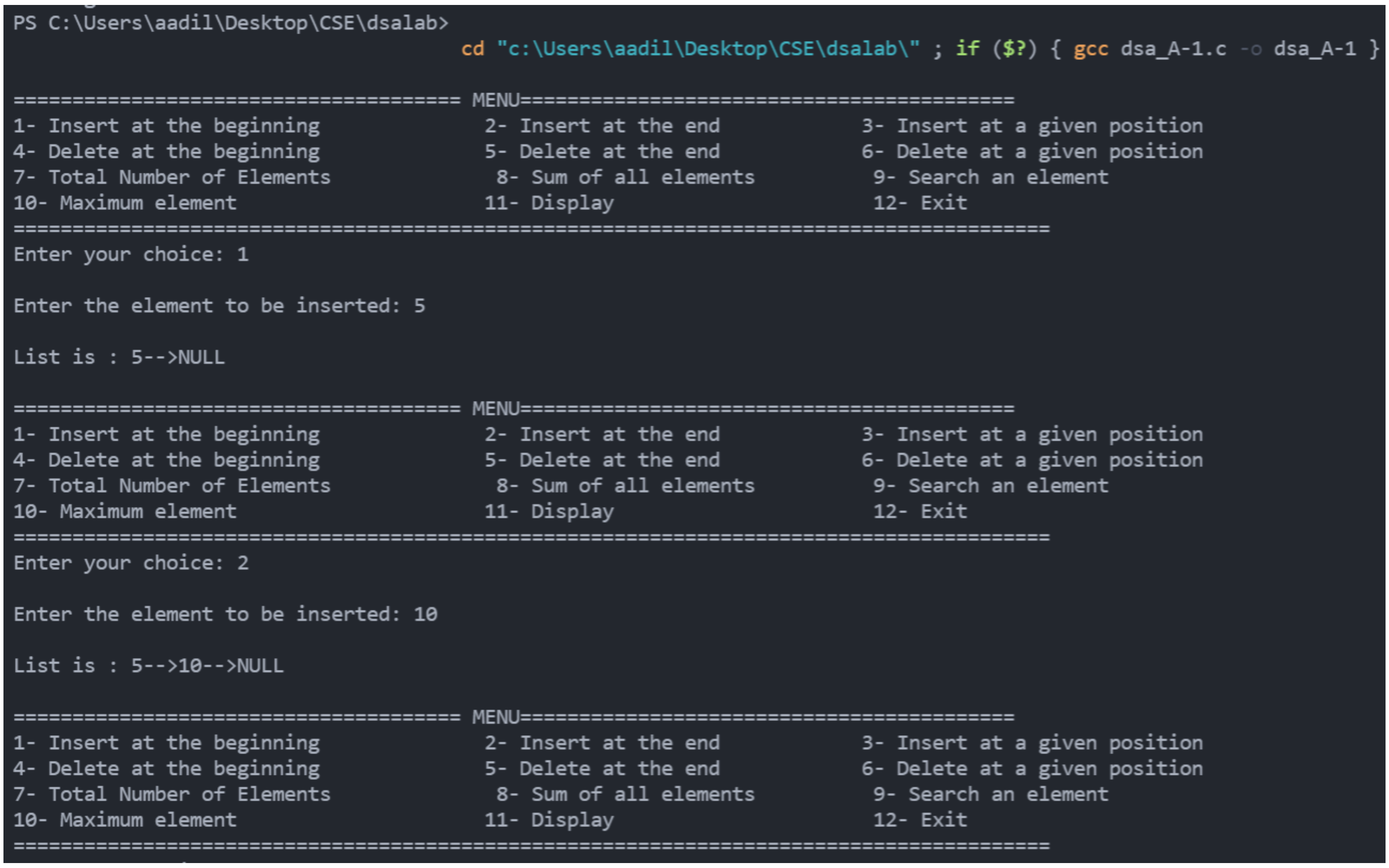
```c
#include <stdio.h>
#include <stdlib.h>
// node structure of the linked list
struct node
{
    int data;
    struct node *next;
};
// global variables for the linked list
struct node *start = NULL;
struct node *header = NULL;
// function to insert a node at the beginning of the list
void insert_begin(int data)
{
    struct node *temp = (struct node *)malloc(sizeof(struct node));
    temp->data = data;
    temp->next = start->next;
    start->next = temp;
    start->data++;
}
// function to insert a node at the end of the list
void insert_end(int data)
{
    struct node *new_node = (struct node *)malloc(sizeof(struct node));
    new_node->data = data;
    new_node->next = NULL;
    if (start->next == NULL)
    {
        start->next = new_node;
    }
    else
    {
        struct node *temp = start->next;
        while (temp->next != NULL)
        {
            temp = temp->next;
        }
        temp->next = new_node;
    }
    start->data++;
}
// function to insert a node at a given position
void insert_pos(int data, int pos)
{
    struct node *new_node = (struct node *)malloc(sizeof(struct node));
    new_node->data = data;
    if (pos >= 0 && pos <= start->data)
    {
        if (pos == 0)
        {
            new_node->next = start->next;
            start->next = new_node;
        }
        else
        {
            struct node *temp = start->next;
            int i = 0;
            while (i < pos - 1)
            {
                temp = temp->next;
                i++;
            }
            new_node->next = temp->next;
            temp->next = new_node;
        }
        start->data++;
    }
    else
    {
        printf("Invalid position!!!\n");
    }
}
// function to delete a node from the beginning of the list
int delete_begin()
{
    if (start->next == NULL)
    {
        printf("List Underflow\n");
        return -1;
    }
    else
    {
        struct node *temp = start->next;
        start->next = start->next->next;
        int data = temp->data;
        free(temp);
        start->data--;
        return data;
    }
}
```

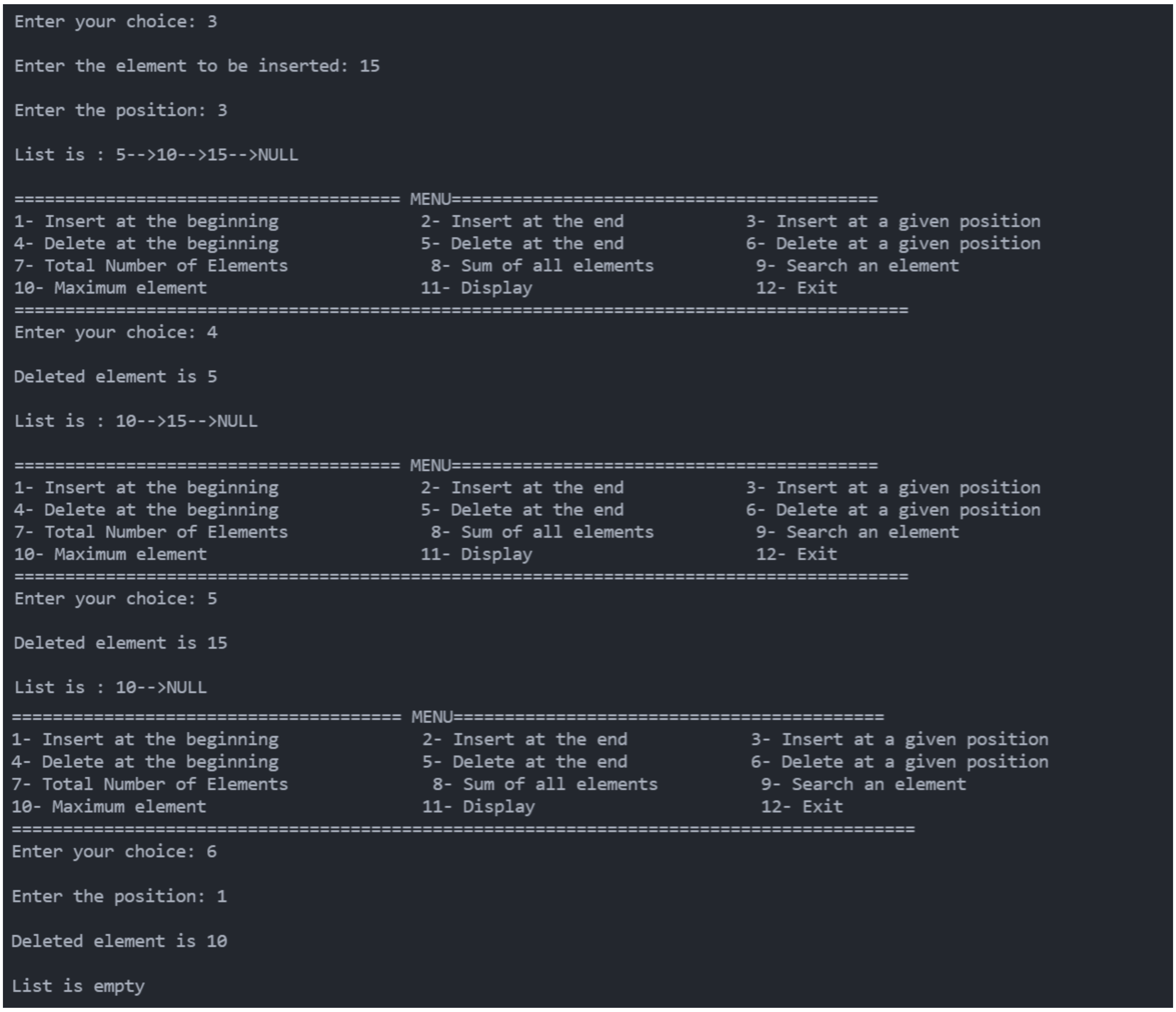
```c
// function to delete a node from the end of the list
int delete_end()
{
    if (start->next == NULL)
    {
        printf("List Underflow!!!\n");
        return -1;
    }
    else
    {
        struct node *temp = start->next;
        while (temp->next->next != NULL)
        {
            temp = temp->next;
        }
        int data = temp->next->data;
        free(temp->next);
        temp->next = NULL;
        start->data--;
        return data;
    }
}
// function to delete a node from a given position
int delete_pos(int pos)
{
    if (pos >= 0 && pos < start->data)
    {
        if (pos == 0)
        {
            return delete_begin();
        }
        else
        {
            struct node *temp = start->next;
            int i = 0;
            while (i < pos - 1)
            {
                temp = temp->next;
                i++;
            }
            struct node *temp2 = temp->next;
            temp->next = temp->next->next;
            int data = temp2->data;
            free(temp2);
            start->data--;
            return data;
        }
    }
    else
    {
        printf("Invalid position!!!\n");
        return -1;
    }
}
// function for calculating the size of the list
int total_elements()
{
    return start->data;
}
// function to sum the elements of the list
int sum_elements()
{
    if (start->next == NULL)
    {
        return 0;
    }
    else
    {
        int sum = 0;
        struct node *temp = start->next;
        while (temp != NULL)
        {
            sum += temp->data;
            temp = temp->next;
        }
        return sum;
    }
}
// function to search for a given element in the list
int search_element(int data)
{
    if (start->next == NULL)
    {
        return -1;
    }
    else
    {
        int pos = 1;
        struct node *temp = start->next;
        while (temp != NULL)
        {
            if (temp->data == data)
            {
                return pos;
            }
            temp = temp->next;
            pos++;
        }
        return -1;
    }
}
// function to find the maximum element in the list
int max_element()
{
    if (start->next == NULL)
    {
        return -1;
    }
    else
    {
        int max = start->next->data;
        struct node *temp = start->next;
        while (temp != NULL)
        {
            if (temp->data > max)
            {
                max = temp->data;
            }
            temp = temp->next;
        }
        return max;
    }
}
// function to display the list
void display()
{
    if (start->next == NULL)
    {
        printf("\nList is empty\n");
    }
    else
    {
        printf("\nList is : ");
        struct node *temp = start->next;
        while (temp != NULL)
        {
            printf("%d-->", temp->data);
            temp = temp->next;
        }
        printf("NULL\n");
    }
}
int main()
{
    header = malloc(sizeof(struct node));
    header->data = 0;
    header->next = NULL;
    start = header;
    int element, position;
    int choice = 0;
    while (choice != 12)
    {
 printf("\n=====================================
MENU==========================================");
 printf("\n1- Insert at the beginning");
 printf("\t\t2- Insert at the end");
 printf("\t\t3- Insert at a given position");
 printf("\n4- Delete at the beginning");
 printf("\t\t5- Delete at the end");
 printf("\t\t6- Delete at a given position");
 printf("\n7- Total Number of Elements");
 printf("\t\t 8- Sum of all elements");
 printf("\t\t 9- Search an element");
 printf("\n10- Maximum element");
 printf("\t\t\t11- Display");
 printf("\t\t\t 12- Exit");
 printf("\n============================================================
=======================");
 printf("\nEnter your choice: ");
 scanf("%d", &choice);
 switch (choice)
 {
        case 1:
            printf("\nEnter the element to be inserted: ");
            scanf("%d", &element);
            insert_begin(element);
            display();
            break;
        case 2:
            printf("\nEnter the element to be inserted: ");
            scanf("%d", &element);
            insert_end(element);
            display();
            break;
        case 3:
            printf("\nEnter the element to be inserted: ");
            scanf("%d", &element);
            printf("\nEnter the position: ");
            scanf("%d", &position);
            insert_pos(element, position - 1);
            display();
            break;
        case 4:
            element = delete_begin();
            printf("\nDeleted element is %d\n", element);
            if (element != -1)
                display();
            break;
        case 5:
            element = delete_end();
            printf("\nDeleted element is %d\n", element);
            if (element != -1)
                display();
            break;
        case 6:
            printf("\nEnter the position: ");
            scanf("%d", &position);
            element = delete_pos(position - 1);
            printf("\nDeleted element is %d\n", element);
            if (element != -1)
                display();
            break;
        case 7:
            printf("\nTotal number of elements: %d", total_elements());
            display();
            break;
        case 8:
            printf("\nSum of all elements: %d", sum_elements());
            display();
            break;
        case 9:
            printf("\nEnter the element to be searched: ");
            scanf("%d", &element);
            if (search_element(element) == -1)
            {
                printf("\nElement not found");
            }
            else
            {
```

```c
                    printf("\nElement found at position %d",
search_element(element));
                }
                display();
                break;
            case 10:
                printf("\nMaximum element: %d", max_element());
                display();
                break;
            case 11:
                display();
                break;
            case 12:
                printf("\nExiting...");
                break;
            default:
                printf("\nInvalid choice!!!");
 }
    }
}
```

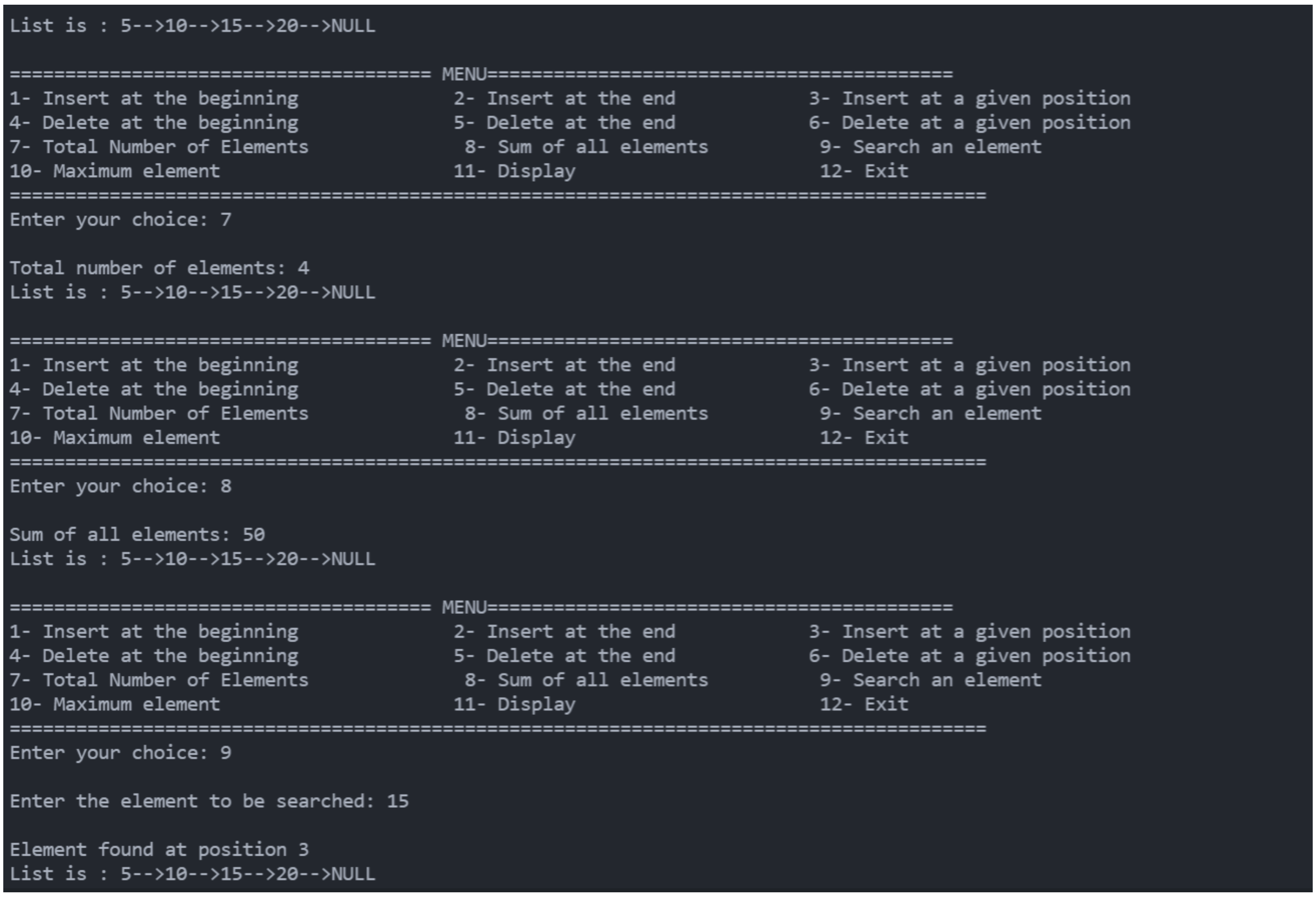
OUTPUT:



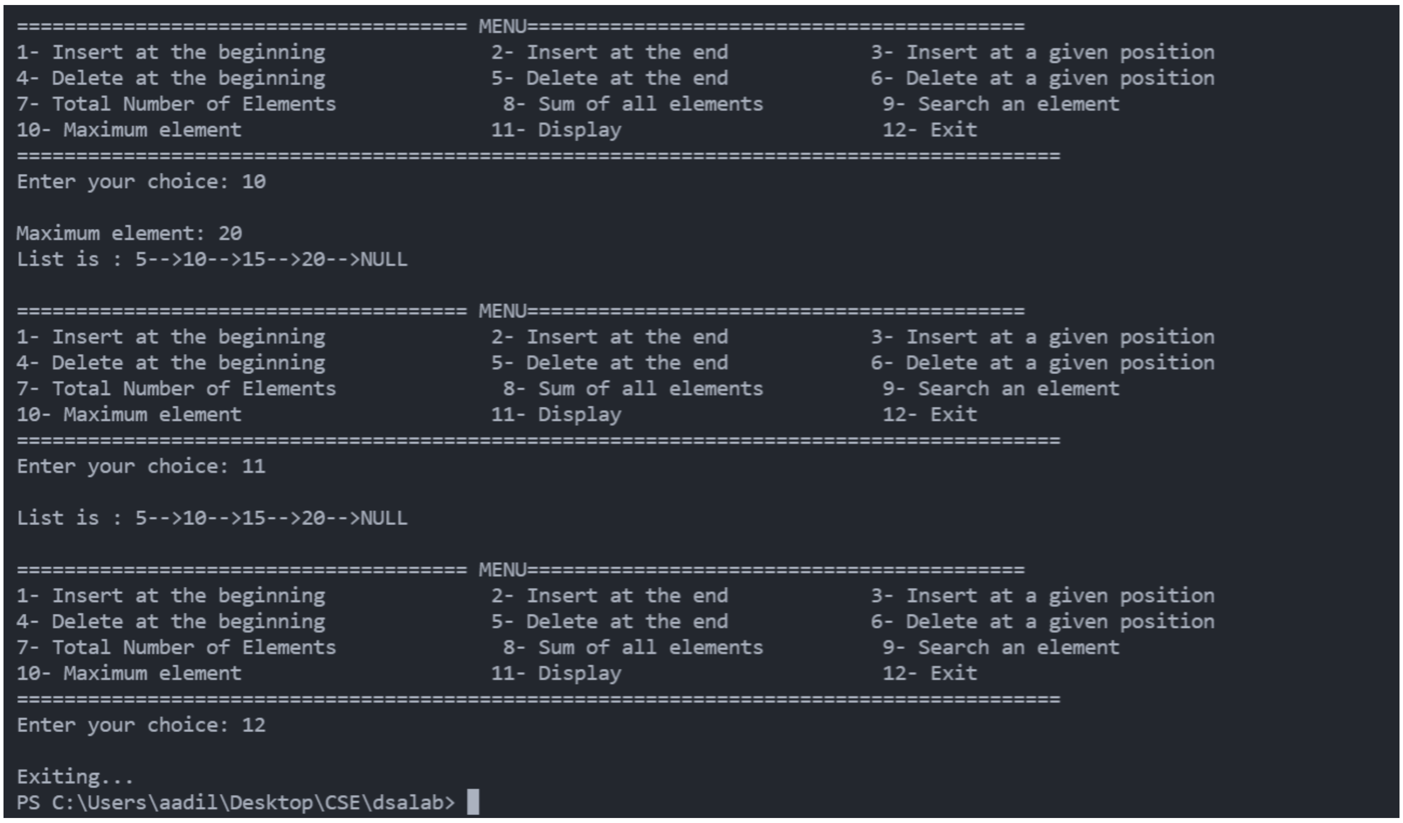
```
PS C:\Users\aadil\Desktop\CSE\dsalab>
                                        cd "c:\Users\aadil\Desktop\CSE\dsalab\" ; if ($?) { gcc dsa_A-1.c -o dsa_A-1 }

====================================== MENU=========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 1

Enter the element to be inserted: 5

List is : 5-->NULL

====================================== MENU=========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 2

Enter the element to be inserted: 10

List is : 5-->10-->NULL

====================================== MENU=========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
```



```
Enter your choice: 3

Enter the element to be inserted: 15

Enter the position: 3

List is : 5-->10-->15-->NULL

====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 4

Deleted element is 5

List is : 10-->15-->NULL

====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 5

Deleted element is 15

List is : 10-->NULL
====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 6

Enter the position: 1

Deleted element is 10

List is empty
```



```
List is : 5-->10-->15-->20-->NULL

====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 7

Total number of elements: 4
List is : 5-->10-->15-->20-->NULL

====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 8

Sum of all elements: 50
List is : 5-->10-->15-->20-->NULL

====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 9

Enter the element to be searched: 15

Element found at position 3
List is : 5-->10-->15-->20-->NULL
```



```
====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 10

Maximum element: 20
List is : 5-->10-->15-->20-->NULL

====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 11

List is : 5-->10-->15-->20-->NULL

====================================== MENU==========================================
1- Insert at the beginning              2- Insert at the end            3- Insert at a given position
4- Delete at the beginning              5- Delete at the end            6- Delete at a given position
7- Total Number of Elements              8- Sum of all elements          9- Search an element
10- Maximum element                     11- Display                      12- Exit
=========================================================================================
Enter your choice: 12

Exiting...
PS C:\Users\aadil\Desktop\CSE\dsalab>
```

Thank You...