



NAME : D V N Surya Kiran
SEC : CSE-B
ROLL NO : CH.SC.U4CSE24116

1. Bubble Sort

PROGRAM:

```
#include <stdio.h>
int main() {
    int n;
    printf("ENTER NO.OF.ELEMENTS:");
    scanf("%d",&n);
    int arr[n];
    printf("ENTER THE VALUES:");
    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;
            }
        }
    }
}
```

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

```
amma@amma11:~$ gcc bubble_sort.c -o bubble_sort
amma@amma11:~$ ./bubble_sort
enter element 0: 6
enter element 1: 9
enter element 2: 8
enter element 3: 1
enter element 4: 3
1
3
6
8
9
```

2. Insertion Sort

PROGRAM:

```
#include <stdio.h>

int main() {
    int n;

    printf("ENTER NO.OF ELEMENTS");
    scanf("%d",&n);
    int arr[n];

    printf("ENTER THE ELEMENTS:");
    for(int i=0; i <n;i++){
```

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

```
root@amma55:/home/amma# gcc insertionsort.c -o insertionsort  
root@amma55:/home/amma# ./insertionsort  
Enter number of elements: 7  
Enter 7 elements:  
4  
5  
6  
7  
1  
0  
2  
Sorted array: 0 1 2 4 5 6 7
```

3. Selection Sort

PROGRAM:

```
#include <stdio.h>

int main() {
    int n;

    printf("ENTER NO. OF ELEMENTS: ");
    scanf("%d", &n);

    int arr[n];

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

    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[i];
        arr[i] = arr[minIndex];
        arr[minIndex] = temp;
    }
}
```

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

```
amma@amma11:~$ gcc selection_sort.c -o selection_sort  
amma@amma11:~$ ./selection_sort  
enter element 0: 9  
enter element 1: 1  
enter element 2: 0  
enter element 3: 3  
enter element 4: 6  
0  
1  
3  
6  
9
```

4. Bucket Sort

PROGRAM:

```
#include <stdio.h>
```



```
int main() {  
    int n;  
    printf("ENTER NO. OF ELEMENTS: ");  
    scanf("%d", &n);  
    int arr[n];  
    printf("ENTER THE VALUES (0 to 100): ");  
    for (int i = 0; i < n; i++) {  
        scanf("%d", &arr[i]);  
    }  
    int bucket[101] = {0};  
    for (int i = 0; i < n; i++) {  
        bucket[arr[i]]++;  
    }  
    printf("Sorted array: ");  
    for (int i = 0; i <= 100; i++) {  
        while (bucket[i] > 0) {  
            printf("%d ", i);  
            bucket[i]--;  
        }  
    }  
    return 0;  
}
```

```
amma@amma11:~$ gcc bucket_sort.c -o bucket_sort
amma@amma11:~$ ./bucket_sort
enter element 0: 8
enter element 1: 2
enter element 2: 0
enter element 3: 1
enter element 4: 7
0
1
2
7
8
```

5) Max Heap

Program :

```
#include <stdio.h>
```

Max Heap

```
void maxHeapify(int a[], int  
n, int i) {
```

```
    int largest = i;
```

```
    int left = 2 * i + 1;
```

```
    int right = 2 * i + 2;
```

```
    int temp;
```

```
    if (left < n && a[left] >
a[largest])
        largest = left;

    if (right < n && a[right] >
a[largest])
        largest = right;

    if (largest != i) {
        temp = a[i];
        a[i] = a[largest];
        a[largest] = temp;

        maxHeapify(a, n,
largest);
    }
}
```

```
void buildMaxHeap(int a[],
int n) {
    int i;
    for (i = n / 2 - 1; i >= 0; i--)
    )
        maxHeapify(a, n, i);
}
```

```
int main() {

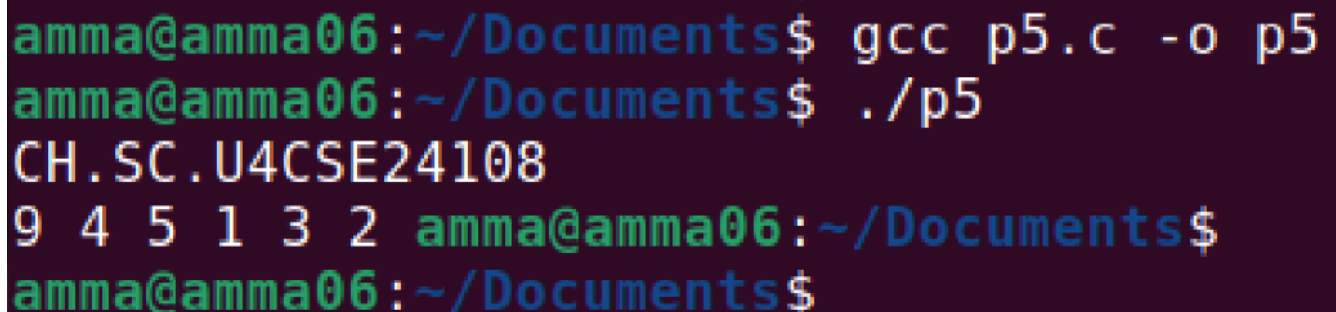
printf("CH.SC.U4CSE2410
8\n");
```

```
    int a[6] = {3, 9, 2, 1, 4,
5};
    int n = 6, i;

    buildMaxHeap(a, n);
```

```
    for (i = 0; i < n; i++)  
        printf("%d ", a[i]);  
  
    return 0;  
}
```

OUTPUT :



```
amma@amma06:~/Documents$ gcc p5.c -o p5  
amma@amma06:~/Documents$ ./p5  
CH.SC.U4CSE24108  
9 4 5 1 3 2 amma@amma06:~/Documents$  
amma@amma06:~/Documents$
```

6) Min Heap

Program:

```
#include <stdio.h>
```

Min Heap

```
void minHeapify(int a[], int  
n, int i) {  
    int smallest = i;  
    int left = 2 * i + 1;
```

```
int right = 2 * i + 2;
```

```
int temp;
```

```
if (left < n && a[left] <  
a[smallest])
```

```
    smallest = left;
```

```
if (right < n && a[right] <  
a[smallest])
```

```
    smallest = right;
```

```
if (smallest != i) {
```

```
    temp = a[i];
```

```
    a[i] = a[smallest];
```

```
    a[smallest] = temp;
```

```
    minHeapify(a, n,  
smallest);
```

```
}
```

```
}
```

```
void buildMinHeap(int a[],  
int n) {  
    int i;  
    for (i = n / 2 - 1; i >= 0; i--  
)  
        minHeapify(a, n, i);  
}
```

```
int main() {  
  
    printf("CH.SC.U4CSE2410  
8\n");  
  
    int a[6] = {3, 9, 2, 1, 4,  
5};  
    int n = 6, i;
```

```

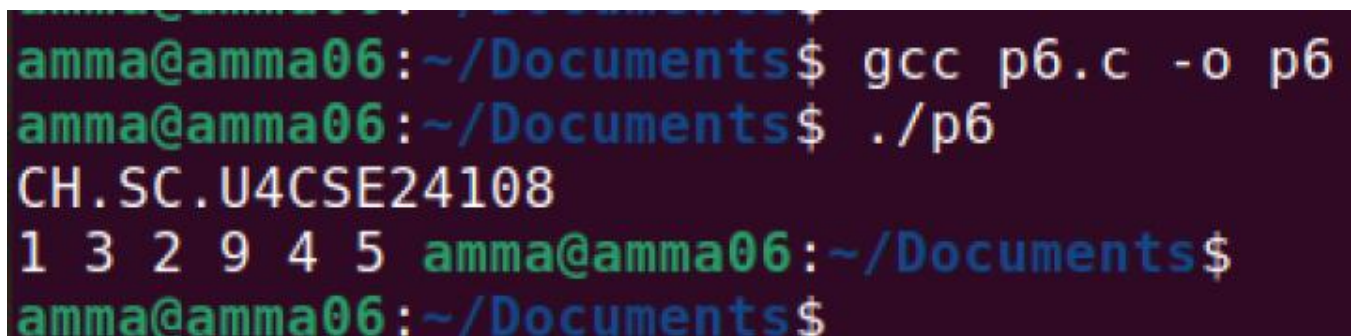
    buildMinHeap(a, n);

    for (i = 0; i < n; i++)
        printf("%d ", a[i]);

    return 0;
}

```

OUTPUT :



```

amma@amma06:~/Documents$ gcc p6.c -o p6
amma@amma06:~/Documents$ ./p6
CH.SC.U4CSE24108
1 3 2 9 4 5 amma@amma06:~/Documents$
amma@amma06:~/Documents$

```

7) BFS

Program :

```
#include <stdio.h>
```

```
#define MAX 100
```

BFS

```
int queue[MAX], front = 0,
```



```
rear = 0;
```

```
void enqueue(int x) {  
    queue[rear++] = x;  
}
```

```
int dequeue() {  
    return queue[front++];  
}
```

```
void bfs(int  
graph[MAX][MAX], int n, int  
start) {  
    int visited[MAX];  
    int i, node;  
  
    for (i = 0; i < n; i++)  
        visited[i] = 0;
```

```
enqueue(start);
visited[start] = 1;

while (front != rear) {
    node = dequeue();
    printf("%d ", node);

    for (i = 0; i < n; i++) {
        if (graph[node][i] ==
1 && !visited[i]) {
            visited[i] = 1;
            enqueue(i);
        }
    }
}

int main() {
```

```
printf("CH.SC.U4CSE2410  
8\n");
```

```
int n = 4;  
int graph[MAX][MAX] = {  
    {0, 1, 1, 0},  
    {1, 0, 1, 1},  
    {1, 1, 0, 0},  
    {0, 1, 0, 0}  
};
```

```
bfs(graph, n, 0);
```

```
return 0;  
}
```

Output :

```
amma@amma06:~/Documents$ gcc p7.c -o p7
amma@amma06:~/Documents$ ./p7
CH.SC.U4CSE24108
0 1 2 3 amma@amma06:~/Documents$
amma@amma06:~/Documents$
```