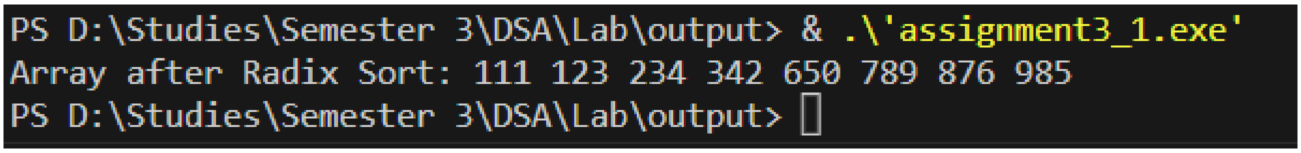
Problem 1:
Code:

```
#include <iostream>



class DigitNode {

public:

   int data;

   DigitNode* next;

   DigitNode(int value) : data(value), next(nullptr) {}

};



void addToDigitBucket(DigitNode* digitBuckets[], int digit, DigitNode* node) {

   if (!digitBuckets[digit]) {

      digitBuckets[digit] = node;

   } else {

      DigitNode* current = digitBuckets[digit];

      while (current->next) {

         current = current->next;

      }

      current->next = node;

   }

}



void radixSort(int arr[], int n) {

   const int numDigits = 10;
```

```
DigitNode* digitBuckets[numDigits] = {nullptr};
int maxNumber = arr[0];
for (int i = 1; i < n; ++i) {
    if (arr[i] > maxNumber) {
        maxNumber = arr[i];
    }
}
int exp = 1;
while (maxNumber / exp > 0) {

    for (int i = 0; i < n; ++i) {
        int digit = (arr[i] / exp) % 10;
        addToDigitBucket(digitBuckets, digit, new DigitNode(arr[i]));
    }

    int index = 0;
    for (int i = 0; i < numDigits; ++i) {
        DigitNode* current = digitBuckets[i];
        while (current) {
            arr[index++] = current->data;
            DigitNode* temp = current;
            current = current->next;
            delete temp;
        }
        digitBuckets[i] = nullptr;
    }
    exp *= 10;
}
```

```
}



int main() {
    int arr[] = {342, 876, 123, 985, 650, 234, 789, 111};
    int n = sizeof(arr) / sizeof(arr[0]);
    radixSort(arr, n);
    std::cout << "Array after Radix Sort: ";
    for (int i = 0; i < n; ++i) {
        std::cout << arr[i] << " ";
    }
    std::cout << std::endl;
    return 0;
}
```

Output:



```
PS D:\Studies\Semester 3\DSA\Lab\output> & .\'assignment3_1.exe'
Array after Radix Sort: 111 123 234 342 650 789 876 985
PS D:\Studies\Semester 3\DSA\Lab\output> 
```

Problem 2:

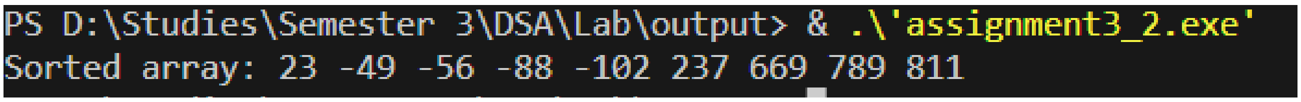
Code:

```
#include <iostream>

#include <vector>

#include <cmath>



class DataNode {

public:

   int value;

   DataNode* next;

   DataNode(int val) : value(val), next(nullptr) {}

};



void addToBucket(DataNode* buckets[], int digit, DataNode* node) {

   if (!buckets[digit]) {

      buckets[digit] = node;

   } else {

      DataNode* current = buckets[digit];

      while (current->next) {

         current = current->next;

      }

      current->next = node;

   }

}



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

```
const int numBuckets = 10;
DataNode* buckets[numBuckets] = {nullptr};
int maxNum = std::abs(arr[0]);


for (int i = 1; i < size; ++i) {
    if (std::abs(arr[i]) > maxNum) {
        maxNum = std::abs(arr[i]);
    }
}


int exp = 1;
while (maxNum / exp > 0) {
    for (int i = 0; i < size; ++i) {
        int digit = (std::abs(arr[i]) / exp) % 10;
        addToBucket(buckets, digit, new DataNode(arr[i]));
    }


    int index = 0;
    for (int i = 0; i < numBuckets; ++i) {
        DataNode* current = buckets[i];
        while (current) {
            arr[index++] = current->value;
            DataNode* temp = current;
            current = current->next;
            delete temp;
        }
```

```
            buckets[i] = nullptr;
        }
        exp *= 10;
    }
}



int main() {
    int arr[] = {237, -56, 811, -102, 669, -49, 23, -88, 789};
    int size = sizeof(arr) / sizeof(arr[0]);

    performRadixSort(arr, size);

    std::cout << "Sorted array: ";
    for (int i = 0; i < size; ++i) {
        std::cout << arr[i] << " ";
    }
    std::cout << std::endl;

    return 0;
}
```

Output:



```
PS D:\Studies\Semester 3\DSA\Lab\output> & .\'assignment3_2.exe'
Sorted array: 23 -49 -56 -88 -102 237 669 789 811
```

```
PS D:\Studies\Semester 3\DSA\Lab\output>
```