

Heaven's Light is Our Guide



Rajshahi University of Engineering and Technology

Department of Computer Science and Engineering

Course No: CSE.2202

Course Title: Sessional based on CSE.2201 (Computer Algorithms)

Lab Report No: 05

Lab Report On: Sorting in linear time: Counting Sort.

Submitted By

Md. Ariful Islam

Roll No: 1803046

Section: A

Department: CSE

Submitted To

Biprodip Pal

Assistant Professor

Dept. of CSE, RUET

Date: 26-06-2021

❖ **Problem Statement:** The problem is observe sorting in linear time with counting sort. That requires

- i. To generate **N** random integers within range 0 to 10000 in a file named **input.txt**.
- ii. To implement bubble sort to sort the numbers (from input.txt) and count the time.
- iii. To implement counting sort to sort the numbers (from input.txt) and count the time.
- iv. To increase the value of N and to plot the performance curve for sufficiently large N to see distinguishable performance.

❖ **Implemented Code:**

```
#include<bits/stdc++.h>
using namespace std;
using namespace std::chrono;

typedef long long ll;
#define M 10001

void menu(){
    cout<<"\nEnter N (Press 0 to Exit): ";
}

int main(){
    ll n,i,j;
    vector<ll>cn,cb,cc;

    while(1){
        ll a;
        menu();
        cin>>a;

        if(a<0) {
            cout<<"Invalid Input"<<endl;
```

```
        continue;
    }

    if(a==0){
        cout<<"\nExiting..."<<endl;
        break;
    }

    ll mx,mn,x;

    vector<ll>bsort_array,pos_csort,csort_array;
    map<ll,ll>mp;

//Creating File

    ofstream f1;
    ifstream f2;
    f1.open("input.txt");

    n=a;
    cn.push_back(n);

    srand(time(0));
    x=rand()%M;
    mx=x;
    mn=x;
    f1<<x;
    csort_array.push_back(-1);

    for(i=1;i<n;i++){
        x=rand()%M;
        f1<<" ";
        f1<<x;
        mx=max(mx,x);
        mn=min(mn,x);
        csort_array.push_back(-1);
    }
    f1.close();

//Counting time for bubble sort

    auto start = high_resolution_clock::now();
```

```
f2.open("input.txt");

while(!f2.eof()){
    f2>>x;
    bsort_array.push_back(x);
}
f2.close();
```

// Bubble sort

```
for(i=0;i<n-1;i++){
    for(j=i+1;j<n;j++){
        if(bsort_array[i]>bsort_array[j]){
            swap(bsort_array[i],bsort_array[j]);
        }
    }
}
```

```
auto stop = high_resolution_clock::now();
```

```
auto duration = duration_cast<milliseconds>(stop - start);
cout<<"Bubble Sort: "<<duration.count()<<" Milliseconds"<<endl;
cb.push_back(duration.count());
```

//Counting time for counting sort

```
start = high_resolution_clock::now();
```

```
f2.open("input.txt");
```

//Counting sort

```
while(!f2.eof()){
    f2>>x;
    mp[x]+=1;
}
f2.close();
```

```

        for(i=mn;i<=mx;i++){
            if(i==mn){
                pos_csort.push_back(mp[i]);
                continue;
            }
            pos_csort.push_back(mp[i]+pos_csort[i-mn-1]);
        }

        f2.open("input.txt");
        while(!f2.eof()){
            f2>>x;
            csort_array[pos_csort[x-mn]-1]=x;
            pos_csort[x-mn]-=1;
        }

        stop = high_resolution_clock::now();

        duration = duration_cast<milliseconds>(stop - start);
        cout<<"Counting Sort: "<<duration.count()<<" Milliseconds"<<endl;
        cc.push_back(duration.count());

        f2.close();
    }

    for(i=0;i<cn.size()-1;i++){
        for(j=i+1;j<cn.size();j++){
            if(cn[i]>cn[j]){
                swap(cn[i],cn[j]);
                swap(cb[i],cb[j]);
                swap(cc[i],cc[j]);
            }
        }
    }

    cout<<"\nN\tB_S\tC_S"<<endl;
    for(i=0;i<cn.size();i++){
        cout<<cn[i]<<"\t"<<cb[i]<<"\t"<<cc[i]<<endl;
    }

    return 0;
}

```

❖ Output:

"F:\4th Semester\CSE\CSE.2202\Lab 7\1803046.exe"

Enter N (Press 0 to Exit): 100

Bubble Sort: 8 Milliseconds

Counting Sort: 16 Milliseconds

Enter N (Press 0 to Exit): 400

Bubble Sort: 13 Milliseconds

Counting Sort: 19 Milliseconds

Enter N (Press 0 to Exit): 700

Bubble Sort: 22 Milliseconds

Counting Sort: 21 Milliseconds

Enter N (Press 0 to Exit): 1000

Bubble Sort: 27 Milliseconds

Counting Sort: 16 Milliseconds

Enter N (Press 0 to Exit): 1300

Bubble Sort: 33 Milliseconds

Counting Sort: 14 Milliseconds

Enter N (Press 0 to Exit): 1600

Bubble Sort: 42 Milliseconds

Counting Sort: 15 Milliseconds

Enter N (Press 0 to Exit): 1900

Bubble Sort: 40 Milliseconds

Counting Sort: 15 Milliseconds

Enter N (Press 0 to Exit): 2200

Bubble Sort: 59 Milliseconds

Counting Sort: 15 Milliseconds

Enter N (Press 0 to Exit): 2500

Bubble Sort: 68 Milliseconds

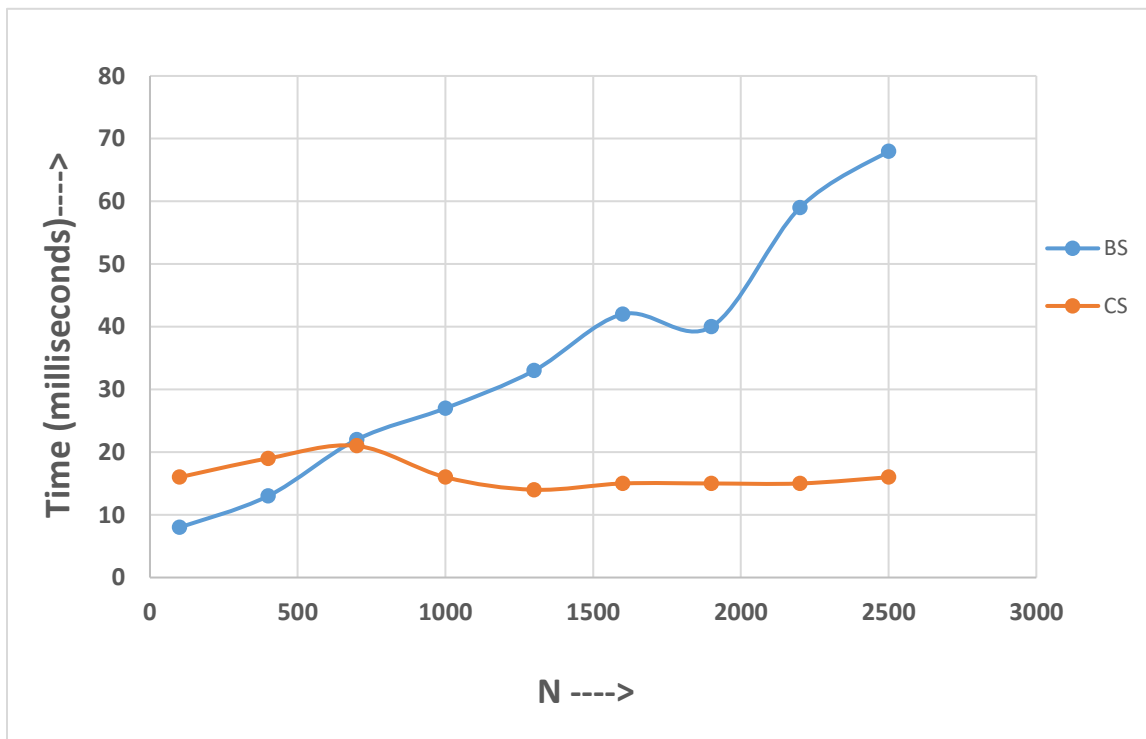
Counting Sort: 16 Milliseconds

Enter N (Press 0 to Exit): 0

Exiting...

N	B_S	C_S
100	8	16
400	13	19
700	22	21
1000	27	16
1300	33	14
1600	42	15
1900	40	15
2200	59	15
2500	68	16

❖ Performance Curve of Bubble Sort & Counting Sort:



❖ Discussion & Conclusion: From the output and graph we saw that for little value of **N** the sorting time of bubble sort was lower than counting sort. But as the value of **N** increases, the sorting time for bubble sort increases more than counting sort. For the higher value of **N** the Counting sort runs faster than the Bubble sort.

END