if(arr[mid]==key){
                        return mid;
                }
                else if(arr[mid<key]){
                        low=mid+1;

                }
                else{
                        high=mid-1;
                }
        }
                return -1;
}
int main(){
        int n,i,key;
        printf("Enter value of n : ");
        scanf("%d",&n);
        int arr[n];
        for(i=0;i<n;i++){
                printf("Enter element for a[%d] : ",i);
                scanf("%d",&arr[i]);
        }
        printf("Enter key element : ");
        scanf("%d",&key);
        for(int i=0;i<n-1;i++){
                int j,temp;
                for(j=i+1;j<n;j++){
                if(arr[i]>arr[j]){
                        temp =arr[i];
                        arr[i]=arr[j];
                        arr[j]=temp;
                }
                }
        }
        printf("After sorting the elements in the array are\n");
        for(int i=0;i<n;i++){
                printf("Value of a[%d] = %d\n",i,arr[i]);
        }
        int result = binarysearch(arr,n,key);
        if(result == -1){
                printf("The Key element %d is not found in the array\n",key);
                }
                else{
                        printf("The key element %d is found at the position
%d\n",key,result);
                }
                return 0;
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
5
Enter element for a[0] :
4
Enter element for a[1] :
8
Enter element for a[2] :
6
Enter element for a[3] :
2
Enter element for a[4] :
1
Enter key element :
8
After sorting the elements in the array are
Value of a[0] = 1
Value of a[1] = 2
Value of a[2] = 4
Value of a[3] = 6
Value of a[4] = 8
The key element 8 is found at the position 4

Test Case - 2
User Output
Enter value of n :
7
Enter element for a[0] :
56
Enter element for a[1] :
89
Enter element for a[2] :
63
Enter element for a[3] :
215
Enter element for a[4] :
325
Enter element for a[5] :
156
Enter element for a[6] :
256
Enter key element :
458
After sorting the elements in the array are
Value of a[0] = 56

```
Value of a[1] = 63
Value of a[2] = 89
Value of a[3] = 156
Value of a[4] = 215
Value of a[5] = 256
Value of a[6] = 325
The Key element 458 is not found in the array
```

S.No: 3	Exp. Name: ***Write a C program to Sort given elements using Insertion sort***	**Date: 2023-02-28**

Aim:

Write a program to `sort` (`Ascending order`) the given elements using `insertion sort technique`.

At the time of execution, the program should print the message on the console as:

```
Enter value of n :
```

For example, if the user gives the **input** as:

```
Enter value of n : 3
```

Next, the program should print the messages one by one on the console as:

```
Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :
```

if the user gives the **input** as:

```
Enter element for a[0] : 22
Enter element for a[1] : 33
Enter element for a[2] : 12
```

then the program should **print** the result as:

```
Before sorting the elements in the array are
Value of a[0] = 22
Value of a[1] = 33
Value of a[2] = 12
After sorting the elements in the array are
Value of a[0] = 12
Value of a[1] = 22
Value of a[2] = 33
```

Note: Do use the **printf()** function with a **newline** character (`\n`).

Source Code:

Program505.c

```c
#include<stdio.h>
int main(){
        int n,i,j,temp;
        printf("Enter value of n : ");
        scanf("%d",&n);
        int arr[n];
        for(i=0;i<n;i++){
                printf("Enter element for a[%d] : ",i);
                scanf("%d",&arr[i]);
                }
printf("Before sorting the elements in the array are\n");
for(i=0;i<n;i++){
        printf("Value of a[%d] = %d\n",i,arr[i]);
        }
        for(i=1;i<n;i++){
                temp=arr[i];
                j=i-1;
                while(j>=0 && arr[j]>temp){
                        arr[j+1]=arr[j];
                        j=j-1;
                }
                arr[j+1]=temp;
        }
        printf("After sorting the elements in the array are\n");
        for(i=0;i<n;i++){
                printf("Value of a[%d] = %d\n",i,arr[i]);

        }
        return 0;
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
5
Enter element for a[0] :
7
Enter element for a[1] :
33
Enter element for a[2] :
12
Enter element for a[3] :
56
Enter element for a[4] :
9
Before sorting the elements in the array are
Value of a[0] = 7
Value of a[1] = 33
Value of a[2] = 12

```
Value of a[3] = 56
Value of a[4] = 9
After sorting the elements in the array are
Value of a[0] = 7
Value of a[1] = 9
Value of a[2] = 12
Value of a[3] = 33
Value of a[4] = 56
```

S.No: 4	Exp. Name: ***Write a C program to Search an element using Linear Search process***	**Date: 2023-02-28**

Aim:

Write a program to `search` a key element with in the given array of elements using `linear search` process.

At the time of execution, the program should print the message on the console as:

```
Enter value of n :
```

For example, if the user gives the **input** as:

```
Enter value of n : 3
```

Next, the program should print the messages one by one on the console as:

```
Enter element for a[0] :
Enter element for a[1] :
Enter element for a[2] :
```

if the user gives the **input** as:

```
Enter element for a[0] : 89
Enter element for a[1] : 33
Enter element for a[2] : 56
```

Next, the program should print the message on the console as:

```
Enter key element :
```

if the user gives the **input** as:

```
Enter key element : 56
```

then the program should **print** the result as:

```
The key element 56 is found at the position 2
```

Similarly if the key element is given as **25** for the above one dimensional array elements then the program should print the output as "**The Key element 25 is not found in the array**".

Note: Do use the **printf()** function with a **newline** character (`\n`) at the end.

Source Code:

```
Program509.c
```

```c
#include<stdio.h>
int main(){
        int n,i,key,flag=0;
        printf("Enter value of n : ");
        scanf("%d",&n);
        int arr[n];
        for(i =0;i<n;i++){
                printf("Enter element for a[%d] : ",i);
                scanf("%d",&arr[i]);
        }
        printf("Enter key element : ");
        scanf("%d",&key);
        for(i=0;i<n;i++){
                if(arr[i]==key){
                        flag=1;
                        printf("The key element %d is found at the position %d",key,i);
                        printf("\n");
                }
        }
        if(flag !=1){
                printf("The key element %d is not found in the array",key);
                printf("\n");
        }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter value of n :
5
Enter element for a[0] :
45
Enter element for a[1] :
67
Enter element for a[2] :
35
Enter element for a[3] :
28
Enter element for a[4] :
16
Enter key element :
28
The key element 28 is found at the position 3

Test Case - 2
User Output
Enter value of n :

ID: 21SCSE1011615
E2UC403B-2023-Section-21
Galgotias University Greater Noida

5
Enter element for a[0] :
2
Enter element for a[1] :
7
Enter element for a[2] :
5
Enter element for a[3] :
1
Enter element for a[4] :
4
Enter key element :
2
The key element 2 is found at the position 0

Test Case - 3
User Output
Enter value of n :
4
Enter element for a[0] :
452
Enter element for a[1] :
356
Enter element for a[2] :
754
Enter element for a[3] :
127
Enter key element :
127
The key element 127 is found at the position 3

Test Case - 4
User Output
Enter value of n :
3
Enter element for a[0] :
5
Enter element for a[1] :
7
Enter element for a[2] :
3
Enter key element :
4
The key element 4 is not found in the array

Test Case - 5

User Output
Enter value of n :
3
Enter element for a[0] :
11
Enter element for a[1] :
45
Enter element for a[2] :
37
Enter key element :
25
The key element 25 is not found in the array

S.No: 5	Exp. Name: ***Write a C program to Sort given elements using Quick sort***	**Date: 2023-03-01**

Aim:

Write a program to `sort` (`Ascending order`) the given elements using `quick sort` technique.

Note: Pick the last element as pivot. You will not be awarded marks if you do not follow this instruction.

At the time of execution, the program should print the message on the console as:

```
Enter array size :
```

For example, if the user gives the **input** as:

```
Enter array size : 5
```

Next, the program should print the following message on the console as:

```
Enter 5 elements :
```

if the user gives the **input** as:

```
Enter 5 elements : 34 67 12 45 22
```

then the program should **print** the result as:

```
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67
```

Note: Do use the **printf()** function with a **newline** character (`\n`).

Source Code:

QuickSortMain.c

```
#include <stdio.h>
#include "QuickSortFunctions.c"
void main() {
        int arr[15], i, n;
        printf("Enter array size : ");
        scanf("%d", &n);
        printf("Enter %d elements : ", n);
        for (i = 0; i < n; i++) {
                scanf("%d", &arr[i]);
        }
        printf("Before sorting the elements are : ");
        display(arr, n);
        quickSort(arr, 0, n - 1);
        printf("After sorting the elements are : ");
        display(arr, n);
}
```

QuickSortFunctions.c

```
void display(int arr[15], int n) {
   int i;
   for(i=0;i<n;i++){
        printf("%d ",arr[i]);
   }
        printf("\n");

}
int partition(int arr[15], int lb, int ub) {
        int pivot=arr[ub];
        int i=(lb-1);
        for(int j=lb;j<ub;j++){
                if(arr[j]<=pivot){
                        i++;
                        int temp=arr[i];
                        arr[i]=arr[j];
                        arr[j]=temp;
                }
        }
        int temp=arr[i+1];
        arr[i+1]=arr[ub];
        arr[ub]=temp;
        return (i+1);
}
void quickSort(int arr[15], int low, int high) {
   if(low<high){
        int pi=partition(arr,low,high);
            quickSort(arr,low,pi-1);
        quickSort(arr,pi+1,high);
   }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter array size :
5
Enter 5 elements :
34 67 12 45 22
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Test Case - 2
User Output
Enter array size :
8
Enter 8 elements :
77 55 22 44 99 33 11 66

Before sorting the elements are : 77 55 22 44 99 33 11 66
After sorting the elements are : 11 22 33 44 55 66 77 99

Test Case - 3
User Output
Enter array size :
5
Enter 5 elements :
-32 -45 -67 -46 -14
Before sorting the elements are : -32 -45 -67 -46 -14
After sorting the elements are : -67 -46 -45 -32 -14

S.No: 6	Exp. Name: ***Write a C program to sort the given elements using Heap sort***	**Date: 2023-03-01**

Aim:

Write a program to sort (ascending order) the given elements using heap sort technique.

Note: Do use the **printf()** function with a **newline** character (\n).

Source Code:

HeapSortMain.c

```
#include <stdio.h>
#include "HeapSortFunctions.c"
void main() {
        int arr[15], i, n;
        printf("Enter array size : ");
        scanf("%d", &n);
        printf("Enter %d elements : ", n);
        for (i = 0; i < n; i++) {
                scanf("%d", &arr[i]);
        }
        printf("Before sorting the elements are : ");
        display(arr, n);
        heapsort(arr,n);
        printf("After sorting the elements are : ");
        display(arr, n);
}
```

HeapSortFunctions.c

ID: 21SCSE1011615
E2UC403B-2023-Section-21
Galgotias University Greater Noida

```c
#include<stdio.h>
void swap(int *a,int *b){
        int temp = *a;
        *a = *b;
        *b = temp;

}
void heapify(int arr[],int n,int i){
        int large = i;
        int left = 2*i+1;
        int right = 2*i+2;

        if(left<n && arr[left] > arr[large]){
        large = left;
    }
        if(right<n && arr[right]>arr[large]){
                large = right;
        }
        if(large!=i){
                swap(&arr[i], &arr[large]);
                heapify(arr,n,large);
        }

}
void heapsort(int arr[],int n){
                for(int i=n/2-1;i>=0;i--){
                        heapify(arr,n,i);}
                        for(int i = n-1;i>=0;i--)
                        {
                                swap(&arr[0],&arr[i]);
                                heapify(arr,i,0);
                        }
                }

void display(int arr[],int n){
                for(int i=0;i<n;++i){
                        printf("%d ",arr[i]);}
                        printf("\n");

}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter array size :
5
Enter 5 elements :
23 54 22 44 12
Before sorting the elements are : 23 54 22 44 12
After sorting the elements are : 12 22 23 44 54

ID: 21SCSE1011615

Test Case - 2
User Output
Enter array size :
6
Enter 6 elements :
12 65 23 98 35 98
Before sorting the elements are : 12 65 23 98 35 98
After sorting the elements are : 12 23 35 65 98 98

Test Case - 3
User Output
Enter array size :
4
Enter 4 elements :
-23 -45 -12 -36
Before sorting the elements are : -23 -45 -12 -36
After sorting the elements are : -45 -36 -23 -12

Test Case - 4
User Output
Enter array size :
6
Enter 6 elements :
1 -3 8 -4 -2 5
Before sorting the elements are : 1 -3 8 -4 -2 5
After sorting the elements are : -4 -3 -2 1 5 8

S.No: 7	Exp. Name: ***Write a C program to Sort given elements using Merge sort***	Date: 2023-03-02

Aim:

Write a program to `sort` (`Ascending order`) the given elements using `merge sort` technique.

At the time of execution, the program should print the message on the console as:

```
Enter array size :
```

For example, if the user gives the **input** as:

```
Enter array size : 5
```

Next, the program should print the following message on the console as:

```
Enter 5 elements :
```

if the user gives the **input** as:

```
Enter 5 elements : 34 67 12 45 22
```

then the program should **print** the result as:

```
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67
```

Note: Do use the **printf()** function with a **newline** character (`\n`).

Source Code:

MergeSortMain.c

```
#include <stdio.h>
#include "MergeSortFunctions.c"
void main() {
        int arr[15], i, n;
        printf("Enter array size : ");
        scanf("%d", &n);
        printf("Enter %d elements : ", n);
        for (i = 0; i < n; i++) {
                scanf("%d", &arr[i]);
        }
        printf("Before sorting the elements are : ");
        display(arr, n);
        splitAndMerge(arr, 0, n - 1);
        printf("After sorting the elements are : ");
        display(arr, n);
}
```

MergeSortFunctions.c

```
void display(int arr[15],int n){
        for(int i=0; i<n;i++){
                printf("%d ",arr[i]);
        }
        printf("\n");
}
void merge(int arr[15],int low,int mid,int high){
        int i=low,j = mid+1,B[100],k=low;
        while(i<=mid && j<=high){
                if(arr[i]< arr[j]){
                        B[k] = arr[i];
                        k++ ; i++;
                }
                else{
                        B[k]=arr[j];
                        j++; k++;
                }
                }
                while(i<=mid){
                        B[k]=arr[i];
                        i++; k++;
                }
                while(j<=high){
                        B[k]=arr[j];
                        j++; k++;
                }
                for(int i=low; i<=high;i++){
                        arr[i]=B[i];
                }
        }
void splitAndMerge(int arr[15],int low,int high){
        if(low < high){
                int mid = (low+high)/2;
                splitAndMerge(arr,low,mid);
                splitAndMerge(arr,mid+1,high);
                merge(arr,low,mid,high);
   }
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter array size :
5
Enter 5 elements :
34 67 12 45 22
Before sorting the elements are : 34 67 12 45 22
After sorting the elements are : 12 22 34 45 67

Test Case - 2

User Output
Enter array size :
8
Enter 8 elements :
77 55 22 44 99 33 11 66
Before sorting the elements are : 77 55 22 44 99 33 11 66
After sorting the elements are : 11 22 33 44 55 66 77 99

Test Case - 3
User Output
Enter array size :
5
Enter 5 elements :
-32 -45 -67 -46 -14
Before sorting the elements are : -32 -45 -67 -46 -14
After sorting the elements are : -67 -46 -45 -32 -14

S.No: 10	Exp. Name: ***Fractional Knapsack Problem***	Date: 2023-03-01

Aim:

The below program has a method void knapsack(). Which takes four parameters **number of objects**, the **weight of each object**, the **profit** corresponding to each one and the **capacity of the knapsack**. Write a program using a fractional knapsack algorithm to get the maximum profit.

Print the output as follows:

```
Sample Input and Output:
Enter the no. of objects: 6
Enter the weights and profits of each object:
1 2
4 5
8 9
4 6
5 2
3 5
Enter the capacity of knapsack:10
Maximum profit is:- 15.500000
```

Source Code:

knapsack.c

ID: 21SCSE1011615

```
# include<stdio.h>
void knapsack(int n, float weight[], float profit[], float capacity) {
  // write your code here
  float x[20],tp=0;
  int i,j,u;
  u=capacity;

  for(i=0;i<n;i++){
        x[i]=0.0;
  }
        for(i=0;i<n;i++){
                if(weight[i]>u){
                        break ;
                }
                else{
                        x[i]=1.0;
                        tp =tp+ profit[i];
                        u =u-weight[i];
                }
        }
        if(i<n){
                x[i] = u/weight[i];
        }
                tp =tp+(x[i] * profit[i]);
                printf("Maximum profit is:- %f\n", tp);

}

int main() {
        float weight[20], profit[20], capacity;
        int num, i, j;
        float ratio[20], temp;
        printf("Enter the no. of objects: ");
        scanf("%d", &num);
        printf("Enter the weights and profits of each object:\n");
        for (i = 0; i < num; i++) {
                scanf("%f %f", &weight[i], &profit[i]);
        }
        printf("Enter the capacity of knapsack:");
        scanf("%f", &capacity);
        for (i = 0; i < num; i++) {
                ratio[i] = profit[i] / weight[i];
        }

        for (i = 0; i < num; i++) {
                for (j = i + 1; j < num; j++) {
                        if (ratio[i] < ratio[j]) {
                                temp = ratio[j];
                                ratio[j] = ratio[i];
                                ratio[i] = temp;
                                temp = weight[j];
                                weight[j] = weight[i];
                                weight[i] = temp;
                                temp = profit[j];
                                profit[j] = profit[i];
```

```
                }
        }
        knapsack(num, weight, profit, capacity);
        return(0);
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter the no. of objects:
6
Enter the weights and profits of each object:
1 2
4 5
8 9
4 6
5 2
3 5
Enter the capacity of knapsack:
10
Maximum profit is:- 15.500000

Test Case - 2
User Output
Enter the no. of objects:
5
Enter the weights and profits of each object:
4 6
1 3
7 5
5 3
3 4
Enter the capacity of knapsack:
10
Maximum profit is:- 14.428572