

LAB 8 :LAB EXERCISES :

/\*1) Write a program to create a heap for the list of integers using top-down heap construction algorithm and analyze its time efficiency. Obtain the experimental results for order of growth and plot the result.\*/

```
#include <stdio.h>
#include <stdlib.h>
int opCount = 0;
void HeapTopDown(int nameArray[], int startingIndex)
{
    int parentalDominance = (startingIndex - 1) / 2;
    while (parentalDominance ≥ 0)
    {
        opCount++;
        if (nameArray[parentalDominance] < nameArray[startingIndex])
        {
            int temp = nameArray[parentalDominance];
            nameArray[parentalDominance] = nameArray[startingIndex];
            nameArray[startingIndex] = temp;
            startingIndex = parentalDominance;
            parentalDominance = (startingIndex - 1) / 2;
        }
        else
            return;
    }
}
void main()
{
    int heapArray[20];
    int size;
    int i;
    int j;
    printf("Enter the size of elements : ");
    scanf("%d", &size);
    printf("Enter the elements : ");
    for (i = 0; i < size; i++)
    {
        scanf("%d", &heapArray[i]);
        printf("\n");
        printf("The Array is getting in Heap : \n\n");
        HeapTopDown(heapArray, i);
        for (j = 0; j ≤ i; j++)
            printf("%d ", heapArray[j]);
        printf("\n");
    }
}
```

```

printf("\n\n");
printf("Orgnized Heap Array is : \n\n");
for (i = 0; i < size; i++)
printf("%d\t", heapArray[i]);
printf("\n\n");
printf("Opcount of HeapTopDown is = %d ", opCount);
printf("\n\n\n");
exit(0);}

```

OUTPUT :

The screenshot shows a C program in Visual Studio Code. The code implements a HeapTopDown sort. The terminal output shows the execution of the program with the following steps:

```

linuxcode@linuxcode:~/FOURTH_SEMESTER/DAA_LAB/lab8$ ls
avg2.png  pgm1  pgm1.png  pgm2.c  Plot1.png
avg.png   pgm1.c  pgm2     pgm2.png
linuxcode@linuxcode:~/FOURTH_SEMESTER/DAA_LAB/lab8$ gcc pgm1.c -o pgm1
linuxcode@linuxcode:~/FOURTH_SEMESTER/DAA_LAB/lab8$ ./pgm1
Enter the size of elements : 6
Enter the elements : 8 9 7 6 5 4

The Array is getting in Heap :
8

The Array is getting in Heap :
9 8

The Array is getting in Heap :
9 8 7

The Array is getting in Heap :
9 8 7 6

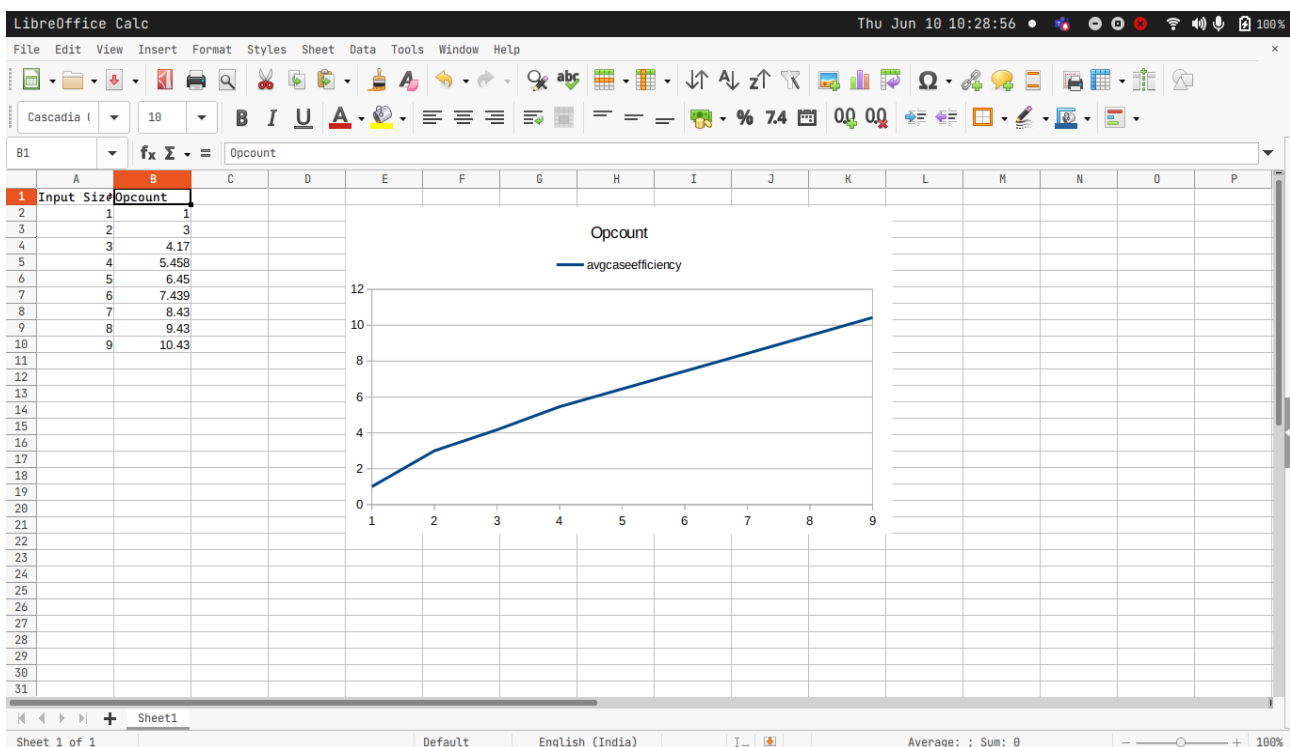
The Array is getting in Heap :
9 8 7 6 5

The Array is getting in Heap :
9 8 7 6 5 4

Orgnized Heap Array is :
9 8 7 6 5 4

Opcount of HeapTopDown is = 7

```



/\*Write a program to sort the list of integers using heap sort with bottom up max heap construction and analyze its time efficiency. Prove experimentally that the worst case time complexity is  $O(n \log n)$ \*/

```
#include <stdio.h>
#include <stdlib.h>
int opCount = 0;
void heapifyBottomUp(int heapArray[], int left, int size)
{
    int i;
    int k;
    int trace;
    int heapifyMenu;
    int j;
    for (i = (size / 2); i ≥ left; i--)
    {
        k = i;
        trace = heapArray[k];
        heapifyMenu = 0;
        while (heapifyMenu == 0 && 2 * k ≤ size)
        {
            j = 2 * k;
            opCount++;
            if (j < size)
            if (heapArray[j] < heapArray[j + 1])
            j = j + 1;
            if (trace ≥ heapArray[j])
            heapifyMenu = 1;
            else
            {
                heapArray[k] = heapArray[j];
                k = j;
            }
        }
        heapArray[k] = trace;
    }
    return;
}
void HeapSortUsingBottomUp(int heapArray[], int size)
{
    int j = 0;
    for (int i = 1; i ≤ size; i++)
    {
        heapifyBottomUp(heapArray, 1, size - j);
        int temp = heapArray[1];
```

```

heapArray[1] = heapArray[size - j];
heapArray[size - j] = temp;
j++;
}
}
void main()
{
int heapArray[20];
int size;
int i;
printf("Enter the size of Elemets : ");
scanf("%d", &size);
printf("\n\n");
printf("Enter the Elements : ");
for (i = 1; i ≤ size; i++)
scanf("%d", &heapArray[i]);
HeapSortUsingBottomUp(heapArray, size);
printf("\n\n");
printf("The Heap Sort Array is : \n");
printf("\n");
for (i = 1; i ≤ size; i++)
printf("%d ", heapArray[i]);
printf("\n\n");
printf("The Opcount is = %d\n", opCount);
printf("\n\n");
}

```

### OUTPUT :

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer:** Shows a project structure with folders 'lab6', 'lab7', 'lab8', and 'lab9'. The 'lab8' folder is expanded, showing files like 'avg.png', 'avg2.png', 'pgm1', 'pgm1.c', 'pgm2', and 'pgm2.c'.
- Code Editor:** Displays the source code for 'pgm2.c'. The code includes comments and implements a heap sort algorithm using a 'HeapSortUsingBottomUp' function. The code is partially visible, showing the main function and the heap sort logic.
- Terminal:** Shows the execution of the program. The user enters the size of the array as 6 and the elements as 8 9 7 6 5 4. The output shows the sorted array as 4 5 6 7 8 9 and the operation count as 11.

