1)

#include <iostream>

using namespace std;

 int getMax(int arr[], int n)

{

    int mx = arr[0];

    for (int i = 1; i < n; i++)

        if (arr[i] > mx)

            mx = arr[i];

    return mx;

}

void countSort(int arr[], int n, int exp)

{

    // Output array

    int output[n];

    int i, count[10] = { 0 };

    for (i = 0; i < n; i++)

        count[(arr[i] / exp) % 10]++;

    for (i = 1; i < 10; i++)

        count[i] += count[i - 1];

    // Build the output array

    for (i = n - 1; i >= 0; i--) {

        output[count[(arr[i] / exp) % 10] - 1] = arr[i];

        count[(arr[i] / exp) % 10]--;

    }

    for (i = 0; i < n; i++)

        arr[i] = output[i];

}

void radixsort(int arr[], int n)

{

    int m = getMax(arr, n);

    for (int exp = 1; m / exp > 0; exp \*= 10)

        countSort(arr, n, exp);

}

// print an array

void print(int arr[], int n)

{

    for (int i = 0; i < n; i++)

        cout << arr[i] << " ";

}

int main()

{

    int arr[] = { 136,487,358,469,570,247,598,639,205,609};

    int n = sizeof(arr) / sizeof(arr[0]);

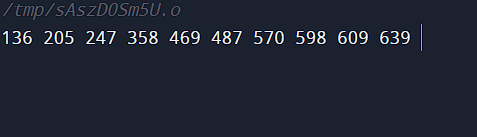
    radixsort(arr, n);

    print(arr, n);

    return 0;

}

Output:



2)

#include <stdio.h>

#include <stdlib.h>

// Define a structure for a single linked list node

struct Node {

    int data;

    struct Node\* next;

};

/t

void insert(struct Node\*\* head, int data) {

    struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

    newNode->data = data;

    newNode->next = NULL;

    if (\*head == NULL) {

        \*head = newNode;

    } else {

        struct Node\* current = \*head;

        while (current->next != NULL) {

            current = current->next;

        }

        current->next = newNode;

    }

}

int getMax(struct Node\* head) {

    int max = head->data;

    struct Node\* current = head->next;

    while (current != NULL) {

        if (current->data > max) {

            max = current->data;

        }

        current = current->next;

    }

    return max;

}

void countingSort(struct Node\*\* head, int exp) {

    if (\*head == NULL) {

        return;

    }

    int count[10] = {0};

    struct Node\* current = \*head;

    while (current != NULL) {

        count[(current->data / exp) % 10]++;

        current = current->next;

    }

    struct Node\* output = NULL;

    struct Node\* buckets[10] = {NULL};

    for (int i = 0; i < 10; i++) {

        if (count[i] > 0) {

            struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

            newNode->data = i;

            newNode->next = NULL;

            buckets[i] = newNode;

        }

    }

    for (int i = 0; i < 10; i++) {

        struct Node\* bucket = buckets[i];

        while (bucket != NULL) {

            insert(&output, bucket->data);

            struct Node\* temp = bucket;

            bucket = bucket->next;

            free(temp);

        }

    }

    //\*head = output;

     current = \*head;

    while (current != NULL) {

        struct Node\* temp = current;

        current = current->next;

        free(temp);

    }

    \*head = output;

}

void radixSort(struct Node\*\* head) {

    if (\*head == NULL) {

        return;

    }

    int max = getMax(\*head);

    for (int exp = 1; max / exp > 0; exp \*= 10) {

        countingSort(head, exp);

    }

}

void printList(struct Node\* head) {

    struct Node\* current = head;

    while (current != NULL) {

        printf("%d ", current->data);

        current = current->next;

    }

    printf("\n");

}

int main() {

    struct Node\* head = NULL;

    insert(&head, 136);

    insert(&head, 487);

    insert(&head, 358);

    insert(&head, 469);

    insert(&head, 570);

    insert(&head, 247);

    insert(&head, 598);

    insert(&head, 639);

    insert(&head, 205);

    insert(&head, 609);

    printf("Original list: ");

    printList(head);

    radixSort(&head);

    printf("Sorted list: ");

    return 0;

}

Output:

