CODING PROJECT 1
CSN-212 Design & Analysis of Algorithms

Group 18 - 14114009_14114006
Ambar Zaidi_Akshit Kalra

Sorting Algorithms

7h September 2016

OBJECTIVE

- To know implementation of different sorting algorithms.

OVERVIEW

Sorting is the process of placing elements from a collection in some kind of order. There are many sorting algorithms that have been developed and analyzed. In this project, we will implement various sorting algorithms using arrays and linked lists as data structures.

CPU TIME	for 10 elements		for 10000 elements	(in ms)
	array	list	array	list
Insertion Sort	0	0	1065	2062
Selection Sort	0	0	1969	2964
Bubble Sort	0	0	1977	2045
Shell Sort	0	0	1945	3048
Merge Sort	0	0	1045	2003
Quick Sort	0	0	1057	1983
Heap Sort	0	0	1079	2164
Counting Sort	0	0	1049	1980
Radix Sort	0	0	1074	1999
Bucket Sort	0	0	1050	2005

main.cpp

```
#include <iostream>
#include <stdlib.h>
#include <sys/time.h>

using namespace std;

#include "utilities.h"
#include "linkedlist.h"

#include "insertionSortlist.h"
#include "selectionSortlist.h"
#include "bubblesortlist.h"
#include "shellSortlist.h"
#include "mergeSortlist.h"
#include "quickSortlist.h"
#include “heapSortlist.h”
#include "countingSortlist.h"
#include "radixSortlist.h"
#include "bucketSortlist.h"

#include "insertionSortarray.h"
#include "selectionSortarray.h"
```

```
#include "bubbleSortarray.h"
#include "shellSortarray.h"
#include "mergeSortarray.h"
#include "quickSortarray.h"
#include "countingSortarray.h"
#include "heapSortarray.h"
#include "radixSortarray.h"
#include "bucketSortarray.h"

int main()
{

///////////////////////////////////////////////// TESTING FOR LINKED LIST
    node* a=createlist();
    int n = 10;
    for(int i=0;i<n;i++)
    {
        addnode(a,(rand()%200));
    }
    cout<<"Number of elements in array: "<<n<<"\n\n";
    printlist(a);

    struct timeval time1, time2;
    gettimeofday(&time1, NULL);
    //cout<<"\n\nUsing Insertion Sort: \n";   insertionSortList(a);

    //cout<<"\n\nUsing Selection Sort: \n";   selectionSortList(a);

    //cout<<"\n\nUsing Bubble Sort: \n";   bubbleSortList(a);

    //cout<<"\n\nUsing Shell Sort: \n";   shellSortList(a);

    cout<<"\n\nUsing Merge Sort: \n";   mergeSortList(a);

    //cout<<"\n\nUsing Quick Sort: \n";   quickSortList(a);

    //cout<<"\n\nUsing Heap Sort: \n";   heapSortArray()SortList(a);

    //cout<<"\n\nUsing Counting Sort: \n";   countingSortList(a);

    //cout<<"\n\nUsing Radix Sort: \n";   radixSortList(a);

    //cout<<"\n\nUsing Bucket Sort: \n";   bucketSortList(a);

    printlist2(a);


///////////////////////////////////////////////// TESTING FOR ARRAY


//    int b[10000] ;

//    int n = 10000;

//    randomArray(b,n);

//    cout<<"Number of elements in array: "<<n<<"\n\n";

//    cout<<"Initial Array: \n";

//    print(b,n);

//

//    struct timeval time1, time2;

//    gettimeofday(&time1, NULL);
```

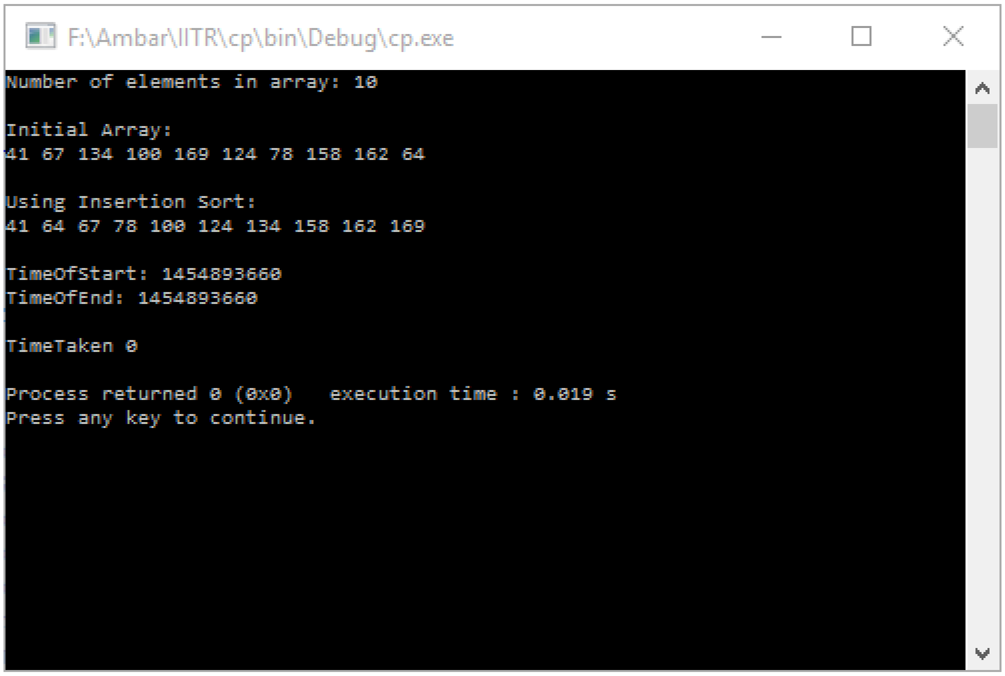
```
//

//    //cout<<"\n\nUsing Insertion Sort: \n";     insertionSortArray(b,n);

//    //cout<<"\n\nUsing Selection Sort: \n";     selectionSortArray(b,n);

//    //cout<<"\n\nUsing Bubble Sort: \n";        bubbleSortArray(b,n);

//    //cout<<"\n\nUsing Shell Sort: \n";         shellSortArray(b,n);

//    //cout<<"\n\nUsing Merge Sort: \n";         mergeSortArray(b,n);

//    //cout<<"\n\nUsing Quick Sort: \n";         quickSortArray(b,n);

//    //cout<<"\n\nUsing Heap Sort: \n";          heapSortArray(b,n);

//    cout<<"\n\nUsing Counting Sort: \n";      countingSortArray(b,n);

//    //cout<<"\n\nUsing Radix Sort: \n";         radixSortArray(b,n);

//    //cout<<"\n\nUsing Bucket Sort: \n";        bucketSortArray(b,n);

//    print(b,n);


//////////////////////////////////////////////////////  TIMING OUTPUT

   gettimeofday(&time2, NULL);

   cout<<"\n\nTimeOfStart"<<": "<<time1.tv_sec;

   cout<<"\nTimeOfEnd"<<": "<<time2.tv_sec<<endl;

   long long int ms1 = time1.tv_sec * 1000 + time1.tv_usec/10000 ;

   long long int ms2 = time2.tv_sec * 1000 + time2.tv_usec/10000;

   cout<<"\nTimeTaken "<<(long long)(ms2-ms1)<<endl;

}
```

ALGORITHMS

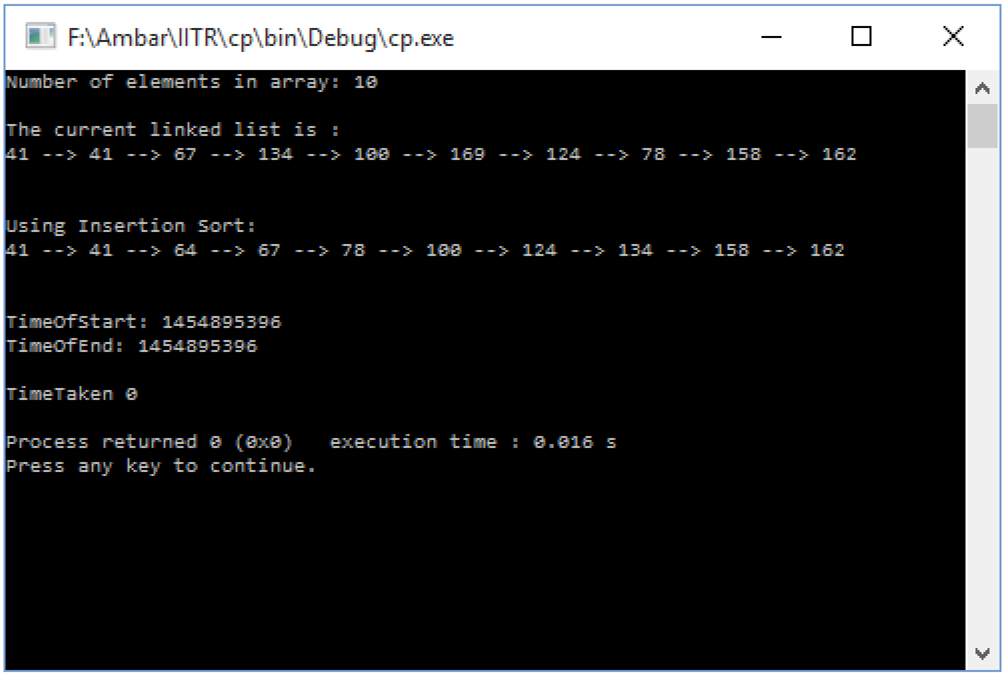
Insertion Sort

```
//////////////////////////////////////////////////////////////// INSERTION SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  insertionSortarray.h - Insertion Sort for input type array

//////////////////////////////////////////////////////////////// INSERTION SORT


void insertionSortArray(int a[],int n){

    for(int i=1;i<n;i++){

        for(int j=i;j>0;j--){

            if(a[j]<a[j-1]){

                int temp=a[j];

                a[j]=a[j-1];

                a[j-1]=temp;

            }

        }

    }

}
```

```
////////////////////////////////////////////////////////////// INSERTION SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  insertionSortlist.h - Insertion Sort for input type list

////////////////////////////////////////////////////////////// INSERTION SORT


void insertionSortList(node* head){

    int len=length(head);

    if(len==0)return ;

    if(len==1)return ;

    node* cur=head->next->next;

    for(int i=2;i<=len;i++){

        int xyz=cur->data;

        node* temp=head->next;

        int posi=1;

        while(xyz>temp->data ){

            temp=temp->next;

            posi++;

            if(posi>=i)break;

        }


        xyz=1;

        temp=head;

        while(xyz<posi){
            temp=temp->next;
            xyz++;
        }
        xyz=cur->data;
        deleteNode(head,i);
        node* abcd=  (node*)malloc(sizeof(node));
        abcd->data=xyz;
        abcd->next=temp->next;
        temp->next=abcd;
        cur=cur->next;
        //printlist(head);
    }
}
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Insertion Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454893660
TimeOfEnd: 1454893660

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```

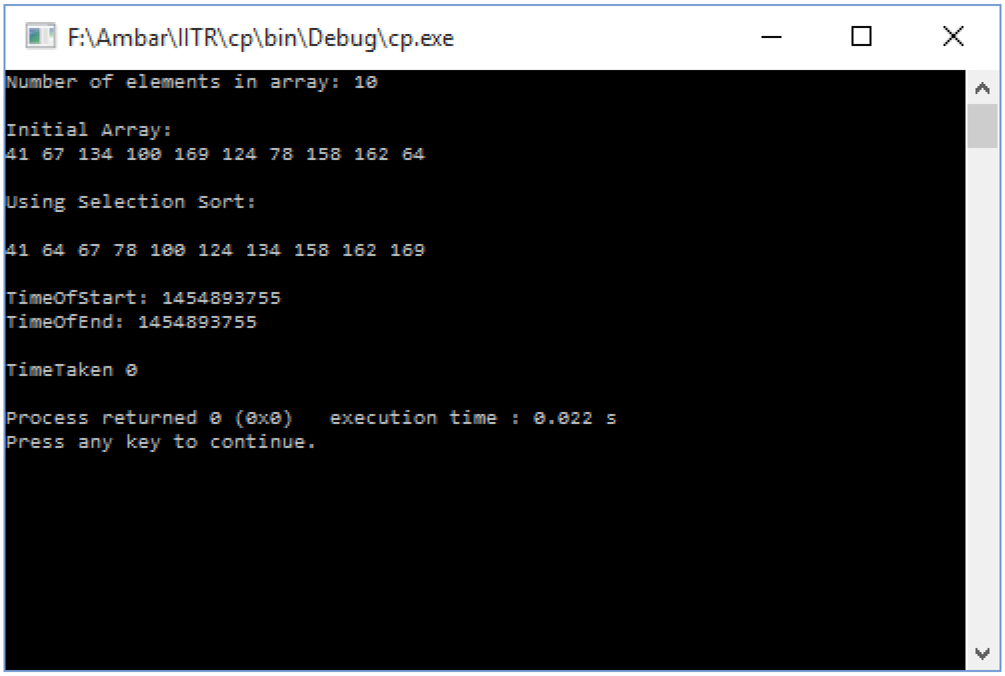


```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Insertion Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454895396
TimeOfEnd: 1454895396

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

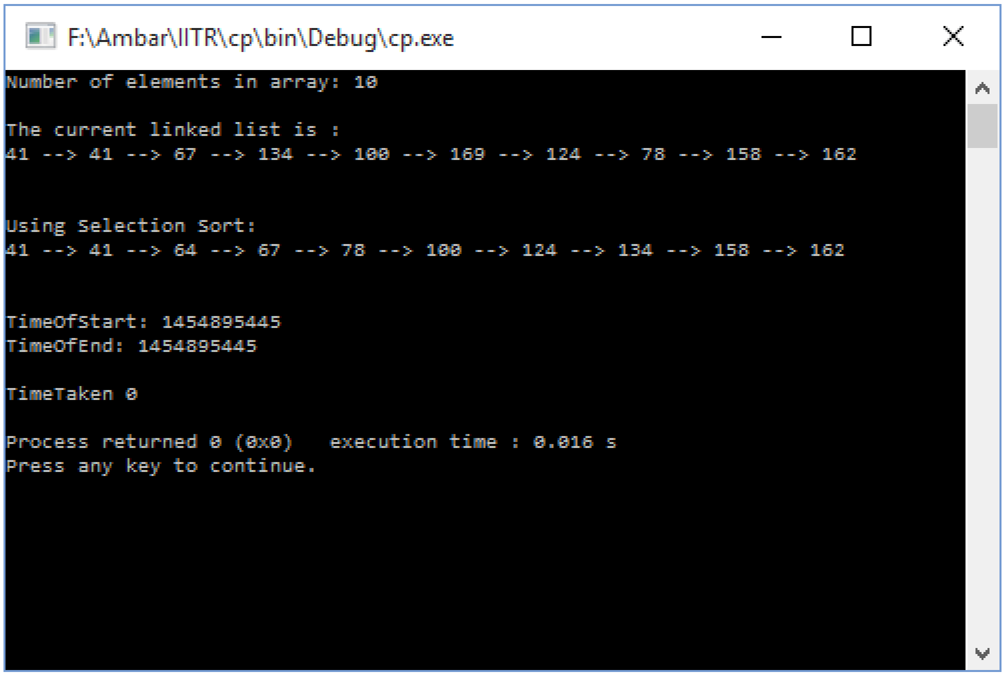
Selection Sort

```
///////////////////////////////////////////////////////////////// SELECTION SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  selectionSortarray.h - Heap Sort for input type array

///////////////////////////////////////////////////////////////// SELECTION SORT


void selectionSortArray(int* a, int n)

{

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

    {

        int min=i;

        for(int j=i;j<n;j++)

        {

            if(a[j]<a[min])

            {

                min=j;

            }

        }

        swap(a[i],a[min]);

    }

}
```

```
///////////////////////////////////////////////////////////////// SELECTION SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  selectionSortlist.h - Selection Sort for input type list

///////////////////////////////////////////////////////////////// SELECTION SORT


void selectionSortList(node *head){

    int len=length(head);

    for(int i=len;i>0;i--){

        int mini=posmax(head,i);

        swapNodes(head,mini,i);

    }

}
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Selection Sort:

41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454893755
TimeOfEnd: 1454893755

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.022 s
Press any key to continue.
```

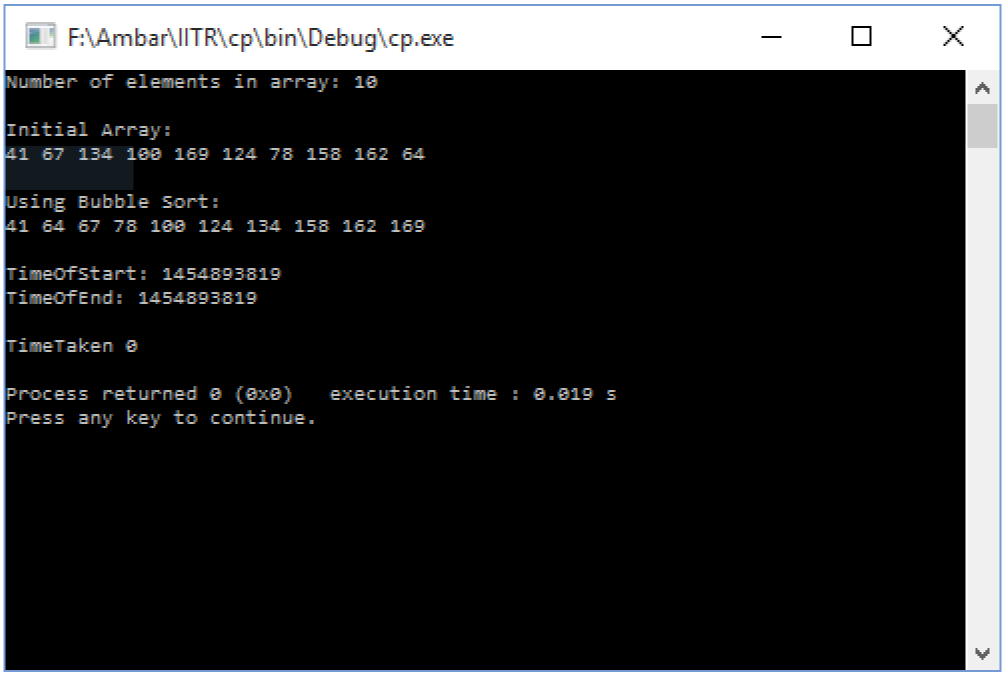


```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Selection Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454895445
TimeOfEnd: 1454895445

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

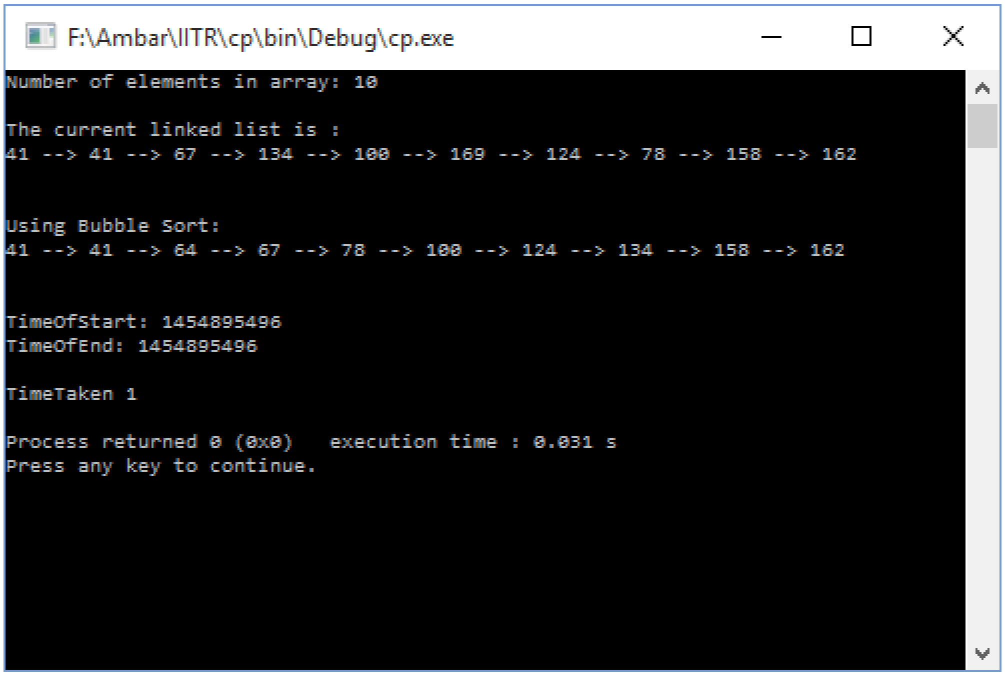
Bubble Sort

```
//////////////////////////////////////////////////////////////////////////////// BUBBLE SORT
//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra
//  Date: February 7,2016
//  bubbleSortarray.h - Bubble Sort for input type array
//////////////////////////////////////////////////////////////////////////////// BUBBLE SORT
void bubbleSortArray(int a[],int n){
    for(int i=n-1;i>0;i--){
        for(int j=0;j<i;j++){
            if(a[j]>a[j+1]){
                int temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
}
```

```
//////////////////////////////////////////////////////////////// BUBBLE SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  bubbleSortlist.h - Bubble Sort for input type list

//////////////////////////////////////////////////////////////// BUBBLE SORT

void bubbleSortList(node * head){

    int len=length(head);

    if(len==1)return ;

    node* cur=head->next;

//    node* next1=cur->next;

    for(int i=len-1;i>0;i--){

        node* cur=head->next;

        node* next1=cur->next;

        int temp=1;

        while(temp<=i){

            if(cur->data > next1->data ){

                int xyz=cur->data;

                cur->data=next1->data;

                next1->data=xyz;

            }

            temp++;

            cur=cur->next;

            next1=next1->next;

        }   }   }
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Bubble Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454893819
TimeOfEnd: 1454893819

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```

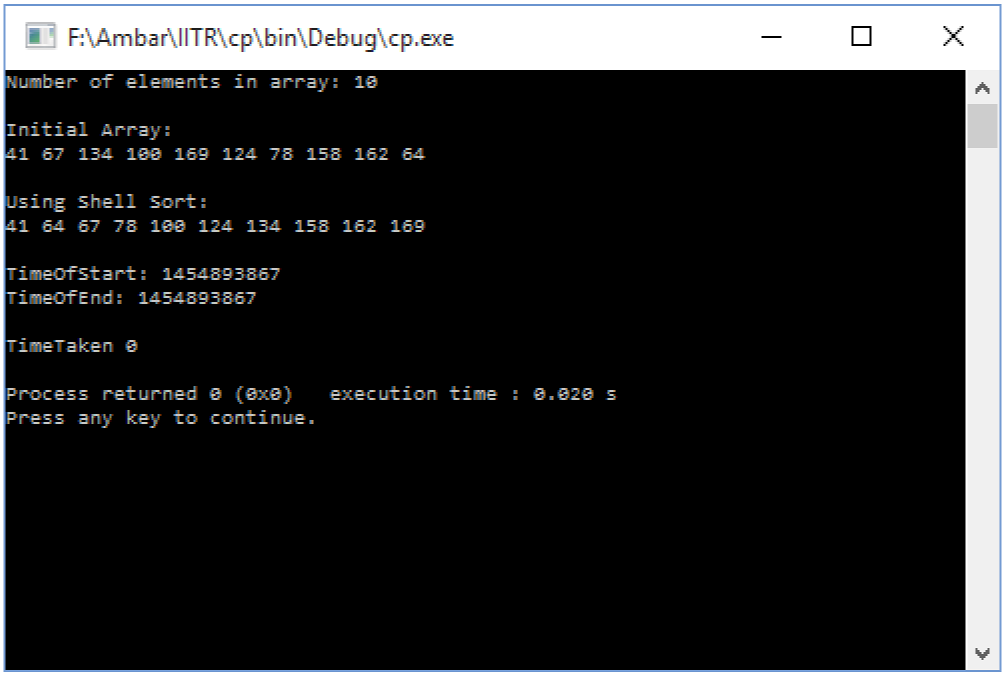


```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Bubble Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454895496
TimeOfEnd: 1454895496

TimeTaken 1

Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

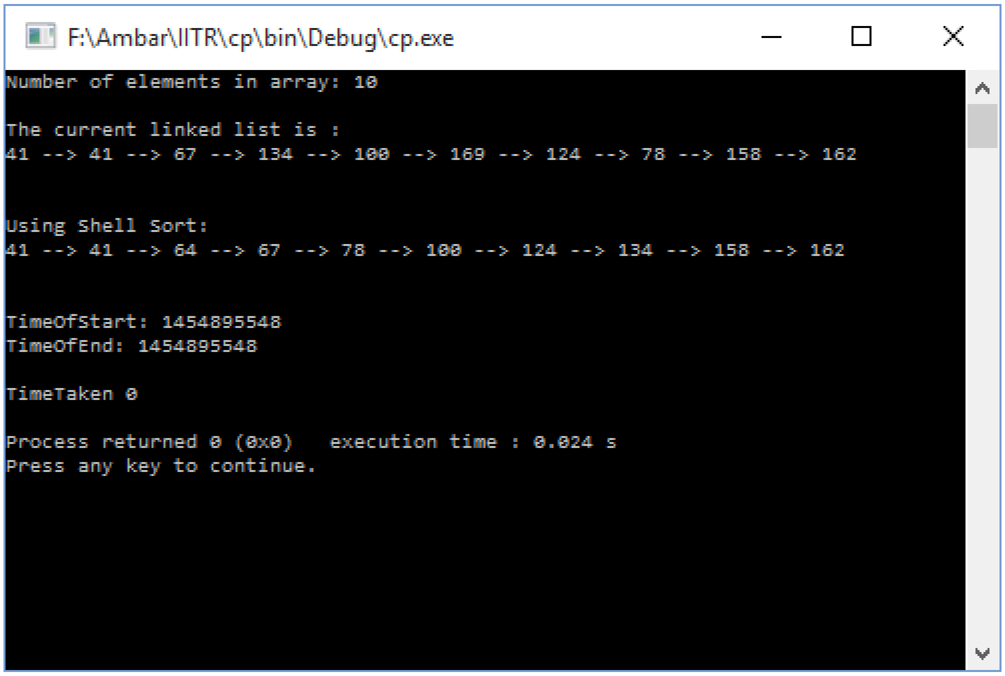
Shell Sort

```
///////////////////////////////////////////////////////////////// SHELL SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  shellSortarray.h - Shell Sort for input type array

///////////////////////////////////////////////////////////////// SHELL SORT


void insertionGapSort(int* a, int n,int gap)

{

        for(int k=0;k<gap;k++)

        {

                for (int i = gap+k; i < n; i+=gap)

     {

      int j = i;

      while (j >= gap && a[j - gap] > a[j] && j<n)

      {

        swap(a[j],a[j-gap]);

        j-=gap;

      }

     }

        }

}

void shellSortArray(int* a, int n)

{
    int gap = 1;
    while(gap<=n/3)
    {
        gap=gap*3+1;
    }
    while(gap>=1)
    {
        insertionGapSort(a,n,gap);
        gap=gap/2;
        //println(a,n);
    }
}
```

```
////////////////////////////////////////////////////////////////// SHELL SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  shellSortlist.h - Shell Sort for input type list

////////////////////////////////////////////////////////////////// SHELL SORT


void insertionGapSort(node* head,int n,int gap){

    node** abc=new node*[gap];

    for(int i=0;i<gap;i++)abc[i]=createlist();

    node* cur=head->next;

    int i=0;

    while(cur){

        addnode(abc[i%gap],cur->data);

        cur=cur->next;

        i++;

    }

    for(int i=0;i<gap;i++){

        insertionSortList(abc[i]);

    }

    cur=head->next;

    for(int i=0;i<gap;i++){

        node* temp=abc[i]->next;

        while(temp){

            cur->data=temp->data;
            cur=cur->next;

            temp=temp->next;

        }

    }

}

void shellSortList(node* a)

{   int n=length(a);

            int gap = 1;

            while(gap<=n/3)

            {

                        gap=gap*3+1;

            }

            while(gap>=1)

            {

                        insertionGapSort(a,n,gap);

                        gap=gap/2;

                        //println(a,n);

            }

}
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Shell Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454893867
TimeOfEnd: 1454893867

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.020 s
Press any key to continue.
```



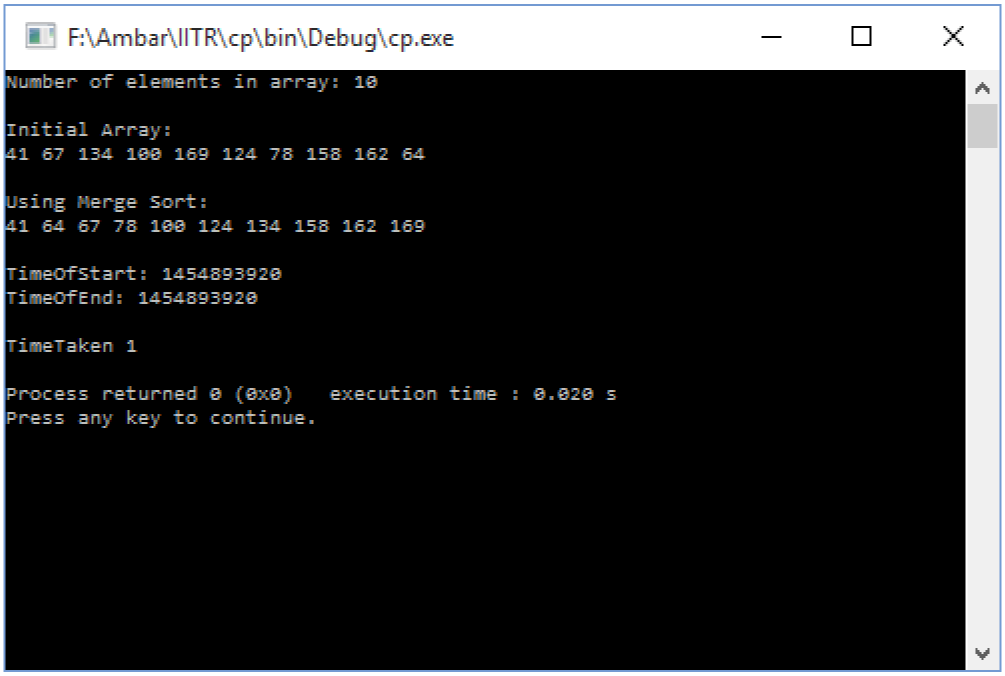
```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Shell Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454895548
TimeOfEnd: 1454895548

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.024 s
Press any key to continue.
```

Merge Sort

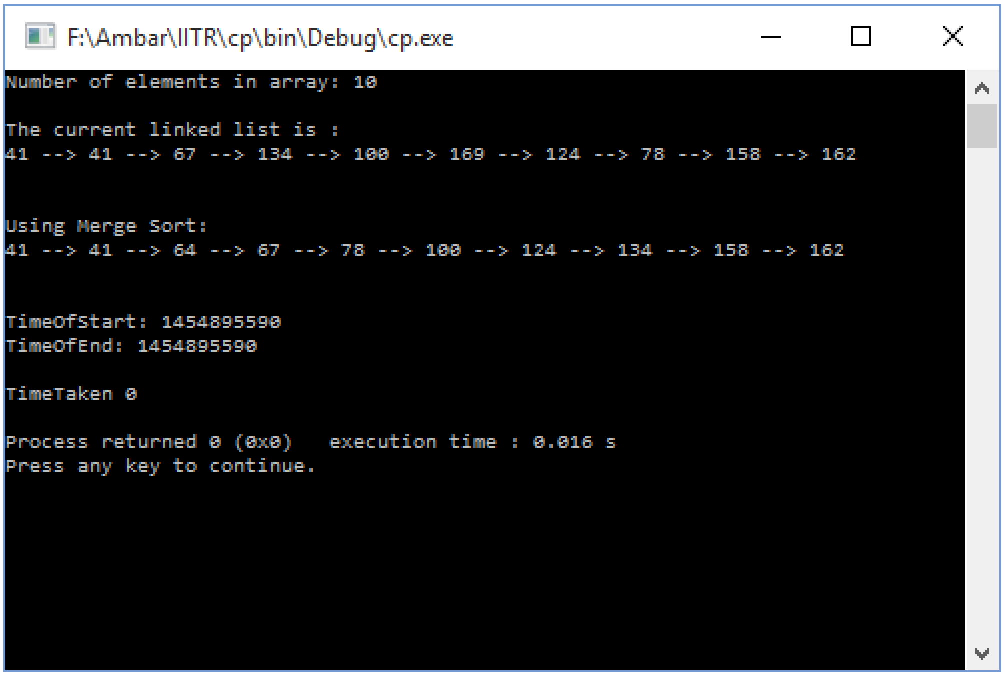
```
//////////////////////////////////////////////////////////////// MERGE SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  mergeSortarray.h - Merge Sort for input type array

//////////////////////////////////////////////////////////////// MERGE SORT


void mergeSort(int a[],int start,int ending){

    //cout<<start<<" "<<ending<<endl;

    if(start>=ending)return ;

    int mid=(start+ending)/2;

    mergeSort(a,start,mid);

    mergeSort(a,mid + 1,ending);

        //cout<<start<<" "<<mid<<" "<<ending<<endl;

    int temp1=start;

    int temp2=mid+1;

    int temp3=start;

    int c[10000];

    while(temp1<=mid && temp2<=ending){

        if(a[temp1]<a[temp2]){

            c[temp3]=a[temp1];

            temp1++;

            temp3++;

          //  cout<<1<<endl;
    }
    else {
        c[temp3]=a[temp2];
        temp2++;
        temp3++;
        //cout<<2<<endl;
    }
}
while(true){

    if(temp1==mid+1)break;
    //cout<<3<<endl;
    c[temp3]=a[temp1];
    temp3++;
    temp1++;
}
while(true){
    if(temp2==ending+1)break;
    c[temp3]=a[temp2];
    temp2++;
    temp3++;
}
```

```
    for(int i=start;i<=ending;i++)a[i]=c[i];
}
void mergeSortArray(int a[],int n){
    mergeSort(a,0,n-1);
}
///////////////////////////////////////////////////////////////// MERGE SORT
//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra
//  Date: February 7,2016
//  mergeSortlist.h - Merge Sort for input type list
///////////////////////////////////////////////////////////////// MERGE SORT


node* mergingToLinkedList(node* head1, node* head2){
    node* ans= NULL;
    if (head1==NULL)
        return head2;
    else if (head2==NULL)
        return head1;
    if (head1->data <= head2->data){
        ans= head1;
        ans->next= mergingToLinkedList(head1->next, head2);
    }else{
        ans= head2;
        ans->next= mergingToLinkedList(head1, head2->next);
    }
    return ans;
}
void split(node* head, node** start, node** ending){
    node* temp1;
    node* temp2;
    if (head==NULL||head->next==NULL){
        *start = head;
        *ending = NULL;
    }
    else{
        temp2 = head;
        temp1 = head->next;
        while (temp1 != NULL){
            temp1 = temp1->next;
            if (temp1!=NULL){
                temp2= temp2->next;
                temp1= temp1->next;
            }
        }
        *start= head;
        *ending= temp2->next;
        temp2->next= NULL;
    }
}
```

```
void MergeSortl( node** heading){

    node* head= *heading;

    node* temp1;node* temp2;

    if ((head==NULL)||(head->next==NULL))return;

    split(head,&temp1,&temp2);

    MergeSortl(&temp1);

    MergeSortl(&temp2);

    *heading= mergingToLinkedList(temp1,temp2);

}

void mergeSortList(node* head){

    int l=length(head);

    if(l==0||l==1)return ;

    MergeSortl(&(head->next));

}
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Merge Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454893920
TimeOfEnd: 1454893920

TimeTaken 1

Process returned 0 (0x0)   execution time : 0.020 s
Press any key to continue.
```



```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Merge Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454895590
TimeOfEnd: 1454895590

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

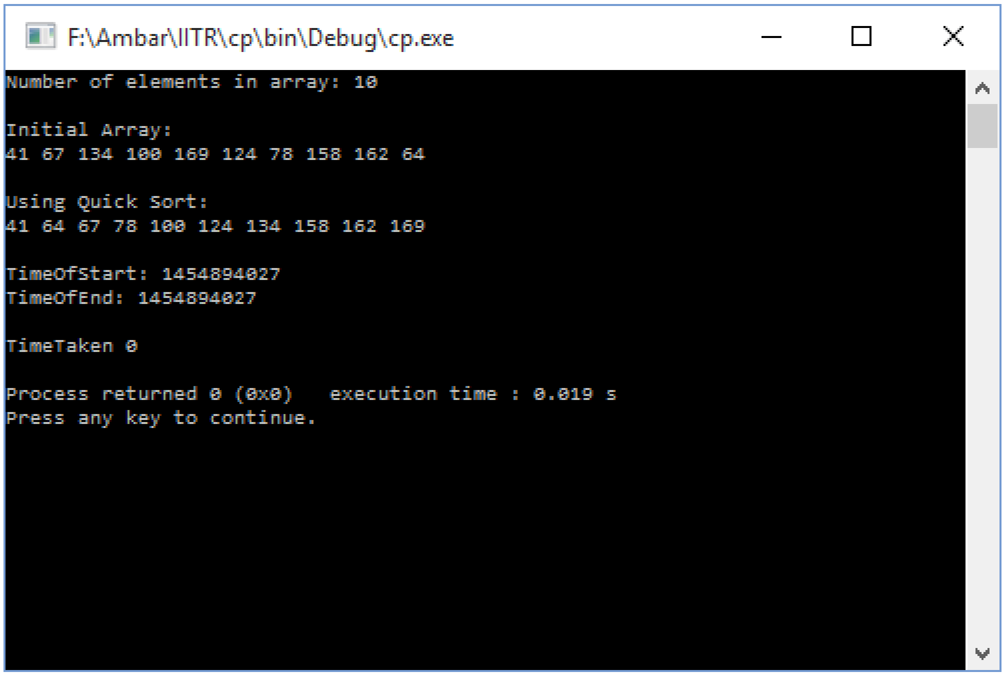
Quick Sort

```
//////////////////////////////////////////////////////////////// QUICK SORT
//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra
//  Date: February 7,2016
//  quickSortarray.h - Quick Sort for input type array
//////////////////////////////////////////////////////////////// QUICK SORT


void quickSort(int* a, int l, int r)
{
    if(r-l<1) return;
//  cout<<"\n"<<a[l]<<" ___||"<<l<<","<<r<<"||___ ";
    int lower = l+1;
    int higher = l+1;
    while(higher<=r)
    {
        if(a[higher]<a[l])
        {
            if(lower!=higher)
            {
//              cout<<"|"<<a[lower]<<"<-->"<<a[higher]<<" ..
"<<lower-l<<","<<higher-l<<"|";
                swap(a[lower],a[higher]);
            }
            lower++;
        }
```

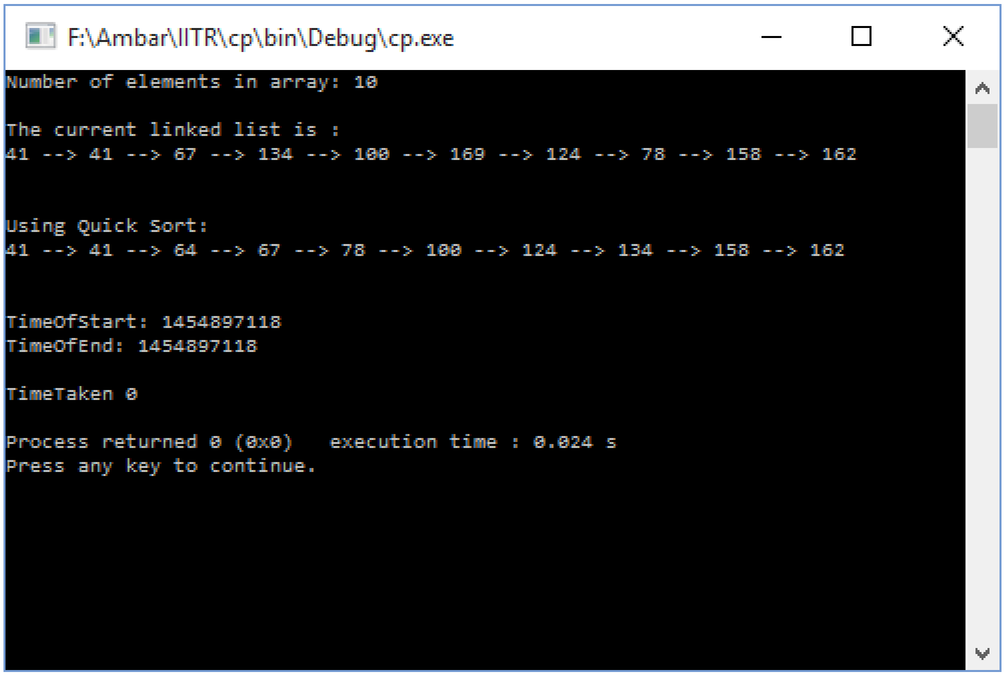
```
			higher++;
		}
//		cout<<"\n ???|"<<a[l]<<"<-->"<<a[lower-1]<<" .. "<<lower-l<<","<<higher-l<<"|";
		swap(a[l],a[lower-1]);
//		println(a,8);
		quickSort(a,l,lower-2);
		quickSort(a,lower,r);
}
void quickSortArray(int *a, int n)
{
    quickSort(a, 0, n-1);
}

//////////////////////////////////////////////////////////////// QUICK SORT
//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra
//  Date: February 7,2016
//  quicksortlist.h - Quick Sort for input type list
//////////////////////////////////////////////////////////////// QUICK SORT

node* split(node* a1, node* r, node*& idx)
{
    node* idx2 = a1;
    idx = NULL;
    int Pivot = r->data;
```

```
node* cur =a1;

    //cout<<Pivot<<endl;

    while(cur!=r && cur)

    {

        if (cur->data <= Pivot)

        {

            if (cur != idx2)

            {

                swap1(cur, idx2);

            }

        idx = idx2;

        idx2 = idx2->next;

        }

        //cout<<idx->data<<endl;

        //cout<<idx2->data<<endl;

        cur = cur->next;

    }

if (idx2 != r)

    {

        swap1(idx2, r);

    }

    //cout<<idx2<<endl

return idx2;
```

```
}


void quicksort(node* a1, node* r)
{
    if (a1==NULL||r== NULL)return;
    if (a1==r)return;
    node* idx= NULL;
    node* idx2= split(a1, r, idx);
    //cout<<idx2<<endl;
    quicksort(a1, idx);
    if (idx2 != r){ quicksort(idx2->next, r);  }
}
void quick_sort(node* head)
{
    if (head == NULL) return;
    node* r = head;
    while (r->next)
    {
        r = r->next;
    }
    quicksort(head, r);
}
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Quick Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454894027
TimeOfEnd: 1454894027

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```

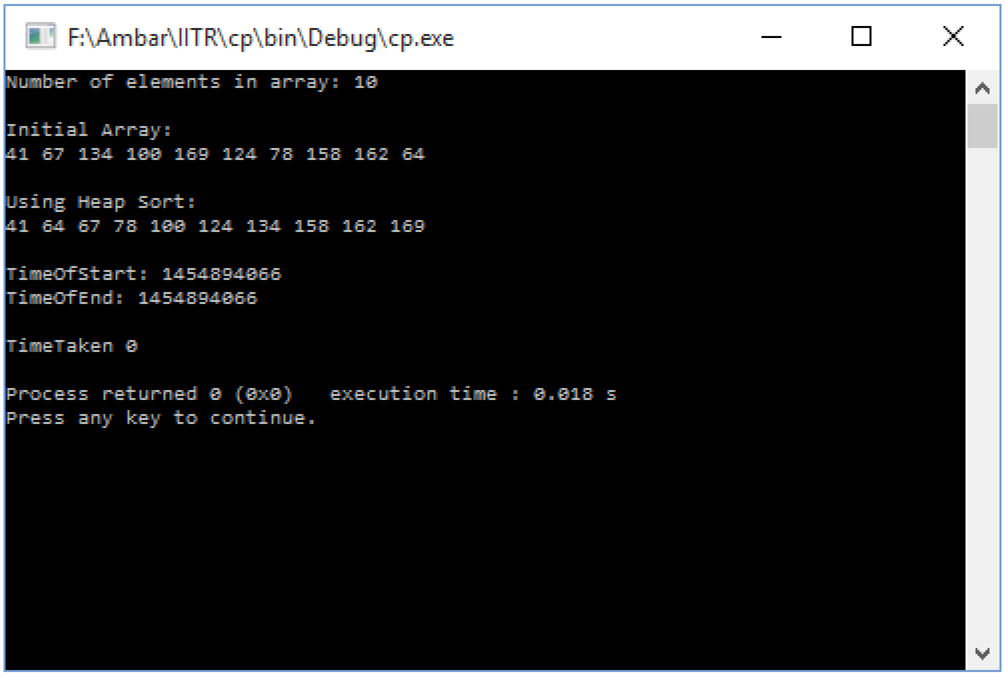


```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Quick Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454897118
TimeOfEnd: 1454897118

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.024 s
Press any key to continue.
```

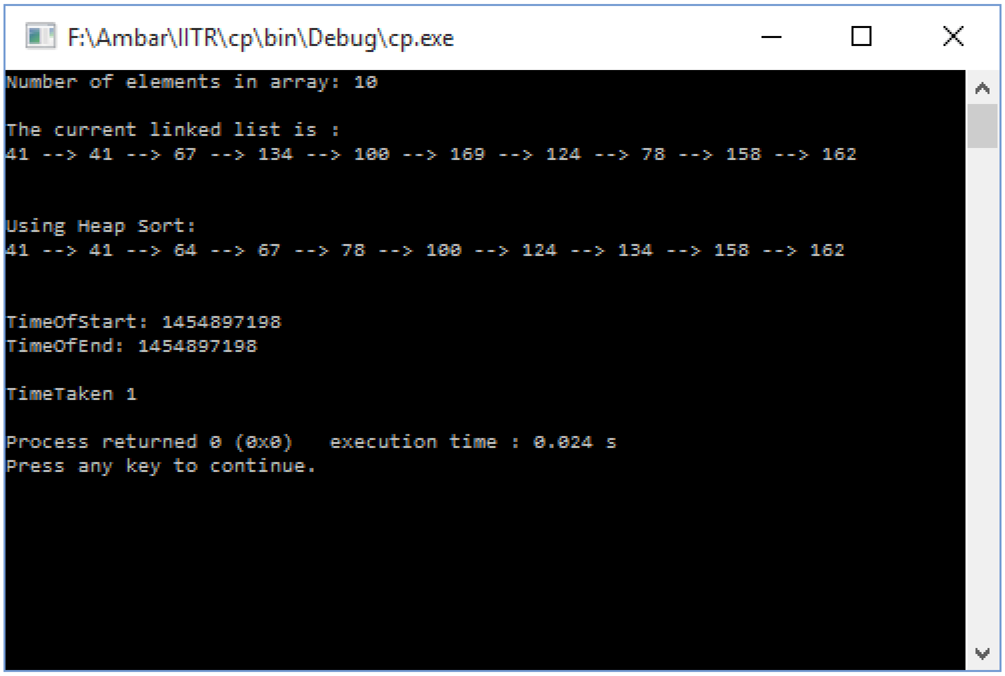
Heap Sort

```
////////////////////////////////////////////////////////////// COUNTING SORT
//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra
//  Date: February 7,2016
//  countingSortlist.h - Counting Sort for input type list
////////////////////////////////////////////////////////////// COUNTING SORT


// assuming numbers are between 0 and 10000 both inclusive


void countingSortList(node* head){
    int temp[10001];
    for(int i=0;i<10001;i++)temp[i]=0;
    node* cur=head->next;
    while(cur){
        temp[cur->data]++;
        cur=cur->next;
    }
    cur=head->next;
    for(int i=0;i<10001;i++){
        while(temp[i]){
            cur->data=i;
            cur=cur->next;
            temp[i]--;
        }
```

```
  }
}
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Heap Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454894066
TimeOfEnd: 1454894066

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.018 s
Press any key to continue.
```



```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Heap Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454897198
TimeOfEnd: 1454897198

TimeTaken 1

Process returned 0 (0x0)   execution time : 0.024 s
Press any key to continue.
```

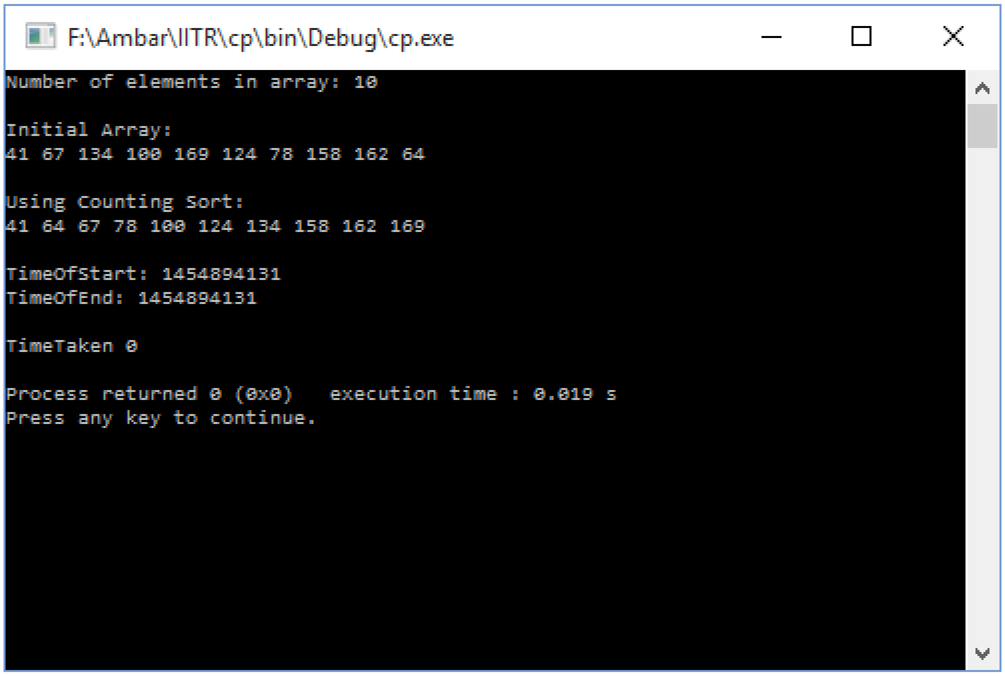
Counting Sort

```
//////////////////////////////////////////////////////////////// COUNTING SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  countingSortarray.h - Counting Sort for input type array

//////////////////////////////////////////////////////////////// COUNTING SORT


/* assuming all non-negative integer elements with maximum value k.*/


void countingSortArray(int* a, int n)

{

  int k = maxElem(a,n);

        int* b = new int[n];

        int* count = new int[k+1];


        for(int i=0;i<=k;i++)

        {

                count[i]=0;

        }  //   println(a,n);

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

        {

                count[a[i]]++;

        }  //   println(count,k+1);

        for(int i=1;i<k+1;i++)
```

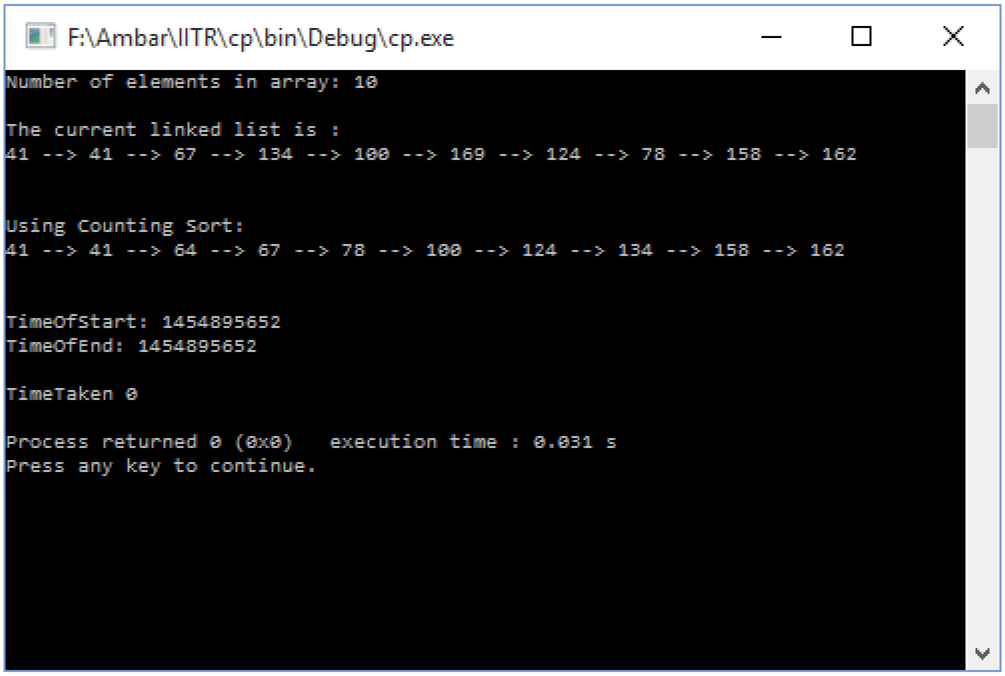
```
    {
        count[i]+=count[i-1];
    }  //   println(count,k+1);
    for(int i=0;i<n;i++)
    {
        b[count[a[i]]-1]= a[i];
        count[a[i]]--;
    }  //   println(b,7);
    for(int i=0;i<n;i++)
    {
        a[i]=b[i];
    }
}
void countingSortArray(int* a, int n, int k)
{
    int* b = new int[n];
    int* count = new int[k+1];

    for(int i=0;i<=k;i++)
    {
        count[i]=0;
    }  //   println(a,n);
    for(int i=0;i<n;i++)
    {
```

```
                count[a[i]]++;
        }  //   println(count,k+1);
        for(int i=1;i<k+1;i++)
        {
                count[i]+=count[i-1];
        }  //   println(count,k+1);
        for(int i=0;i<n;i++)
        {
                b[count[a[i]]-1]= a[i];
                count[a[i]]--;
        }  //   println(b,7);
        for(int i=0;i<n;i++)
        {
                a[i]=b[i];
        }
}
```

```
////////////////////////////////////////////////////////////// COUNTING SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  countingSortlist.h - Counting Sort for input type list

////////////////////////////////////////////////////////////// COUNTING SORT


// assuming numbers are between 0 and 10000 both inclusive

void countingSortList(node* head){

    int temp[10001];

    for(int i=0;i<10001;i++)temp[i]=0;

    node* cur=head->next;

    while(cur){

        temp[cur->data]++;

        cur=cur->next;

    }

    cur=head->next;

    for(int i=0;i<10001;i++){

        while(temp[i]){

            cur->data=i;

            cur=cur->next;

            temp[i]--;

        }

    }

}
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Counting Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454894131
TimeOfEnd: 1454894131

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```



```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Counting Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454895652
TimeOfEnd: 1454895652

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

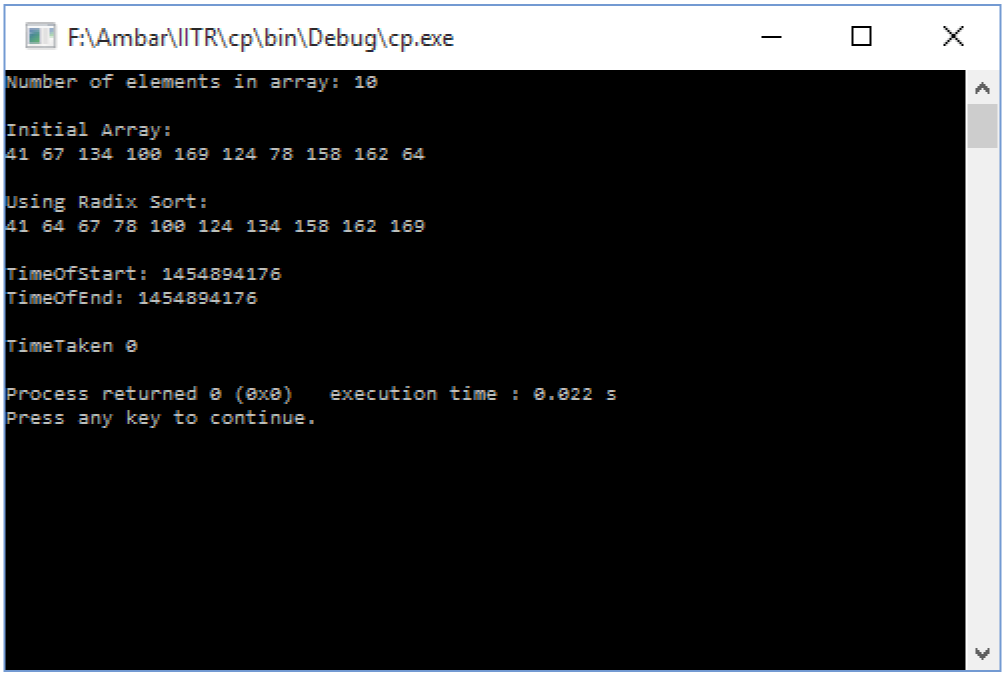
Radix Sort

```
///////////////////////////////////////////////////////////////// RADIX SORT
//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra
//  Date: February 7,2016
//  radixSortarray.h - Radix Sort for input type array
///////////////////////////////////////////////////////////////// RADIX SORT

void radixSortArray(int* a,int n)
{
  int max = maxElem(a,n);
  int maxD = nDigits(max);

  node* buckets[10];
  for(int i=0;i<10;i++)
    buckets[i]=createlist();

  int k= 0;
  while(k<maxD)
  {
    for(int i=0;i<n;i++)
    {
      addnode(buckets[digit(a[i],k)],a[i]);
    }
    //printlist(buckets[2]);
    for(int i=0;i<10;i++)
    {
      //  cout<<length(buckets[i])<<" "<<i<<endl;
       // insertionSortList(buckets[i]);
    }
//cout<<1;
    int abcd=0;
    for(int i=0;i<10;i++)
    {
        node* temp=buckets[i]->next;
        //cout<<"\n"<<i<<" || ";
        while(temp)
        {
            a[abcd]=temp->data;
       //      cout<<temp->data<<", ";
            temp=temp->next;
            abcd++;
        }
    }//cout<<"\n";
 //   for(int i=0;i<n;i++)
 //   {
 //       cout<<" "<<a[i]<<" ";
 //   }
 //   cout<<'\n';
```

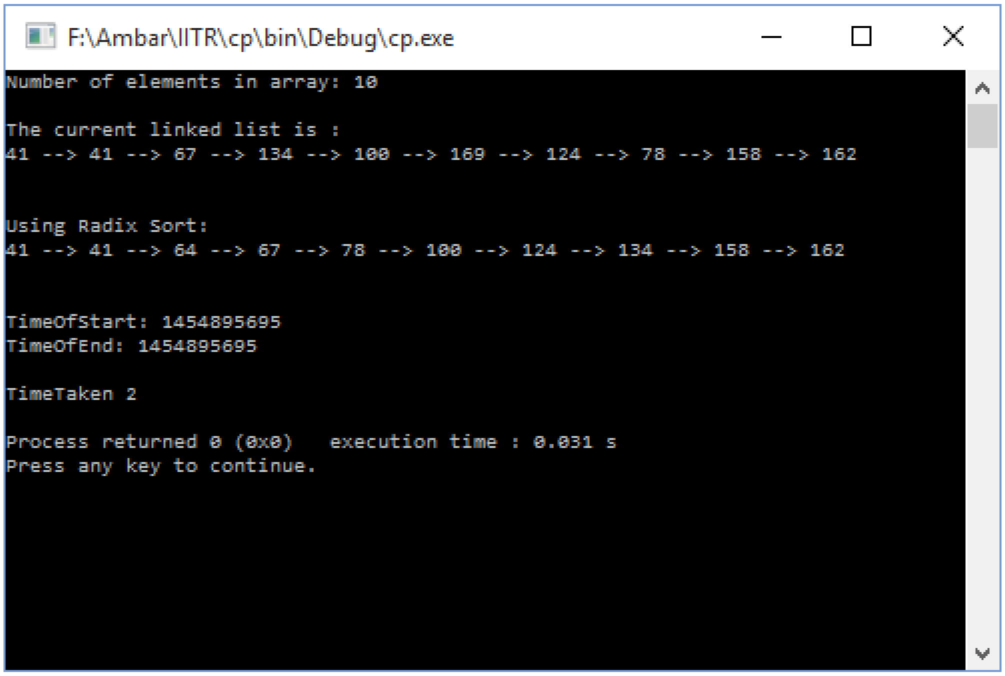
```
    for(int i=0;i<10;i++){
        buckets[i]->next=NULL;
    }
    k++;
  }
}

///////////////////////////////////////////////////////////////// RADIX SORT
//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra
//  Date: February 7,2016
//  radixSortlist.h - Radix Sort for input type list
///////////////////////////////////////////////////////////////// RADIX SORT

void radixSortList(node* head)
{
  //  printlist(head);
  node* cur=head->next;
  int max=cur->data;
  while(cur){
    if(max < cur->data)max=cur->data;
    cur=cur->next;
  }
  int maxD = nDigits(max);
```

```
node* buckets[10];
for(int i=0;i<10;i++)
  buckets[i]=createlist();

 // cout<<max<<" "<<maxD<<" pppp "<<digit(999,2)<<"\n";
int k= 0;
while(k<maxD)
{
     //cout<<" "<<k<<" pppp \n";

  node* cur=head->next;
  while(cur){
      //cout<<cur->data<<" "<<digit(cur->data,k)<<", ";
      addnode(buckets[digit(cur->data,k)],cur->data);
      cur=cur->next;
  }

  cur=head->next;
  for(int i=0;i<10;i++)
  {
      node* temp=buckets[i]->next;
      //cout<<"\n"<<i<<" || ";
      while(temp)
      {
```

```
      cur->data=temp->data;

      // cout<<temp->data<<", ";

      temp=temp->next;

      cur=cur->next;

    }

  }//cout<<"\n";

  //printlist(head);

  //cout<<'\n';

  for(int i=0;i<10;i++){

    buckets[i]->next=NULL;

  }

  k++;

 }

}
```



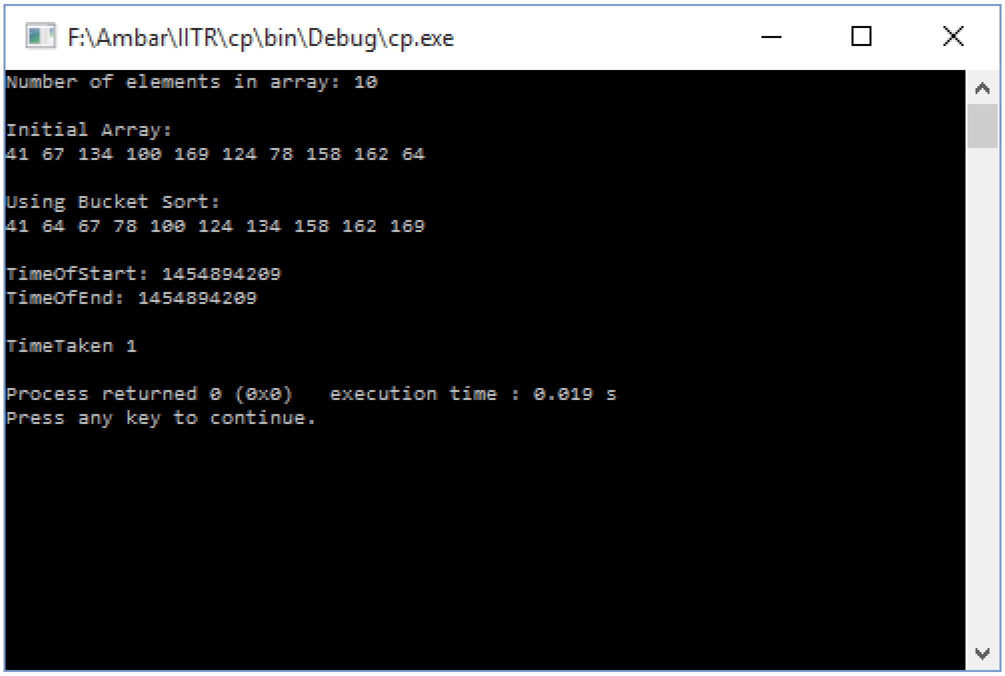
```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Radix Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454894176
TimeOfEnd: 1454894176

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.022 s
Press any key to continue.
```



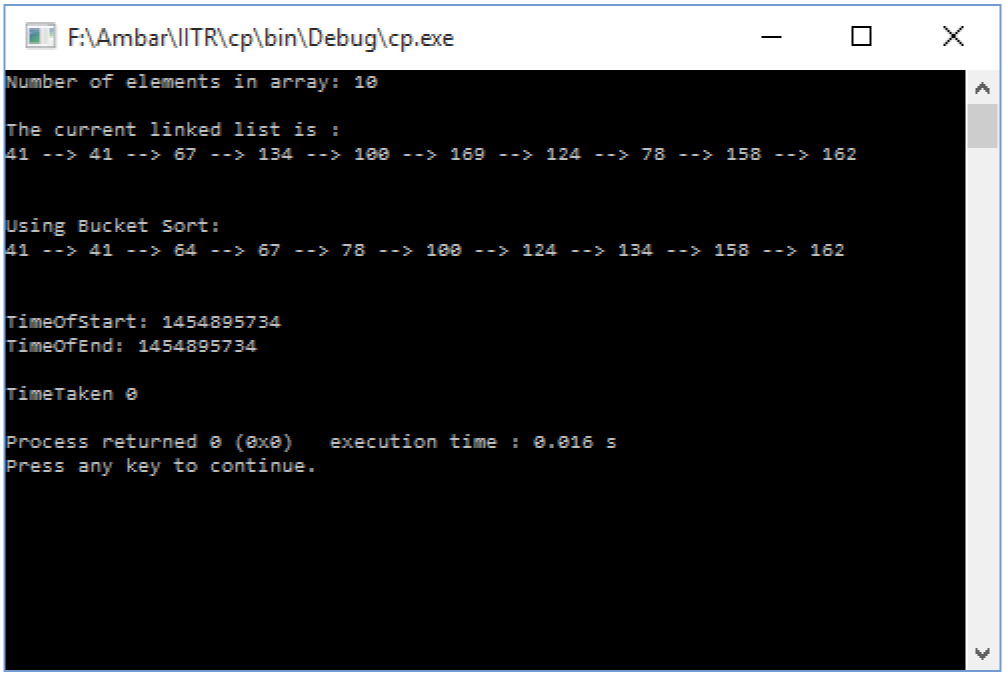
```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Radix Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454895695
TimeOfEnd: 1454895695

TimeTaken 2

Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

Bucket Sort

```
/////////////////////////////////////////////////////////////// BUCKET SORT

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  bucketSortarray.h - Bucket Sort for input type array

/////////////////////////////////////////////////////////////// BUCKET SORT


void bucketSortArray(int a[],int n)

{

    node* temp[1000];

    for(int i=0;i<1000;i++)

        temp[i]=createlist();


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

        addnode(temp[a[i]/10],a[i]);




    for(int i=0;i<1000;i++){


        insertionSortList(temp[i]);

       }

       int j=0;

    for(int i=0;i<1000;i++){
        node* cur=temp[i]->next;
        while(cur){
           // cout<<j<<" "<<i<<endl;
            a[j]=cur->data;
            cur=cur->next;
            j++;
        }
    }
}
////////////////////////////////////////////////////////////// BUCKET SORT
//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra
//  Date: February 7,2016
//  bucketSortlist.h - Bucket Sort for input type list
////////////////////////////////////////////////////////////// BUCKET SORT

//assume numbers are between 0 and 10000 both inclusive

void bucketSortList(node* head){
    node* a[1000];
    for(int i=0;i<1000;i++)a[i]=createlist();
    //cout<<1<<endl;
    node* cur=head->next;
    while(cur){
        addnode(a[(cur->data)/10],cur->data);
        cur=cur->next;
    }
    //cout<<2<<endl;
    for(int i=0;i<1000;i++){
      //  cout<<i<<endl;
            insertionSortList(a[i]);
    }
    node* temp2=head->next;
    for(int i=0;i<1000;i++){
        node* temp=a[i]->next;
        while(temp){
            temp2->data=temp->data;
            temp=temp->next;
            temp2=temp2->next;
        }
    }
}
```



```
Number of elements in array: 10

Initial Array:
41 67 134 100 169 124 78 158 162 64

Using Bucket Sort:
41 64 67 78 100 124 134 158 162 169

TimeOfStart: 1454894209
TimeOfEnd: 1454894209

TimeTaken 1

Process returned 0 (0x0)   execution time : 0.019 s
Press any key to continue.
```



```
Number of elements in array: 10

The current linked list is :
41 --> 41 --> 67 --> 134 --> 100 --> 169 --> 124 --> 78 --> 158 --> 162


Using Bucket Sort:
41 --> 41 --> 64 --> 67 --> 78 --> 100 --> 124 --> 134 --> 158 --> 162


TimeOfStart: 1454895734
TimeOfEnd: 1454895734

TimeTaken 0

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

UTILITY FILE

```
//////////////////////////////////////////////////////////////// COMMON UTILITIES

//  Group 18 - 14114009_14114006 - Ambar Zaidi & Akshit Kalra

//  Date: February 7,2016

//  utilities.h - Utility Functions

//////////////////////////////////////////////////////////////// COMMON UTILITIES

void swap(int& a,int& b)

{

        int temp = a;

        a=b;

        b=temp;

}

void print(int* a,int n)

{

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

        {

                cout<<a[i]<<" ";

        }

}

void println(int*a, int n)

{

        cout<<"\n ---- ";

        print(a,n);

}
```

```
int digit(int m, int k)

{

    while(k--)

    {

        m/=10;

    }

    m%=10;

    return m;

}

int maxElem(int*a, int n)

{

            int mx=0;

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

            {

                        if(a[i]>mx)mx=a[i];

            }

            return mx;

}

int nDigits(int m)

{

  int l=1;

  while(m/=10)

  {

    l++;
  }

  return l;

}

void randomArray(int* a, int n)

{

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

    {

        a[i]=rand()%200;

    }

}
```