PROGRAM - 1

```
    HIGHEST PRIORITY SHOULD BE EXECUTED FIRST */
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
struct Node
{
    char arr[4];
    int pr;
    struct Node *next;
} *front, *q;
int count=0;
void insert()
{
    char arr[4];
    int p;
    struct Node *ptr;
    ptr = (struct Node *) malloc (sizeof (struct Node));
    printf ("Enter the name of process to be insert: ");
    scanf ("%s",ptr->arr);
    printf("its priority : ");
    scanf("%d",&p);
    ptr->pr=p;
    ptr->next=NULL;
    if( front == NULL || p>front->pr )
    {
        ptr->next = front;
        front = ptr;
    }
    else
    {
        q = front;
        while( q->next != NULL && q->next->pr>=p)
        {
            q=q->next;
        }
        ptr->next = q->next;
        q->next = ptr;
    }
    count++;
    printf ("\nInsertion is Success!!!\n");
    printf("display : ");
}
void execute()
{

    if(front== NULL)
    {
        printf ("\nQueue Underflow!!!\n");
    }
    else
    {
        printf ("\nDeleted element: %s\n", front->arr);
        front = front->next;
        count--;
    }
    printf("display : ");
}
void display()
{
    if (front == NULL)
    {
        printf ("\nQueue is Empty!!!\n");
    }
    else
    {
        struct Node *temp = front;
        while (temp->next != NULL)
        {
            printf ("| %s | %d |--->", temp->arr, temp->pr);
            temp = temp->next;
        }
        printf ("| %s | %d |--->NULL", temp->arr, temp->pr);
    }
}
void totalelem()
{
    printf("total number of nodes are : %d\n",count);
    printf("display : ");
}
int main ()
{
    front=NULL;
    int choice;
    printf ("\n:: Priority Queue using Linked List ::\n");
    while (choice!=0)
    {
        printf ("\n-------- MENU ---------\n");
        printf ("1.Insert\n2.execute\n3.total 
processes\n4.display\n0.Exit\n");
        printf ("Enter your choice: ");
        scanf ("%d", &choice);
        switch (choice)
        {
            case 1:
            {
                insert();
                display();
                break;
            }
            case 2:
            {
                execute();
                display();
                break;
            }
            case 3:
            {
                totalelem();
                display();
                break;
            }
            case 4:
            {
                display();
                break;
            }
            case 0:
            {
                printf("\nexited..\n");
                break;
            }
            default:
            {
                printf ("\nWrong selection... Please try again\n");
            }
        }
    }
    return 0;
}
```