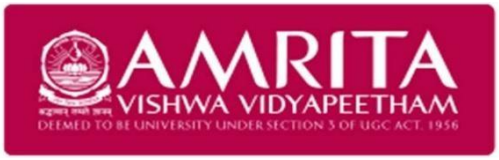

SCHOOL OF
COMPUTING

Design s Analysis of Algorithm

CSE211

Date: 25/01/2025

Week-5

CH.SC.U4CSE24136

P. Geetesh

Quick Sort

Code:

```c
//CH.SC.U4CSE24136
#include <stdio.h>
#include <stdlib.h>

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

int partition(int arr[], int low, int high, int pivotChoice) {
    int pivotIndex;
    if (pivotChoice == 1) {
        pivotIndex = low;
    } else if (pivotChoice == 2) {
        pivotIndex = high;
    } else {
        pivotIndex = low + rand() % (high - low + 1);
    }

    printf("Random Pivot Element Chosen: %d\n", arr[pivotIndex]);
    swap(&arr[pivotIndex], &arr[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, int pivotChoice) {
    if (low < high) {
        int pi = partition(arr, low, high, pivotChoice);
        quickSort(arr, low, pi - 1, pivotChoice);
        quickSort(arr, pi + 1, high, pivotChoice);
    }
}
```

```c
int main() {
    printf("CH.SC.U4CSE24136\n");

    int n, pivotChoice;
    printf("Enter number of elements: ");
    if (scanf("%d", &n) != 1) return 1;

    int *arr = (int *)malloc(n * sizeof(int));

    printf("Enter %d elements: ", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    printf("Choose Pivot Element:\n1. First Element\n2. Last Element\n3. Random Element\nChoice: ");
    scanf("%d", &pivotChoice);

    quickSort(arr, 0, n - 1, pivotChoice);

    printf("Sorted array: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    free(arr);
    return 0;
}
```

Output:

```
geetesh@geetesh-Victus-by-HP-Gaming-Laptop-15-fa1xxx:~$ sudo su
[sudo] password for geetesh:
root@geetesh-Victus-by-HP-Gaming-Laptop-15-fa1xxx:/home/geetesh# gcc Quick.c -o Quick
root@geetesh-Victus-by-HP-Gaming-Laptop-15-fa1xxx:/home/geetesh# ./Quick
CH.SC.U4CSE24136
Enter number of elements: 12
Enter 12 elements: 157 110 147 122 111 149 151 141 123 112 117 133
Choose Pivot Element:
1. First Element
2. Last Element
3. Random Element
Choice: 1
Random Pivot Element Chosen: 157
Random Pivot Element Chosen: 133
Random Pivot Element Chosen: 117
Random Pivot Element Chosen: 112
Random Pivot Element Chosen: 111
Random Pivot Element Chosen: 123
Random Pivot Element Chosen: 141
Random Pivot Element Chosen: 147
Random Pivot Element Chosen: 149
Sorted array: 110 111 112 117 122 123 133 141 147 149 151 157
```

Space Complexity and its Justification:

Time Complexity

Best case: O(n log n)

Average case: O(n log n)

Worst case: $O(n^2)$

Justification (Time)

Partition takes O(n) time.

Recursion depth is log n when partitions are balanced → O(n log n).

Worst case occurs when pivot is always smallest/largest (highly unbalanced) → $O(n^2)$.

Space Complexity

Best / Average case: O(log n)

Worst case: O(n)

Justification (Space)

Extra space is only due to recursion stack.

Balanced recursion → depth log n.

Unbalanced recursion → depth n.