**Dr. B.R. Ambedkar National Institute of Technology, Jalandhar**

****

**System Programming**

**( Jan/2020- May/2020 )**

**Submitted to: Name : Arshdeep**

**Roll No: 17103014**

**Lab -2**

**Bucket Sort:-**

#include <iostream>

#include <algorithm>

#include <vector>

using namespace std;

void bucketSort(vector<float> &arr)

{

vector<float> b[arr.size()];

for (int i=0; i< arr.size(); i++)

{

int bi = arr.size()\*arr[i];

b[bi].push\_back(arr[i]);

}

for (int i=0; i<arr.size(); i++)

sort(b[i].begin(), b[i].end());

int index = 0;

for (int i = 0; i < arr.size(); i++){

for (int j = 0; j < b[i].size(); j++)

arr[index++] = b[i][j];

b[i].clear();

}

}

int main()

{

vector<float> arr= {0.897, 0.565, 0.656, 0.1234, 0.665, 0.3434};

bucketSort(arr);

cout<<"\nArshdeep 17103014\n\n\n\n";

cout << "Sorted array is \n";

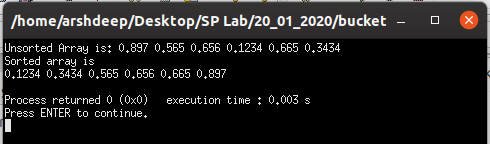
for (int i=0; i<arr.size(); i++)

cout << arr[i] << " ";

cout<<endl;

return 0;

}



**Counting Sort:-**

//Counting sort which takes negative numbers as well

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int getMax(vector<int> arr)

{

int mx = arr[0];

for (int i = 1; i < arr.size(); i++)

if (arr[i] > mx)

mx = arr[i];

return mx;

}

int getMin(vector<int> arr)

{

int mn = arr[0];

for (int i = 1; i < arr.size(); i++)

if (arr[i] < mn)

mn = arr[i];

return mn;

}

void countSort(vector <int>& arr)

{

int max = getMax(arr);

int min = getMin(arr);

int range = max - min + 1;

vector<int> count(range), output(arr.size());

for(int i = 0; i < arr.size(); i++)

count[arr[i]-min]++;

for(int i = 1; i < count.size(); i++)

count[i] += count[i-1];

for(int i = arr.size()-1; i >= 0; i--)

{

output[ count[arr[i]-min] -1 ] = arr[i];

count[arr[i]-min]--;

}

for(int i=0; i < arr.size(); i++)

arr[i] = output[i];

}

void printArray(vector <int> & arr)

{

cout<<"\nArshdeep 17103014\n\n\n\n";

cout<<"Sorted Array is: ";

for (int i=0; i < arr.size(); i++)

cout << arr[i] << " ";

cout << "\n";

}

int main()

{

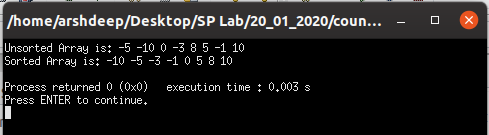
vector<int> arr = {-5, -10, 0, -3, 8, 5, -1, 10};

countSort (arr);

printArray (arr);

return 0;

}



**Radix Sort:-**

#include<iostream>

#include<vector>

using namespace std;

int getMax(vector<int> arr)

{

int mx = arr[0];

for (int i = 1; i < arr.size(); i++)

if (arr[i] > mx)

mx = arr[i];

return mx;

}

void countSort(vector<int>& arr, int exp)

{

vector<int> output(arr.size());

int i, count[10] = {0};

for (i = 0; i < arr.size(); i++)

count[ (arr[i]/exp)%10 ]++;

for (i = 1; i < 10; i++)

count[i] += count[i - 1];

for (i = arr.size()- 1; i >= 0; i--)

{

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

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

}

for (i = 0; i < arr.size(); i++)

arr[i] = output[i];

}

void radixsort(vector<int> &arr)

{

int m = getMax(arr);

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

countSort(arr, exp);

}

void print(vector<int> arr)

{

// cout<<"\nArshdeep 17103014\n\n\n\n";

for (int i = 0; i < arr.size(); i++)

cout << arr[i] << " ";

cout<<endl;

}

int main()

{

vector<int> arr = {170, 45, 75, 90, 802, 24, 2, 66};

cout<<"Unsorted Array is: ";

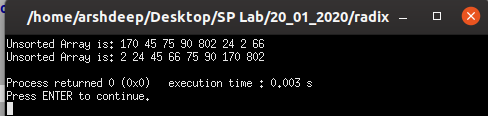
print(arr);

radixsort(arr);

cout<<"Unsorted Array is: ";

print(arr);

return 0;

}

**Lab – 3**

**The Travelling Salesman Problem:-**

#include<iostream>

#include<limits.h>

#include<vector>

using namespace std;

/\*

int graph[4][4]={

{0,20,42,25},

{20,0,30,34},

{42,30,0,10},

{25,34,10,0}

};\*/

int ALL\_COVERED = (1<<4)-1;

int tsp( vector<vector<int>> graph,int n, int bitmask, int pos)

{

if(bitmask==ALL\_COVERED)

return graph[pos][0];

else{

int ans = INT\_MAX;

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

if((1<<i&bitmask)==0){

int newAns = graph[pos][i] + tsp(graph,n, bitmask|(1<<i), i);

ans = min(ans, newAns);

}

}

return ans;

}

}

int main(){

vector<vector<int>> graph = {

{0,20,42,25},

{20,0,30,34},

{42,30,0,10},

{25,34,10,0}

};

cout<<"Graph is: \n";

for(int i=0;i<graph.size();i++)

{

for(int j=0;j<graph[i].size();j++)

cout<<graph[i][j]<<" ";

cout<<endl;

}

cout<<"\nMinimum Distance From Top-Left to Bottom-Right is: "<<tsp(graph,4, 0, 0)<<endl;

return 0;

}

