Experiment : 1

```
// Bubble Sort Program

#include<stdio.h>

void bubbleSort(int *arr, int arrSize)
{
	for (int i=0; i<arrSize-1; ++i)
	{
		for(int j=0; j<arrSize-1-i; ++j)
		{
			if (arr[j]>arr[j+1])
			{
				int temp = arr[j];
				arr[j] = arr[j+1];
				arr[j+1] = temp;
			}
		}
	}
}

void main()
{
	int arrSize;
```

```
	scanf("%d", &arrSize);

	int arr[arrSize];

	for (int i=0; i<arrSize; ++i)
		scanf("%d", &arr[i]);

	bubbleSort(&arr, arrSize);

	for (int i=0; i<arrSize; ++i)
		printf("%d ", arr[i]);
}


// Selection Sort

#include<stdio.h>

void selectionSort(int *arr, int arrSize)
{
	for (int i=0; i<arrSize-1; ++i)
	{
		for(int j=i+1; j<arrSize; ++j)
		{
			if (arr[i]>arr[j])
			{
```

```
                        int temp = arr[i];
                        arr[i] = arr[j];
                        arr[j] = temp;
                    }
                }
            }
}

void main()
{
    int arrSize;

    scanf("%d", &arrSize);

    int arr[arrSize];

    for (int i=0; i<arrSize; ++i)
        scanf("%d", &arr[i]);

    selectionSort(&arr, arrSize);

    for (int i=0; i<arrSize; ++i)
        printf("%d ", arr[i]);
}
```

```
// Insertion Sort

#include<stdio.h>

void insertionSort(int *arr, int arrSize)
{
        int i, key, j;
    for (i = 1; i < arrSize; i++)
    {
        key = arr[i];
        j = i - 1;

        while (j >= 0 && arr[j] > key)
        {
            arr[j + 1] = arr[j];
            j = j - 1;
        }

        arr[j + 1] = key;
    }
}

void main()
{
        int arrSize;
```

```
        scanf("%d", &arrSize);

        int arr[arrSize];

        for (int i=0; i<arrSize; ++i)
                scanf("%d", &arr[i]);

        insertionSort(&arr, arrSize);

        for (int i=0; i<arrSize; ++i)
                printf("%d ", arr[i]);
}


// Counting Sort

#include <stdio.h>

void countingSort(int *arr, int size) {
  int output[10];

  int max = arr[0];
  for (int i = 1; i < size; i++) {
    if (arr[i] > max)
      max = arr[i];
  }
```

```
  int count[10];

  for (int i = 0; i <= max; ++i) {
    count[i] = 0;
  }

  for (int i = 0; i < size; i++) {
    count[arr[i]]++;
  }

  for (int i = 1; i <= max; i++) {
    count[i] += count[i - 1];
  }

  for (int i = size - 1; i >= 0; i--) {
    output[count[arr[i]] - 1] = arr[i];
    count[arr[i]]--;
  }

  for (int i = 0; i < size; i++) {
    arr[i] = output[i];
  }
}

void printArray(int arr[], int size) {
```

```
  for (int i = 0; i < size; ++i) {
    printf("%d ", arr[i]);
  }
  printf("¥n");
}

void main()
{
        int arrSize;

        scanf("%d", &arrSize);

        int arr[arrSize];

        for (int i=0; i<arrSize; ++i)
                scanf("%d", &arr[i]);

        countingSort(&arr, arrSize);
        printArray(&arr, arrSize);
}

// Radix Sort

#include <stdio.h>

int getMax(int *arr, int n) {
```

```
  int max = arr[0];
  for (int i = 1; i < n; i++)
    if (arr[i] > max)
      max = arr[i];
  return max;
}


void countingSort(int *arr, int size, int place) {
  int output[size + 1];
  int max = (arr[0] / place) % 10;

  for (int i = 1; i < size; i++) {
    if (((arr[i] / place) % 10) > max)
      max = arr[i];
  }
  int count[max + 1];

  for (int i = 0; i < max; ++i)
    count[i] = 0;


  for (int i = 0; i < size; i++)
    count[(arr[i] / place) % 10]++;
```

```
  for (int i = 1; i < 10; i++)
    count[i] += count[i - 1];


  for (int i = size - 1; i >= 0; i--) {
    output[count[(arr[i] / place) % 10] - 1] = arr[i];
    count[(arr[i] / place) % 10]--;
  }

  for (int i = 0; i < size; i++)
    arr[i] = output[i];
}


void radixSort(int *arr, int size) {

  int max = getMax(arr, size);

  for (int place = 1; max / place > 0; place *= 10)
    countingSort(arr, size, place);
}

void printArray(int *arr, int size) {
  for (int i = 0; i < size; ++i) {
    printf("%d ", arr[i]);
  }
```

```
  printf("¥n");
}

int main() {
        int arrSize;

        scanf("%d", &arrSize);

        int arr[arrSize];

        for (int i=0; i<arrSize; ++i)
                scanf("%d", &arr[i]);

        radixSort(arr, arrSize);

        printf("Sorted array: ¥n");
        printArray(arr, arrSize);
}
```