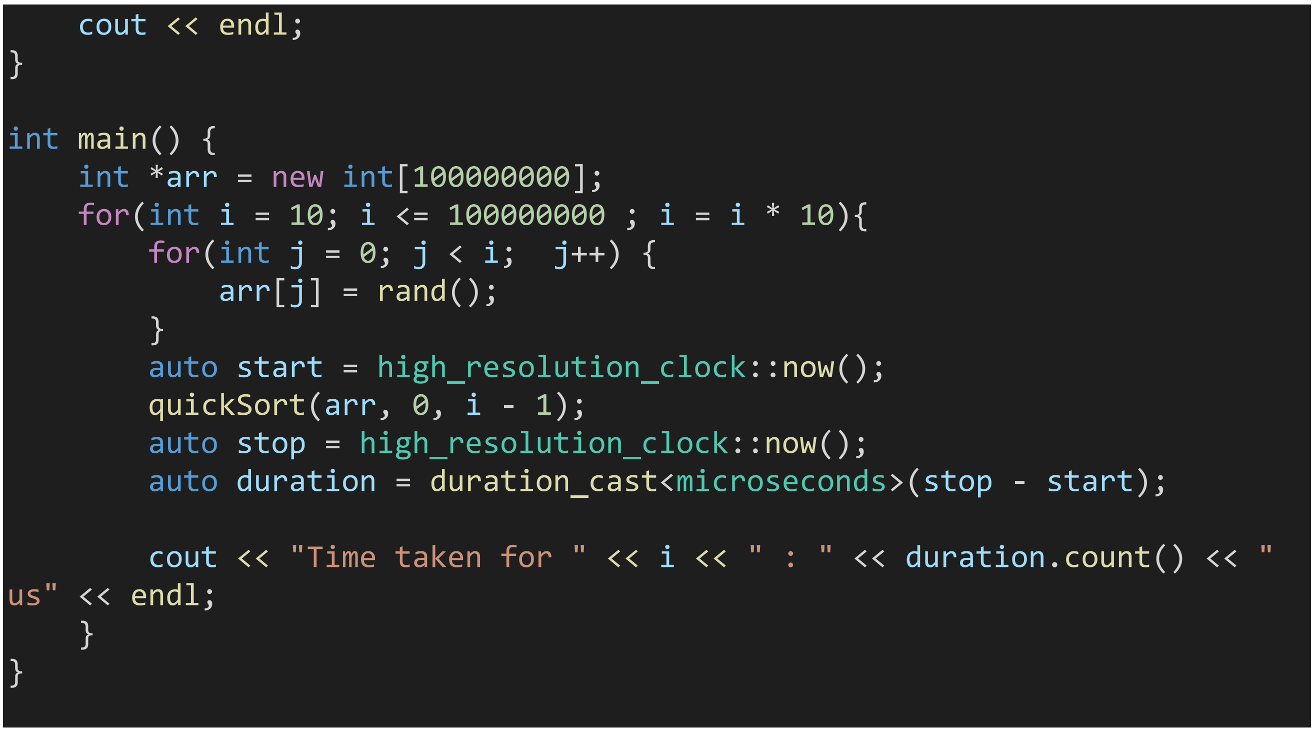
# CB3CO09 - Design and Analysis of Algorithm
# Lab 1 - Quick Sort Analysis

## 1. Average and Best Case

**Code:**

```cpp
#include <algorithm>
#include <chrono>
#include <iostream>
#include <stdlib.h>
using namespace std;
using namespace std::chrono;

void swap(int *a, int *b) {
    int temp;
    temp = *a;
    *a = *b;
    *b = temp;
}

int partition(int arr[], int low, int high) {
    int pivot = arr[high];
    int j = low-1;
    for(int i = low; i < high ; i++) {
        if(arr[i] < pivot) {
            j++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[j+1], &arr[high]);
    return j+1;
}

void quickSort(int arr[], int low, int high) {
    if(high>=low) {
        int index = partition(arr, low, high);

        quickSort(arr, low, index - 1);
        quickSort(arr, index + 1, high);
    }
}

void print(int arr[], int n) {
    for(int i = 0; i < n; i++) {
        cout << arr[i] << "  ";
    }
```



```
    cout << endl;
}

int main() {
    int *arr = new int[100000000];
    for(int i = 10; i <= 100000000 ; i = i * 10){
        for(int j = 0; j < i;  j++) {
            arr[j] = rand();
        }
        auto start = high_resolution_clock::now();
        quickSort(arr, 0, i - 1);
        auto stop = high_resolution_clock::now();
        auto duration = duration_cast<microseconds>(stop - start);

        cout << "Time taken for " << i << " : " << duration.count() << " 
us" << endl;
    }
}
```

## 2. Worst Case Analysis (Sorting a Sorted Array)

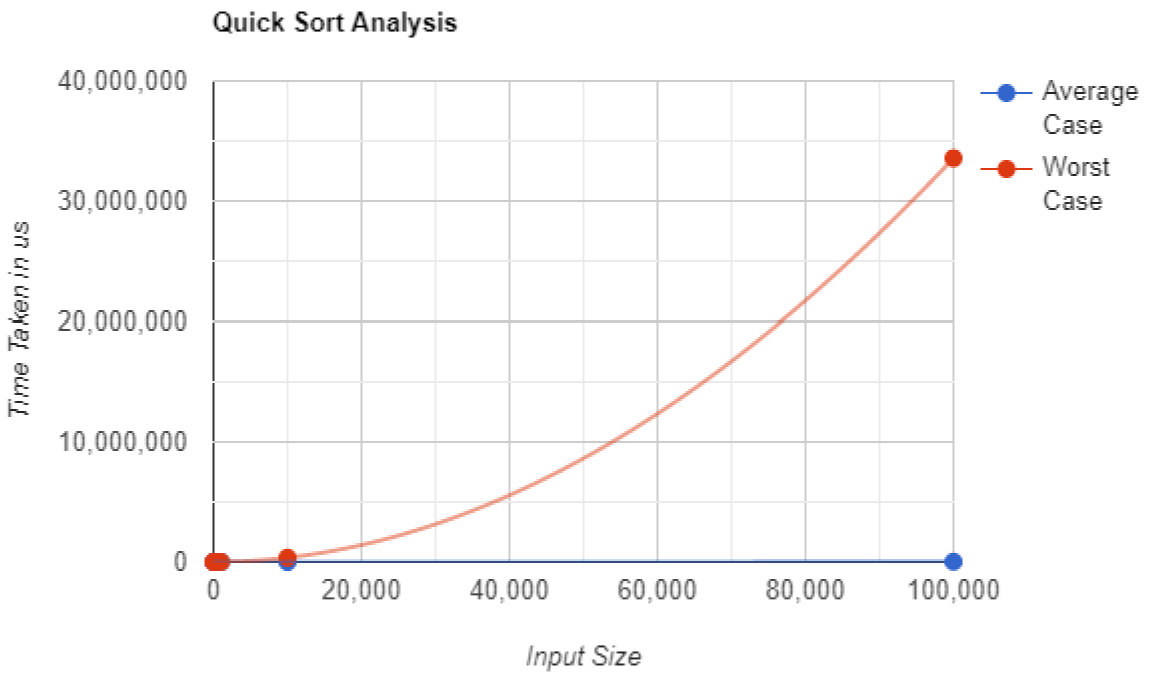
**Code:**

```
#include <algorithm>
#include <chrono>
#include <iostream>
#include <stdlib.h>
using namespace std;
using namespace std::chrono;

void swap(int *a, int *b) {
    int temp;
    temp = *a;
    *a = *b;
    *b = temp;
}

int partition(int arr[], int low, int high) {
    int pivot = arr[high];
    int j = low-1;
    for(int i = low; i < high ; i++) {
        if(arr[i] < pivot) {
            j++;
            swap(&arr[i], &arr[j]);
        }
    }
    swap(&arr[j+1], &arr[high]);
    return j+1;
}

void quickSort(int arr[], int low, int high) {
    if(high>=low) {
        int index = partition(arr, low, high);

        quickSort(arr, low, index - 1);
        quickSort(arr, index + 1, high);
    }
}

void print(int arr[], int n) {
    for(int i = 0; i < n; i++) {
        cout << arr[i] << "  ";
    }
    cout << endl;
}

int main() {
    int *arr = new int[100000000];
    for(int i = 10; i <= 100000000 ; i = i * 10){
        for(int j = 0; j < i;  j++) {
            arr[j] = j+1;
        }
        auto start = high_resolution_clock::now();
        quickSort(arr, 0, i - 1);
        auto stop = high_resolution_clock::now();
        auto duration = duration_cast<microseconds>(stop - start);

        cout << "Time taken for " << i << " : " << duration.count() << " 
us" << endl;
    }
}
```

## Comparison :

| Input Size | Average Case (time in us) | Worst Case (time in us) |
|---|---|---|
| 10 | 1 | 1 |
| 100 | 9 | 49 |
| 1000 | 128 | 4595 |
| 10000 | 1703 | 361095 |
| 100000 | 23919 | 33561464 |

## Graphical Comparison


Quick Sort Analysis
Time Taken in us
Input Size
Average Case
Worst Case