

# Name:- Krishna Pandey Admission No:-U20CS110

* **Insertion Sort RAM Model**

## **Input** :- A sequence of n numbers.

<a1,a2,a3,….an>

**Output** :- A sequence of n numbers

## <a'1, a'2, a'3, …. a'n>

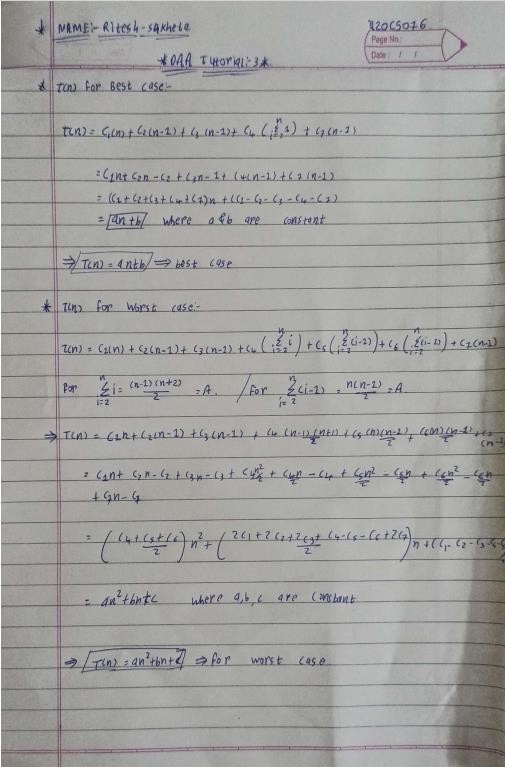
Where a'1 ≤ a'2 ≤ a'3 ≤…. a'n.

**Algorithm:-**

|  |  |  |
| --- | --- | --- |
|  | Insertion Sort(A) | Cost of each instruction |
| 1. | for i=2 to A.lenght | C1 |
| 2. | Key = A[i] | C2 |
| 3. | j = j - 1 | C3 |
| 4. | while j > 0 and A[j] > key | C4 |
| 5. | A[j+1] = A[j] | C5 |
| 6. | j = j - 1 | C6 |
| 7. | A[j+1] = key | C7 |

**Execution time:-**

|  |  |  |
| --- | --- | --- |
|  | Best | Worst |
| 1. | n | n |
| 2. | n – 1 | n – 1 |
| 3. | n – 1 | n – 1 |
| 4. | n  ∑1  i=2 | n  ∑i  i=2 |
| 5. | 0 | n  ∑(i-1)  i=2 |
| 6. | 0 | n  ∑(i-1)  i=2 |
| 7. | 1 | n – 1 |

**CALCULATION:-**

**CODE:-**

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

#define rep(i, n) for (int i = 0; i < n; i++) typedef long long ll;

void insertionSort(vector<ll> v)

{

for (ll i = 0; i < v.size(); i++)

{

ll j = i-1;

ll ritesh = v[i];

while (j>=0 && v[j]>ritesh)

{

v[j+1]=v[j]; j--;

}

v[j+1]=ritesh;

}

}

int main()

{

freopen("File 1.txt", "r", stdin);

freopen("out1.txt", "a", stdout); vector<ll> v;

ll ele;

while (cin >> ele)

{

v.push\_back(ele);

}

auto t1 = std::chrono::high\_resolution\_clock::now(); insertionSort(v);

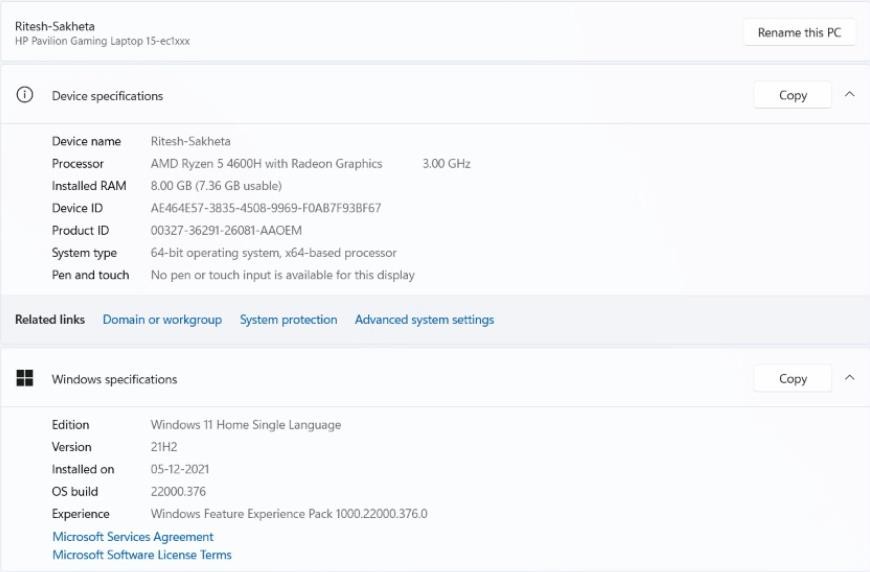
auto t2 = std::chrono::high\_resolution\_clock::now();

auto duration = std::chrono::duration\_cast<std::chrono::nanoseconds>(t2 - t1).count();

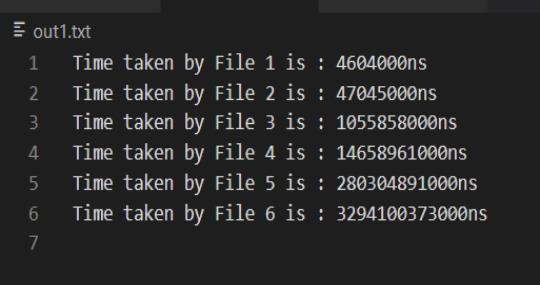
cout << "Time taken by File 1 is : " << duration << "ns" << endl; return 0;

}

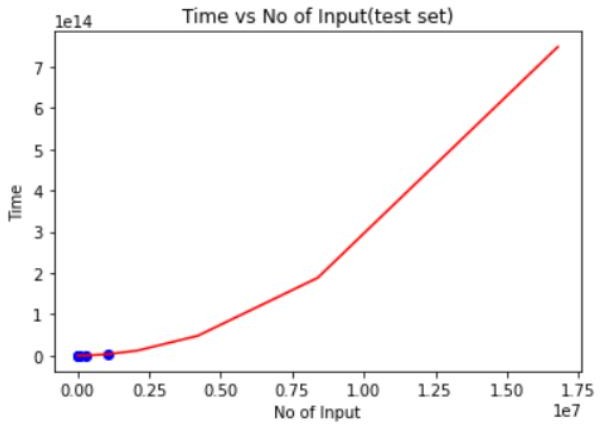
**Provide the details of Hardware/Software you used to implement the algorithm and to measure the time.**



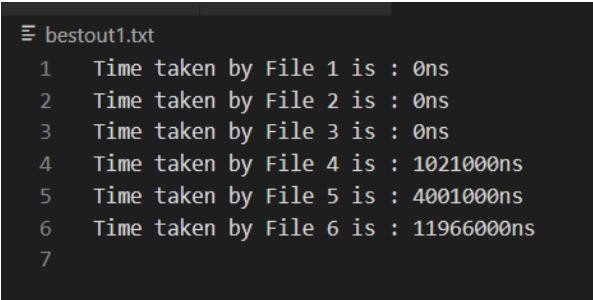
**Average case time:-**



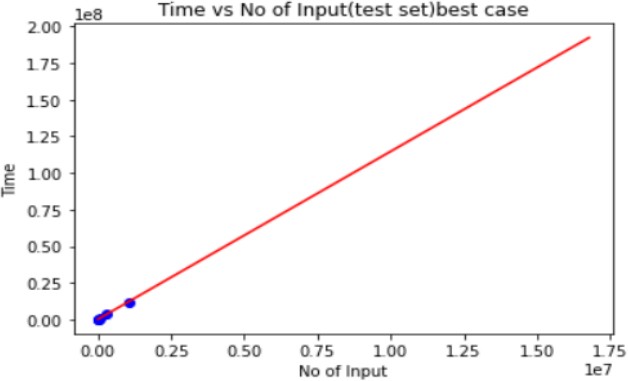
**GRAPH FOR AVERAGE CASE**



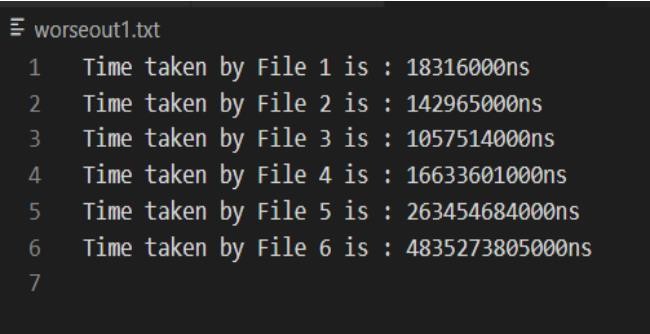
**Best case time:-**



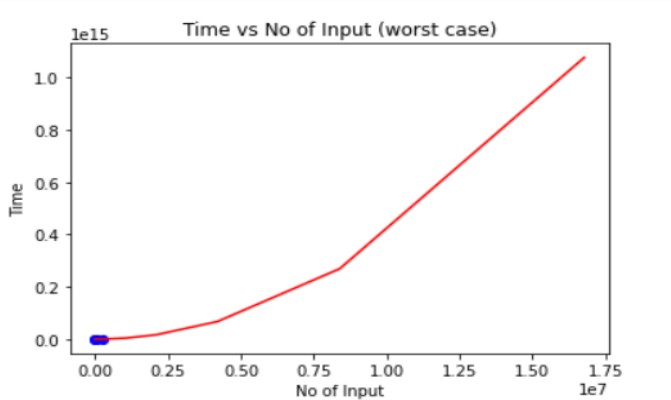
**GRAPH FOR BEST CASE**



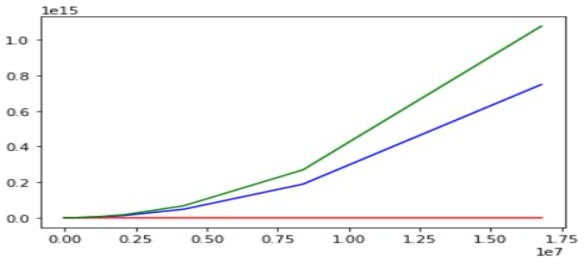
**Worst case time:-**



**GRAPH FOR WORST CASE**



**COMBINE GRAPH**



1. **Assume that you don’t know the time complexity of above**

**algorithms.**

* 1. **Can you predict the same based on your implementation?**

**By observing Graphs of each case:**

**Best case => Straight Line => So equation will be: An+B**

**Average Case/Worst Case => Parabolic graph => So equation will be: An^2+Bn+C**

* 1. **Do they match with theoretical time complexity? Yes/No.**

 **Yes, it is…..**

* 1. **If yes, then write the time complexity of each algorithm. If no, then write the difference.**

 **Time Complexity:**

 **Best Case: An+B => BIG THETA(n)**

 **Worst Case: An^2+Bn+C => BIG THETA(n^2)**