```
#include<stdio.h>

#define n 7

int a[50] ,b[50];
int array[50] ,len;

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

void swap(int i,int j,int arr[]){
	int tmp;
	tmp=arr[i];
	arr[i]=arr[j];
	arr[j]=tmp;
}

void bubblesort(int arr[] ,int len){
	int tmp;
	printf("before bubble sort : ");
	printarray(arr,len);
	printf("\n");

	for(int i=0 ;i<len-1 ;i++){
		for(int j=0 ;j<len-i-1 ;j++)
			if(arr[j]>arr[j+1])
				swap(j,j+1,arr);

		printf("\t");
		printarray(arr,len);
		printf("\n");
	}
}

void insertionsort(int arr[] ,int len){
	int tmp,minv,j;
	printf("before insertion sort : ");
	printarray(arr,len);
	printf("\n");

	for(int i=1 ;i<len ;i++){
		minv=arr[i];
		for(j=i-1 ;j>=0 ;j--){
			if(minv<arr[j])
				arr[j+1]=arr[j];
			else
				break;
```

```
                }
                arr[j+1]=minv;

                printf("\t");
                printarray(arr,len);
                printf("\n");
        }
}

void selectionsort(int arr[] ,int len){
        int min,tmp;
        printf("before selection sort : ");
        printarray(arr,len);
        printf("\n");

        for(int i=0 ;i<len-1 ;i++){
                min=i;
                for(int j=i+1 ;j<len ;j++)
                        if(arr[j]<arr[min])
                                min=j;

                if(arr[i]>arr[min])
                        swap(i,min,arr);
                printf("\t");
                printarray(arr,len);
                printf("\n");
        }
}

//merge sort
void mergesort(int low, int mid, int high, int arr[] ){

        int k=low,i=low,j=mid+1;
        while((i<=mid) && (j<=high)){
                if(arr[i] <= arr[j]){
                        b[k]=arr[i];
                        i++;
                }
                else{
                        b[k]=arr[j];
                        j++;
                }
                k++;
        }
        while(i<=mid){
                b[k]=arr[i];
                k++;
                i++;
        }
        while(j<=high){
```

```c
                b[k]=arr[j];
                k++;
                j++;
        }
        for(i=low; i<=high; i++)
                arr[i]=b[i];

        printf("\t");
        printarray(arr,len);
        printf("\n");
}
void mergesplit(int arr[], int low, int hi){
        if(low < hi){
                int mid=(low+hi)/2;
                mergesplit(arr ,low ,mid);
                mergesplit(arr ,mid+1 ,hi);
                mergesort(low ,mid ,hi ,arr);
        }
}


//quicksort
void quicksort(int arr[], int first, int last){
    if(first<last){
        int i,j,pivot;
        i=first;
        j=last;
        pivot=arr[first];
        while(i<j){
            while(arr[i]<=pivot && i<last)
                i++;
            while(arr[j]>=pivot && j>first)
                j--;
            if(i<j)
                swap(i,j,arr);
        }
        swap(j,first,arr);

                printf("\t");
                printarray(arr,len);
                printf("\n");

        quicksort(arr,first,j-1);
        quicksort(arr,j+1,last);
    }
}

int main(){
        len=8;
        int array[]={100,5,3,8,2,7,9,1};
```

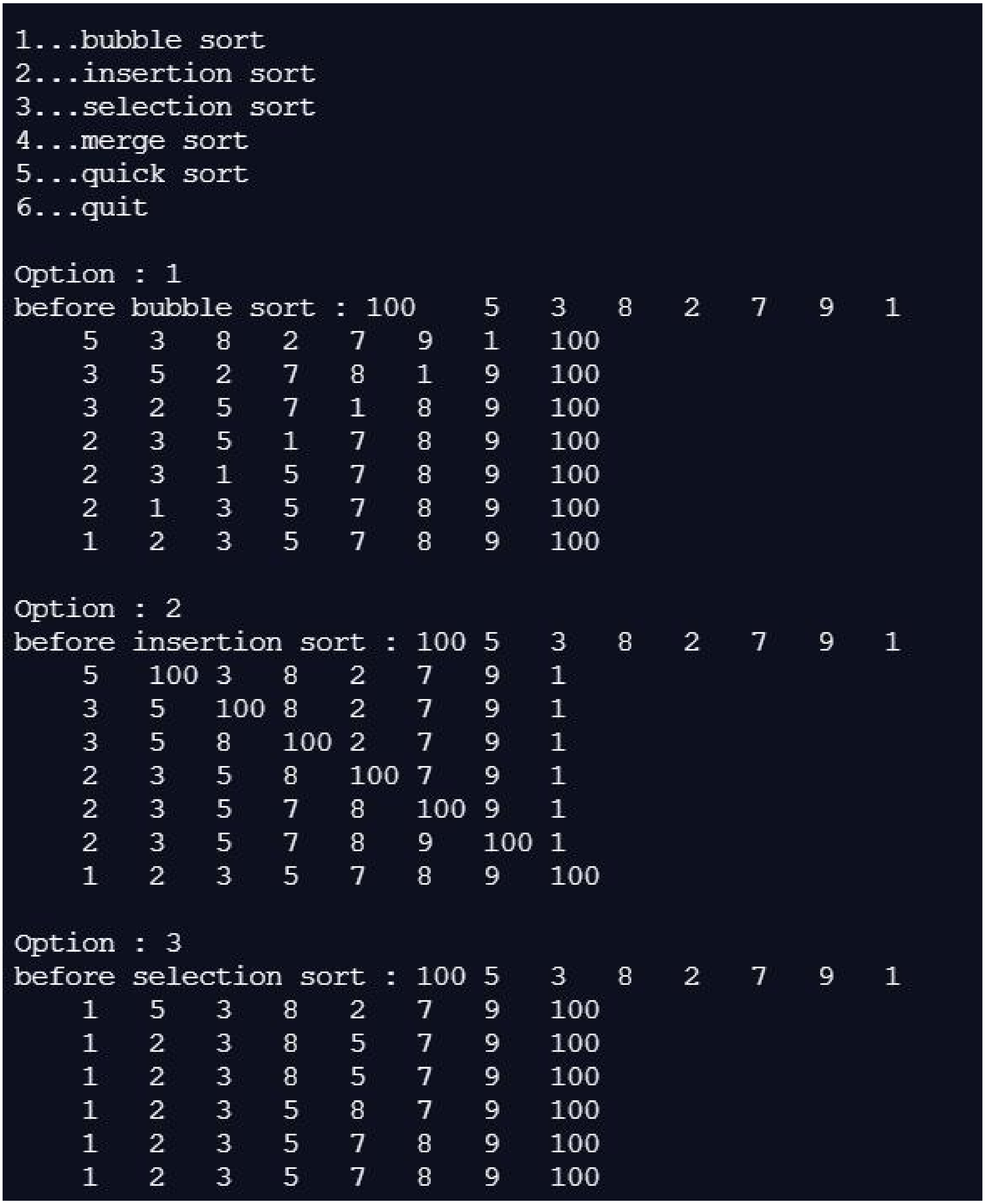
```
//      printf("enter the array size : ");
//      scanf("%d",&len);

//      printf("enter the array elements : ");
//      for(int i=0;i<len;i++)
//          scanf("%d",&array[i]);

    int choice, pos;

    printf("\n1...bubble sort\n");
    printf("2...insertion sort\n");
    printf("3...selection sort\n");
    printf("4...merge sort\n");
    printf("5...quick sort\n");
    printf("6...quit\n");
    int quit=1;
    while(quit!=0){
        printf("\nOption : ");
        scanf("%d",&choice);
        for(int i=0; i<len; i++)
        {
                a[i]=array[i];
        }
        switch(choice){
            case 1: bubblesort(a,len);
                break;
            case 2: insertionsort(a,len);
                break;
                    case 3: selectionsort(a,len);
                break;
                    case 4: printf("before merge sort : ");
                printarray(a,len);
                printf("\n");
                mergesplit(a ,0 ,len-1);
                break;
                    case 5: printf("before quick sort : ");
                printarray(a,len);
                printf("\n");
                quicksort(a ,0 ,len-1);
                break;
            case 6: printf("***program terminated*** ");
                quit=0;
                break;
            default:
                        printf("\n1...bubble sort\n");
                        printf("2...insertion sort\n");
                        printf("3...selection sort\n");
                        printf("4...merge sort\n");
                        printf("5...quick sort\n");
                        printf("6...quit\n");
        }
    }

        return 0;
}
```



```
1...bubble sort
2...insertion sort
3...selection sort
4...merge sort
5...quick sort
6...quit

Option : 1
before bubble sort : 100     5    3    8    2    7    9    1
    5    3    8    2    7    9    1    100
    3    5    2    7    8    1    9    100
    3    2    5    7    1    8    9    100
    2    3    5    1    7    8    9    100
    2    3    1    5    7    8    9    100
    2    1    3    5    7    8    9    100
    1    2    3    5    7    8    9    100

Option : 2
before insertion sort : 100 5    3    8    2    7    9    1
    5    100 3    8    2    7    9    1
    3    5    100 8    2    7    9    1
    3    5    8    100 2    7    9    1
    2    3    5    8    100 7    9    1
    2    3    5    7    8    100 9    1
    2    3    5    7    8    9    100 1
    1    2    3    5    7    8    9    100

Option : 3
before selection sort : 100 5    3    8    2    7    9    1
    1    5    3    8    2    7    9    100
    1    2    3    8    5    7    9    100
    1    2    3    8    5    7    9    100
    1    2    3    5    8    7    9    100
    1    2    3    5    7    8    9    100
    1    2    3    5    7    8    9    100
```

Sorting Algo

Algorithm

```
I  void printarray (int arr[], int len)

    1. START
    2. for (int i = 0 to len)
        1. print (arr[i] + "\t")
    3. End for
    4. STOP
```

```
~~II  void swap (int i, int j, int arr[])~~

    ~~1. START~~
    ~~2. swap int~~
```

```
III  void bubblesort (int arr[], int len)

    1. START
    2. ~~print ("before~~ for(int i = 0 to len-1)
        1. for( int j=0 to len-1-i)
            1. If (arr[j] > arr[j+1]
                1. swap (~~arr~~ arr[i], arr[j]
            2. End If
        2. End for
        3. printarray (arr, len)
    3. End for
    4. STOP
```

III void insertionsort (int arr[], int len)

```
1. START
2. for (int i = 1 to len - 1)
    1. ~~for~~ minval = arr[i];
    2. for (int j = i-1 to 0)
        1. If (minval < arr[j])
            1. arr [j + 1] = arr [j]
        2. Else
            1. break.
    3. End for
    4. arr[j + 1] = min val
    5. Print array (arr, len)
~~3. End for~~
3. End for
4. STOP
```

IV void selectionsort (int arr[], int len)

```
1. START
2. int min;
3. for (int ~~j~~ i = 0 to len - 1)
    1. min = i
    2. for (int j = i+1 to len)
```

```
            1. if (arr[j] < arr[min])
                    1. min = j
            2. End If
        3. End for
        4. if (arr[i] > arr[min])
                1. swap( ~~i, min~~ arr[i], arr[min])
        5. End If
        6. printarray (arr, len)
        7 STOP
```

V void main ()

```
1. START
2. create array
3. call bubblesort, insertion sort & selection
   sort
4. ~~sort~~ STOP
```