**Pseudocode:**

This pseudocode defines two procedures: "quick\_sort" and "partition". The "quick\_sort" procedure sorts an array recursively by selecting a pivot element and partitioning the array around it. The "partition" procedure rearranges the elements of the array such that all the elements smaller than the pivot are on the left of the pivot, and all the elements larger than the pivot are on the right.

PROCEDURE quick\_sort(arr: List[T], low: Integer, high: Integer)

IF low < high

pivot\_index = partition(arr, low, high)

quick\_sort(arr, low, pivot\_index - 1)

quick\_sort(arr, pivot\_index + 1, high)

PROCEDURE partition(arr: List[T], low: Integer, high: Integer)

pivot = arr[high]

i = low - 1

FOR j = low to high - 1

IF arr[j] <= pivot

i = i + 1

arr[i], arr[j] = arr[j], arr[i]

arr[i + 1], arr[high] = arr[high], arr[i + 1]

return i + 1

**Quick Sort:**



**Code:**

#include <bits/stdc++.h>

using namespace std;

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

{

int t = \*a;

\*a = \*b;

\*b = t;

}

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

{

int pivot = arr[high];

int i= (low- 1);

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

if (arr[j] < pivot) {

i++;

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

}

}

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

return (i + 1);

}

void quickSort(int arr[], int low, int high)

{

if (low < high) {

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

quickSort(arr, low, pi - 1);

quickSort(arr, pi + 1, high);

}

}

void Top5(int arr[], int size)

{

int i, count=1;

for (i = size-1; count<=5 ; i--)

{

count++;

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

}

cout << endl;

}

int main()

{

int arr[] = {10000, 7000, 8000, 90000, 100000, 5000};

int n = sizeof(arr) / sizeof(arr[0]);

cout<<"Entered Salary list is:\n";

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

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

cout<<"\n";

quickSort(arr, 0, n - 1);

cout << "Top 5 highest salaries: \n";

Top5(arr, n);

return 0;

}

**Output Screenshot:**

