**Q3**:

Write a function customSort takes a "num" array of integers as an input along with the array of "weights" which contains the weights of the corresponding integers. We wish to sort this vector "num" from the indices start to end based on the fact that numbers with higher weights appear first and in case the weights are equal, we put the greater number first in the list. We also wish efficient algorithms. We expect an O (n\*log(n)) algorithm here instead of a O(n^2) one.

**ALGORITHM:**

**Input :n and array of weights and array of numbers**

**Output : Sorted Array according to their Weights.**

Customsort(A as array, N as array low as int, high as int){

if (low < high) then

pivot = Partition(A,N,low,high)

Customsort(A,N,low, pivot)

Customsort(A,N, pivot + 1, high)

}

Partition(A as array, N as array ,low as int, high as int){

pivot = A[low]

j = low

for i = low + 1 to high{

if (A[i] > pivot) then{

swap(A[i], A[j])

swap(N[i],N[j])

j=j+1

}

}

swap(pivot,A[j])

swap(N[high],N[j])

return j

}

**END ALGORITHM**

**TIME COMPEXITY:**

**Method 1:**

T(n)=C1\*T(n-1)\*T(n/2) +C2

T(n)=O(n2) (Worst Case) T(n)=O(n\*logn) (Other Cases)

**PROGRAM :**

#include <bits/stdc++.h>

using namespace std;

int partition (int arr[],int w[],int low, int high);

void customSort(int arr[],int w[],int low, int high);

void swap(int\* a, int\* b);

//CUSTOM SORT

//SORT ON THE BASIS OF THEIR WEIGHTS IN O(N\*LOGN)

int main()

{

int n;

cin>>n;

int num[100000],weight[100000],i;

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

{

cin>>num[i];

}

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

{

cin>>weight[i];

}

customSort(weight,num,0,n-1);

cout<<"Output Array :";

cout<<endl;

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

cout<<num[i]<<" ";

cout<<endl;

return 0;

}

int partition (int arr[],int w[],int low, int high)

{

int p = arr[high];

int i = (low - 1);

for (int j = low; j <= high- 1; j++)

{

if (arr[j] >= p)

{

i++;

swap(&arr[i], &arr[j]);

swap(&w[i], &w[j]);

}

}

swap(&arr[i + 1], &arr[high]);

swap(&w[i + 1], &w[high]);

return (i + 1);

}

void customSort(int arr[],int w[],int low, int high)

{

if (low < high)

{

int pi = partition(arr,w,low, high);

customSort(arr,w,low, pi - 1);

customSort(arr,w,pi + 1, high);

}

}

void swap(int\* a, int\* b)

{

int t = \*a;

\*a = \*b;

\*b = t;

}

**INPUT/OUTPUT :**

|  |  |  |
| --- | --- | --- |
| **S No.** | **N** | **Time(ms)** |
| 1 | 10 | 15 |
| 2 | 40 | 110 |
| 3 | 80 | 290 |
| 4 | 100 | 350 |
| 5 | 10000 | 1200 |
| 6 | 100000 | 5554 |

**COMPLEXITY GRAPH:**

**CONCLUSION:**

The original algorithm is of **O( n\*logn)** complexity and the graph is also showing similar behavior.