

NAME: T. VASANTHA

ROLL NO: CH.SC.U4CSE24147

Week-2,3

sorting techniques

Bubble sort:

Code:

```
#include <stdio.h>
int main() {
    int n;
    printf("Enter the number of elements you need:");
    scanf("%d",&n);
    int arr[n];
    for(int i=0;i<n;i++){
        scanf("%d",&arr[i]);
    }
    for (int i = 0;i < n - 1; i++) {
        for (int j = 0;j < n - i - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
}
```

Output:

```
root@amma43:/home/amma/Documents# gcc -o bubble bubble.c
root@amma43:/home/amma/Documents# ./bubble
Enter the number of elements you need:5
12 56 34 98 26
12 26 34 56 98 root@amma43:/home/amma/Documents# |
```

Insertion sort:

Code:

```
#include <stdio.h>

int main() {
    int arr[] = {45,23,87,41,98};
    int n = sizeof(arr) / sizeof(arr[0]);
    for (int i = 1; i < n; i++) {
        int key = arr[i];
        int j = i - 1;
        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j--;
        }
        arr[j + 1] = key;
    }
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");

    return 0;
}|
```

Output:

```
root@amma43:/home/amma/Documents# gcc -o sort sort.c
root@amma43:/home/amma/Documents# ./sort
23 41 45 87 98
root@amma43:/home/amma/Documents#
```

Selection sort:

Code:

```
#include <stdio.h>
int main() {
    int arr[] = {56,23,78,46,12};
    int n = sizeof(arr) / sizeof(arr[0]);
    for (int i = 0; i < n - 1; i++) {
        int minIndex = i;
        for (int j = i + 1; j < n; j++)
            if (arr[j] < arr[minIndex])
                minIndex = j;
        int temp = arr[minIndex];
        arr[minIndex] = arr[i];
        arr[i] = temp;
    }
    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);

    return 0;
}
```

Output:

```
root@amma43:/home/amma/Documents# gcc -o selection selection.c
root@amma43:/home/amma/Documents# ./selection
12 23 46 56 78 root@amma43:/home/amma/Documents# |
```

Bucket sort:

Code:

```
#include <stdio.h>

int main() {
    int arr[] = {45,23,87,41,98};
    int n = sizeof(arr) / sizeof(arr[0]);
    int bucket[100] = {0};
    for (int i = 0; i < n; i++)
        bucket[arr[i]]++;
    for (int i = 0; i < 100; i++)
        while (bucket[i]--)
            printf("%d ", i);

    return 0;
}
```

Output:

```
root@amma43:/home/amma/Documents# gcc -o bucket bucket.c
root@amma43:/home/amma/Documents# ./bucket
23 41 45 87 98 root@amma43:/home/amma/Documents#
```

Heap sort:

Code:

```
#include <stdio.h>
int main() {
    int arr[] = {12, 11, 13, 5, 6, 7};
    int n = sizeof(arr) / sizeof(arr[0]);
    for (int i = n / 2 - 1; i >= 0; i--) {
        int root = i;
        while (1) {
            int left = 2 * root + 1;
            int right = 2 * root + 2;
            int largest = root;
            if (left < n && arr[left] > arr[largest])
                largest = left;
            if (right < n && arr[right] > arr[largest])
                largest = right;
            if (largest == root) break;
            int temp = arr[root];
            arr[root] = arr[largest];
            arr[largest] = temp;
            root = largest;
        }
    }
    for (int end = n - 1; end > 0; end--) {
        int temp = arr[0];
        arr[0] = arr[end];
        arr[end] = temp;
        int root = 0;
        while (1) {
            int left = 2 * root + 1;
            int right = 2 * root + 2;
```

```
        int largest = root;
        if (left < end && arr[left] > arr[largest])
            largest = left;
        if (right < end && arr[right] > arr[largest])
            largest = right;
        if (largest == root) break;
        int temp2 = arr[root];
        arr[root] = arr[largest];
        arr[largest] = temp2;
        root = largest;
    }
}
for (int i = 0; i < n; i++)
    printf("%d ", arr[i]);
return 0;
}
```

Output:

```
root@amma43:/home/amma/Documents# gcc -o heap heap.c
root@amma43:/home/amma/Documents# ./heap
23 36 47 59 60 root@amma43:/home/amma/Documents#
```

BFS Traversal:

Code:

```
#include <stdio.h>
#define MAX 100

int adj[MAX][MAX];
int visited[MAX];
int queue[MAX];
int front = 0, rear = 0;
int n;

void BFS(int start) {
    visited[start] = 1;
    queue[rear++] = start;
    while (front != rear) {
        int v = queue[front++];
        printf("%d ", v);
        for (int i = 0; i < n; i++) {
            if (adj[v][i] == 1 && !visited[i]) {
                visited[i] = 1;
                queue[rear++] = i;
            }
        }
    }
}

int main() {
    int edges, u, v;
    printf("Enter number of vertices: ");
    scanf("%d", &n);

    printf("Enter number of edges: ");
    scanf("%d", &edges);

    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            adj[i][j] = 0;

    printf("Enter edges (u v):\n");
    for (int i = 0; i < edges; i++) {
        scanf("%d %d", &u, &v);
        adj[u][v] = adj[v][u] = 1;
```

```
    }

    printf("\nBFS Traversal starting from 0:\n");
    BFS(0);

    return 0;
}
```

Output:

```
root@amma29:/home/amma/Downloads# gcc -o bfs bfs.c
root@amma29:/home/amma/Downloads# ./bfs
Enter number of vertices: 5
Enter number of edges: 4
Enter edges (u v):
0 1
0 2
1 3
1 4

BFS Traversal starting from 0:
0 1 2 3 4 root@amma29:/home/amma/Downloads#
```

DFS Traversal:

Code:

```

#include <stdio.h>
#define MAX 100

int adj[MAX][MAX];
int visited[MAX];
int n;

void DFS(int v) {
    visited[v] = 1;
    printf("%d ", v);
    for (int i = 0; i < n; i++) {
        if (adj[v][i] == 1 && !visited[i]) {
            DFS(i);
        }
    }
}

int main() {
    int edges, u, v;

    printf("Enter number of vertices: ");
    scanf("%d", &n);

    printf("Enter number of edges: ");
    scanf("%d", &edges);

    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            adj[i][j] = 0;

    printf("Enter edges (u v):\n");
    for (int i = 0; i < edges; i++) {
        scanf("%d %d", &u, &v);
        adj[u][v] = adj[v][u] = 1;
    }

    printf("\nDFS Traversal starting from 0:\n");
    DFS(0);

    return 0;
}

```

Output:

```
0 root@amma29:/home/amma/Downloads# gcc -o dfs dfs.c
root@amma29:/home/amma/Downloads# ./dfs
Enter number of vertices: 5
Enter number of edges: 4
Enter edges (u v):
0 1
0 2
1 3
1 4

DFS Traversal starting from 0:
0 1 3 4 2 root@amma29:/home/amma/Downloads#
```