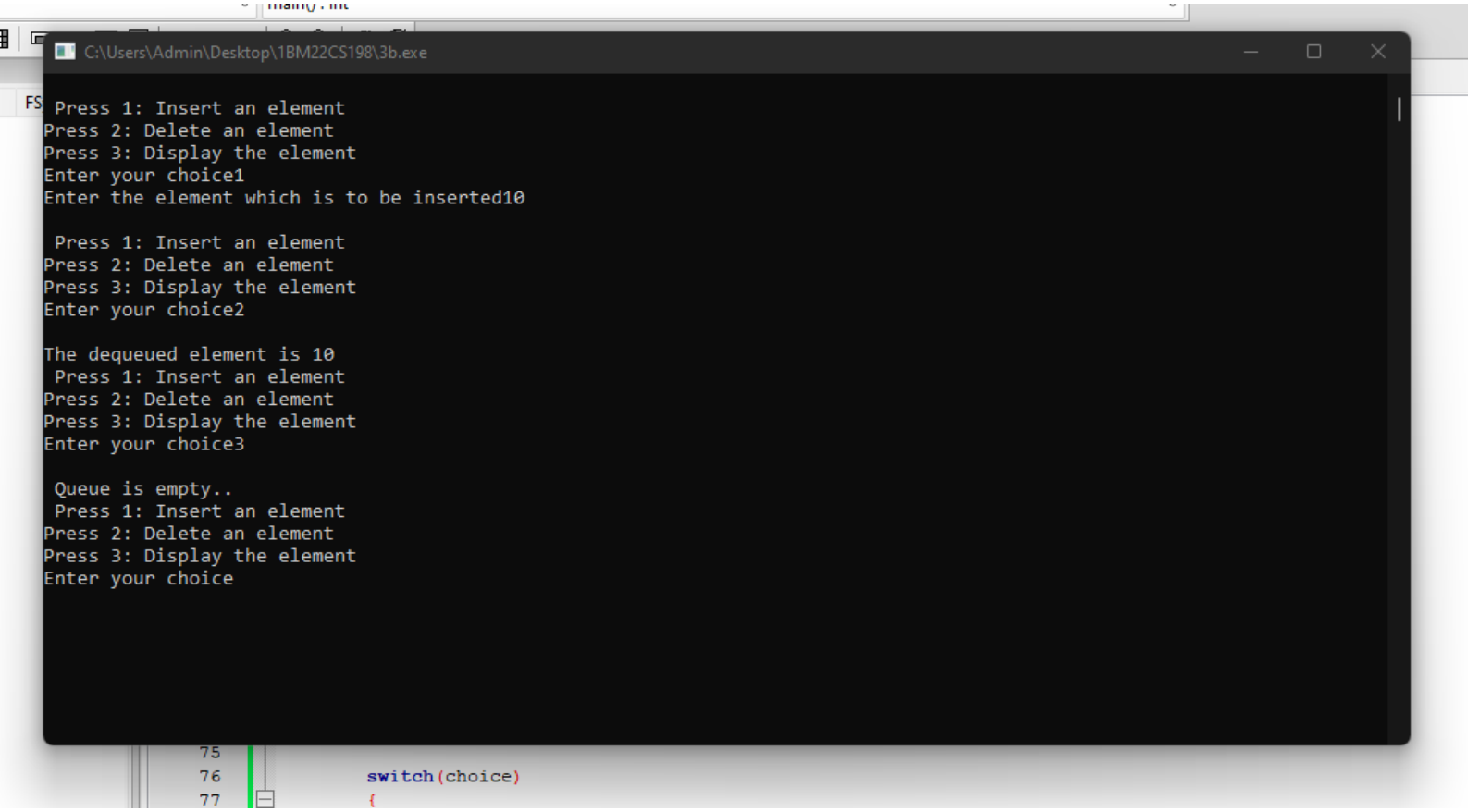
```c
#include <stdio.h>

# define max 6
int queue[max];
int front=-1;
int rear=-1;
void enqueue(int element)
{
    if(front==-1 && rear==-1)
    {
        front=0;
        rear=0;
        queue[rear]=element;
    }
    else if((rear+1)%max==front)
    {
        printf("Queue is overflow..");
    }
    else
    {
        rear=(rear+1)%max;
        queue[rear]=element;
    }
}

int dequeue()
{/ condition to check queue is empty
    if((front==-1) && (rear==-1))
    {
        printf("\nQueue is underflow..");
    }
 else if(front==rear)
{
   printf("\nThe dequeued element is %d", queue[front]);
   front=-1;
   rear=-1;
}
else
{
    printf("\nThe dequeued element is %d", queue[front]);
   front=(front+1)%max;
}
}
void display()
{
    int i=front;
    if(front==-1 && rear==-1)
```

```
    {
        printf("\nQueue is underflow..");
    }
 else if(front==rear)
{
   printf("\nThe dequeued element is %d", queue[front]);
   front=-1;
   rear=-1;
}
else
{
    printf("\nThe dequeued element is %d", queue[front]);
   front=(front+1)%max;
}
}
void display()
{
    int i=front;
    if(front==-1 && rear==-1)
    {
        printf("\n Queue is empty..");
    }
    else
    {
        printf("\nElements in a Queue are :");
        while(i<=rear)
        {
            printf("%d,", queue[i]);
            i=(i+1)%max;
        }
    }
}
int main()
{
    int choice=1,x;

    while(choice<4 && choice!=0)
    {
    printf("\n Press 1: Insert an element");
    printf("\nPress 2: Delete an element");
    printf("\nPress 3: Display the element");
    printf("\nEnter your choice");
    scanf("%d", &choice);

    switch(choice)
    {
```

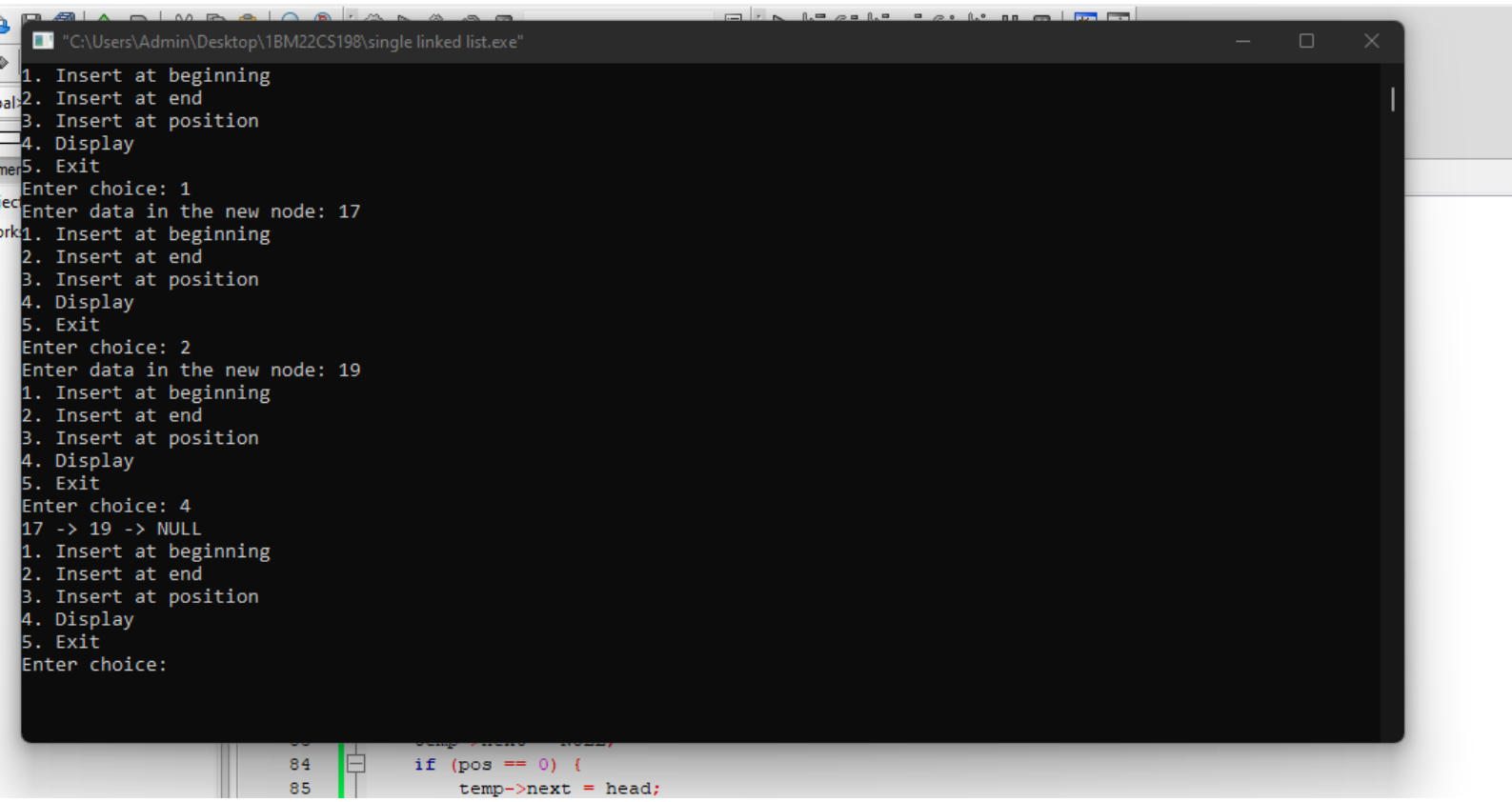
```
;
int main()
{
    int choice=1,x;

    while(choice<4 && choice!=0)
    {
    printf("\n Press 1: Insert an element");
    printf("\nPress 2: Delete an element");
    printf("\nPress 3: Display the element");
    printf("\nEnter your choice");
    scanf("%d", &choice);

    switch(choice)
    {

        case 1:

        printf("Enter the element which is to be inserted");
        scanf("%d", &x);
        enqueue(x);
        break;
        case 2:
        dequeue();
        break;
        case 3:
        display();

    }}
    return 0;
}
```


C:\Users\Admin\Desktop\1BM22CS198\3b.exe
Press 1: Insert an element
Press 2: Delete an element
Press 3: Display the element
Enter your choice1
Enter the element which is to be inserted10
Press 1: Insert an element
Press 2: Delete an element
Press 3: Display the element
Enter your choice2
The dequeued element is 10
Press 1: Insert an element
Press 2: Delete an element
Press 3: Display the element
Enter your choice3
Queue is empty..
Press 1: Insert an element
Press 2: Delete an element
Press 3: Display the element
Enter your choice
75
76 switch(choice)
77 {

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

typedef struct Node {
    int data;
    struct Node* next;
} Node;

Node* head = NULL;

void push();
void append();
void insert();
void display();

int main() {
    int choice;
    while (1) {
        printf("1. Insert at beginning\n");
        printf("2. Insert at end\n");
        printf("3. Insert at position\n");
        printf("4. Display\n");
        printf("5. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                push();
                break;
            case 2:
                append();
                break;
            case 3:
                insert();
                break;
            case 4:
                display();
                break;
            default:
                printf("Exiting the program");
                return 0;
        }
    }
}

void push() {
```

```c
void push() {
    Node* temp = (Node*)malloc(sizeof(Node));
    int new_data;
    printf("Enter data in the new node: ");
    scanf("%d", &new_data);
    temp->data = new_data;
    temp->next = head;
    head = temp;
}

void append() {
    Node* temp = (Node*)malloc(sizeof(Node));
    int new_data;
    printf("Enter data in the new node: ");
    scanf("%d", &new_data);
    temp->data = new_data;
    temp->next = NULL;
    if (head == NULL) {
        head = temp;
        return;
    }
    Node* temp1 = head;
    while (temp1->next != NULL) {
        temp1 = temp1->next;
    }
    temp1->next = temp;
}

void insert() {
    Node* temp = (Node*)malloc(sizeof(Node));
    int new_data, pos;
    printf("Enter data in the new node: ");
    scanf("%d", &new_data);
    printf("Enter position of the new node: ");
    scanf("%d", &pos);
    temp->data = new_data;
    temp->next = NULL;
```

```
    }
    Node* templ = head;
    while (templ->next != NULL) {
        templ = templ->next;
    }
    templ->next = temp;
}

void insert() {
    Node* temp = (Node*)malloc(sizeof(Node));
    int new_data, pos;
    printf("Enter data in the new node: ");
    scanf("%d", &new_data);
    printf("Enter position of the new node: ");
    scanf("%d", &pos);
    temp->data = new_data;
    temp->next = NULL;
    if (pos == 0) {
        temp->next = head;
        head = temp;
        return;
    }
    Node* templ = head;
    while (pos--) {
        templ = templ->next;
    }
    Node* temp2 = templ->next;
    temp->next = temp2;
    templ->next = temp;
}

void display() {
    Node* templ = head;
    while (templ != NULL) {
        printf("%d -> ", templ->data);
        templ = templ->next;
    }
    printf("NULL\n");
}
```


"C:\Users\Admin\Desktop\1BM22CS198\single linked list.exe"
1. Insert at beginning
2. Insert at end
3. Insert at position
4. Display
5. Exit
Enter choice: 1
Enter data in the new node: 17
1. Insert at beginning
2. Insert at end
3. Insert at position
4. Display
5. Exit
Enter choice: 2
Enter data in the new node: 19
1. Insert at beginning
2. Insert at end
3. Insert at position
4. Display
5. Exit
Enter choice: 4
17 -> 19 -> NULL
1. Insert at beginning
2. Insert at end
3. Insert at position
4. Display
5. Exit
Enter choice:
84 if (pos == 0) {
85 temp->next = head;