STUDENT DETAILS:

NAME-ISHA MUDGAL

SMART CARD ID-BTBTC19133

CLASS -BTECH SECOND YEAR

SECTION - A2

S.N0	PROGRAM NAME	Date of assigning	DATE OF SUBMISSION
1	**Menu driven program of basic operations**	17-07-2020	18-07-2020
2	**Menu driven program of searching**	18-07-2020	24-07-2020
3	**Menu driven program of sorting**	24-07-2020	25-07-2020
4	**Menu driven program of basic array operations**	27-07-2020	31-07-2020

1. **Menu driven program to perform arithmetic operations.**

```
#include<stdio.h>
main()
{
        int num1,num2,res,opt;
        do
        {
                printf("\n enter 1st number");
                scanf("%d",&num1);
```

```
            printf("\n enter 2nd number");
            scanf("%d",&num2);
            printf(" \nMAIN MENU \n 1.Add\n 2.Subtract\n 3.Multiply\n 4.Divide \n 5.Exit");
            printf("enter your choice");
            scanf("%d",&opt);

            switch(opt)
            {
                case 1:res=num1+num2;
                    printf("add %d",res);
                    break;
                case 2:res=num1-num2;
                    printf("subtract %d",res);
                    break;
                case 3:res=num1*num2;
                    printf("multiply %d",res);
                    break;
              case 4:res=num1/num2;
                    printf("divide %d",res);
                    break;
              case 5:return;
              default: printf("you entered wrong choice");

            }
        }
         while(1);
}
```

```
 enter 1st number 12

 enter 2nd number13

MAIN MENU
 1.Add
 2.Subtract
 3.Multiply
 4.Divide
 5.Exitenter your choice1
add 25
 enter 1st number2

 enter 2nd number5

MAIN MENU
 1.Add
 2.Subtract
 3.Multiply
 4.Divide
 5.Exitenter your choice5

--------------------------------
Process exited after 21.12 seconds with return value 4200034
Press any key to continue . . .
```

2. **Menu driven program for performing Search operation in array.**

```
#include<stdio.h>

//Declaration of function

int sls(int [],int,int);

int usls(int [],int ,int);

int bs(int [],int,int);

int main()

{

        int r,n,arr[100],i,j,ele,ch;

        do

        {
```

```
        printf("Type 0 to continue and 1 to exit \n");
        scanf("%d",&j);
        if(j==1)
        return 0;
        printf("Enter the number of elements of array\n");
        scanf("%d",&n);
        printf("choices are:\n1- unsorted linear search\n 2- sorted linear search\n 3-binary
search\n");
      printf("enter choice\n");
        scanf("%d",&ch);
        switch(ch)
                {
                        case 1:
  printf("Enter array elements\n");
        for (i=0;i<n;i++)
        {
                scanf("%d",&arr[i]);
        }
        printf("search element\n");
        scanf("%d",&ele);
    r= usls(arr,n,ele);
                                        if(r>=0)
                                        printf("Element found at index %d\n",r);
                                        else
                                        printf("Element not found\n");
                                        break;

  case 2:
```

```
 printf("Enter array elements in a sorted order \n");
        for (i=0;i<n;i++)
        {
                scanf("%d",&arr[i]);
        }
        printf("search element\n");
        scanf("%d",&ele);

                                        r = sls(arr,n,ele);
                                        if(r>=0)
                                        printf("element found at index %d\n",r);
                                        else
                                        printf("Not found\n");
                                        break;

                                case 3:
printf("Enter array elements in a sorted order \n");
        for (i=0;i<n;i++)
        {
                scanf("%d",&arr[i]);
        }
        printf("search element\n");
        scanf("%d",&ele);

                                        r = bs(arr,n,ele);
                                        if(r>=0)
                                        printf("element found at index %d\n",r);
```

```
                                        else
                                        printf("element not found\n");
                                        break;


                                default :
                                        printf("wrong choice\n");
                        }
}
while(1);
}
//Defination of unsorted linear search
int usls (int arr[],int n, int ele)
{       int i=5,r;
        for (i=0;i<n;i++)
        {
                if (arr[i]==ele)
                {
                r = i;
                return r;
                }
        }
        r = -1;
        return r;
}
//Defination of sorted linear search
int sls (int arr[],int n, int ele)
{       int i,r;
```

```
	for (i=0;i<n;i++)
	{	if(arr[i]<=ele)
	{
		if (arr[i]==ele)
		{
		r = i;
		return r;
		}
	}
	}
	r = -1;
	return r;
}
//Defination of binary search
int bs(int arr[],int n, int ele)
{
	int low=0,high=n-1,mid;
	while (low<=high)
	{
		mid=(low+high)/2;
		if(arr[mid]>ele)
		high=mid-1;
		else if (arr[mid]<ele)
		low=mid+1;
		else
		return mid;
	}
```

```
        return -1;
}
```

```
Type 0 to continue and 1 to exit
0
Enter the number of elements of array
 5
choices are:
1- unsorted linear search
 2- sorted linear search
 3-binary search
enter choice
 2
Enter array elements in a sorted order
1 3 5 7 9
search element
3
element found at index 1
Type 0 to continue and 1 to exit
1

--------------------------------
Process exited after 53.21 seconds with return value 0
Press any key to continue . . .
```

ALGORITHM

-Linear Search

Step 1: Set i=1

Step 2: if i>n, step 6 is executed

Step 3: if elem is not equal to arr[i], step 2 is executed

Step 4: if found=1, break executed

Step 5: element found is printed

Step 6: end

- Binary Search

Step 1: set low=0 and high=9

Step 2: check high<low, step 9 is executed

Step 3: if arr[mid]==elem, break takes program to step 9

Step 4: if arr[mid]> elem,step 5 executed

Step 5: set high=mid-1, step 2 executed

Step 6: if arr[mid]<elem, step 7 executed

Step 7: set low=mid+1,step 2 executed

Step 8: set mid=(low+high)/2

Step 9: end

3. Menu driven program for Sorting of an array.

```
#include<stdio.h>
//Function declaration
void bubs(int [],int n);
void sels(int [],int n);
void inss(int [],int n);
void sortedarr(int [],int n);
main()
{
   int i,j,n,ch,arr[100];
   do
   {
      printf("enter 0 to continue and 1 to exit");
      scanf("%d",&j);
      if(j==1)
      return (0);
   printf("enter no of elements in array\n");
```

```
   scanf("%d",&n);
   printf("enter array elements\n");
   for(i=0;i<n;i++)
   {
      scanf("%d",&arr[i]);
   }
   printf("choices are:1-bubble sort\n 2-optimised bubble sort\n 3- selection sort\n 4-insertion
sort\n");
      printf("enter your choice");
      scanf("%d",&ch);
      switch(ch)
      {
         case 1: bubs(arr,n);
         {
            sortedarr(arr,n);
         }
         break;
         case 2: optimisedbs(arr,n);
         {
           sortedarr(arr,n);
         }
          case 3: sels(arr,n);
         {
            sortedarr(arr,n);
         }
         break;
         case 4: inss(arr,n);
         {
```

```
            sortedarr(arr,n);
        }
        break;
        default: printf("wrong choice");
      }

  }
  while(1);
  }
//Bubble sort definition
void bubs(int arr[], int n)
{
  int i,j,temp;
  for(i=0;i<n;i++)
  {
    for(j=0;j<(n-1);j++)
    {
      if(arr[j]>arr[j+1])
      {
        temp=arr[j];
        arr[j]=arr[j+1];
        arr[j+1]=temp;
      }
    }
  }
}
//Optimised Bubble Sort definition
```

```
void optimisedbs(int arr[], int n)
{
int i,j,temp,flag;
for(i=0;i<n-1;i++)
   {
      flag=0;
      for(j=0;j<(n-i-1);j++)
      {
         if(arr[j]>arr[j+1])
         {
            flag=1;
            temp=arr[j];
             arr[j]=arr[j+1];
             arr[j+1]=temp;
         }
      }
      if(flag==1)
      return;
   }
}
//Selection sort definition
void sels(int arr[], int n)
{
   int i,j,min,temp;
   for(i=0;i<(n-1);i++)
   {
      min=i;
```

```
        for(j=i+1;j<n;j++)
        {
            if(arr[j]<arr[min])
            {
                min=j;
            }
        }
        temp=arr[i];
        arr[i]=arr[min];
        arr[min]=temp;
    }
}
//Insertion sort definition
void inss(int arr[],int n)
{
    int i,j,temp;
    for(i=0;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;
    }
```

```
}

//Definition for printing sorted array

void sortedarr(int arr[], int n)

{

   int i;

   for(i=0;i<n;i++)

   {

      printf("%d",arr[i]);

   }

}
```

```
enter 0 to continue and 1 to exit 0
enter no of elements in array
 5
enter array elements
 1 3 2 4 6
choices are:1-bubble sort
 2-optimised bubble sort
 3- selection sort
 4-insertion sort
enter your choice 1
12346enter 0 to continue and 1 to exit 0
enter no of elements in array
 4
enter array elements
1 4 2 5 7
choices are:1-bubble sort
 2-optimised bubble sort
 3- selection sort
 4-insertion sort
enter your choicewrong choiceenter 0 to continue and 1 to exit 1

--------------------------------
Process exited after 30.69 seconds with return value 0
Press any key to continue . . .
```

ALGORITHM

-Bubble Sort:

Step 1: set i=0

Step 2: if i>n step 9 executed

Step 3: set j=0

Step 4: if j>n-1, step 8 is executed

Step 5: arr[j]<arr[j+1] ,step 7 executed

Step 6: swap(arr[j],are[j+1])

Step 7: set j=j+1, step 5 executed

Step 8: set i=i+1, step 3 is executed

Step 9: exit

-Selection Sort:

Step 1: set i=0

Step 2: if i>=n step 10 is executed

Step 3: set min=i

Step 4: set j=i+1

Step 5: if j>n, step 10 executed

Step 6: if arr[j]>arr[min], step 8 is executed

Step 7:set min=j

Step 8: j=j+1, step 6 executed

Step 9: swap(arr[i],arr[min])

Step 10: i=i+1, step 2 executed

Step 11: exit

-Insertion Sort:

Step 1: Set i=0

Step 2: if i>n, step 10 is executed

Step 3: set temp=arr[i]

Step 4: set j=j-1

Step 5: if j>0 and are[j]>temp unsatisfied, step9 is executed

Step 6: check arr[j+1]=arr[j]

Step 7: set j=j-1, step 5 executed

Step 8: set arr[j+1]=temp

Step 9: i++

Step 10: exit

4. **Menu driven program for Array Operations**.

```
#include<stdio.h>
//Function declaration
void ins(int [],int n);
void del(int [],int n);
void rev(int [],int n);
void insarr(int [],int n);
void delarr(int[],int n);
void revarr(int [],int n);
main()
{
   int i,j,n,ch,arr[100];
```

```
do
{
   printf("enter 0 to continue and 1 to exit");
   scanf("%d",&j);
   if(j==1)
   return (0);
printf("enter no of elements in array\n");
scanf("%d",&n);
printf("enter array elements\n");
for(i=0;i<n;i++)
{
   scanf("%d",&arr[i]);
}
printf("choices for array operation are:1-insertion\n 2-deletion\n 3- reversal\n");
   printf("enter your choice");
   scanf("%d",&ch);
   switch(ch)
   {
      case 1: ins(arr,n);
      {
         printf("Array after insertion is:\n");
         insarr(arr,n);
      }
      break;
      case 2: del(arr,n);
      {
         printf("array after deletion:\n");
```

```
            delarr(arr,n);
          }
          break;
           case 3: rev(arr,n);
          {
              printf("array after reversal:\n");
              revarr(arr,n);
          }
          break;
          default: printf("wrong choice");
        }

    }
    while(1);
    }
//Insertion definition
void ins(int arr[], int n)
{
    int i,index,n1;
    printf("enter no to be inserted\n");
    scanf("%d",&n1);
    printf("enter index where no is to be inserted\n");
    scanf("%d",&index);
    for(i=n;i>=index;i--)
    {
       arr[i+1]=arr[i];
    }
```

```
   arr[index]=n1;
   n++;
}
//Deletion definition
void del(int arr[], int n)
{
    int i,index, n1;
   printf("enter no to be deleted\n");
   scanf("%d",&n1);
   printf("enter index from where no is to be deleted\n");
   scanf("%d",&index);
   if(index<0||index>n)
   {
      printf("wrong deletion");
   }
   else
   {
    for(i=index;i<n;i++)
    {
      arr[i]=arr[i+1];
    }
    n--;
   }
}
//Reversal definition
void rev(int arr[], int n)
{
```

```
    int i,j,temp;
    i=0;
    j=n-1;
    while(i<j)
    {
        temp=arr[i];
        arr[i]=arr[j];
        arr[j]=temp;
    i++;
    j--;
    }
}
//Definition for printing final array
void insarr(int arr[], int n)
{
    int i;
    for(i=0;i<(n+1);i++)
    {
        printf("%d",arr[i]);
    }
}
//Definition for printing array after deletion
void delarr(int arr[], int n)
{
    int i;
    for(i=0;i<(n-1);i++)
    {
```

```
        printf("%d",arr[i]);
    }
}
//Definition for printing array after reversal
void revarr(int arr[], int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        printf("%d",arr[i]);
    }
}
```

```
enter 0 to continue and 1 to exit 0
enter no of elements in array
 5
enter array elements
 1 3 2 4 6
choices for array operation are:1-insertion
 2-deletion
 3- reversal
enter your choice 1
enter no to be inserted
7
enter index where no is to be inserted
2
Array after insertion is:
137246enter 0 to continue and 1 to exit 1

--------------------------------
Process exited after 41.61 seconds with return value 0
Press any key to continue . . .
```