EXPERIMENT 6

Stack Using Linked List

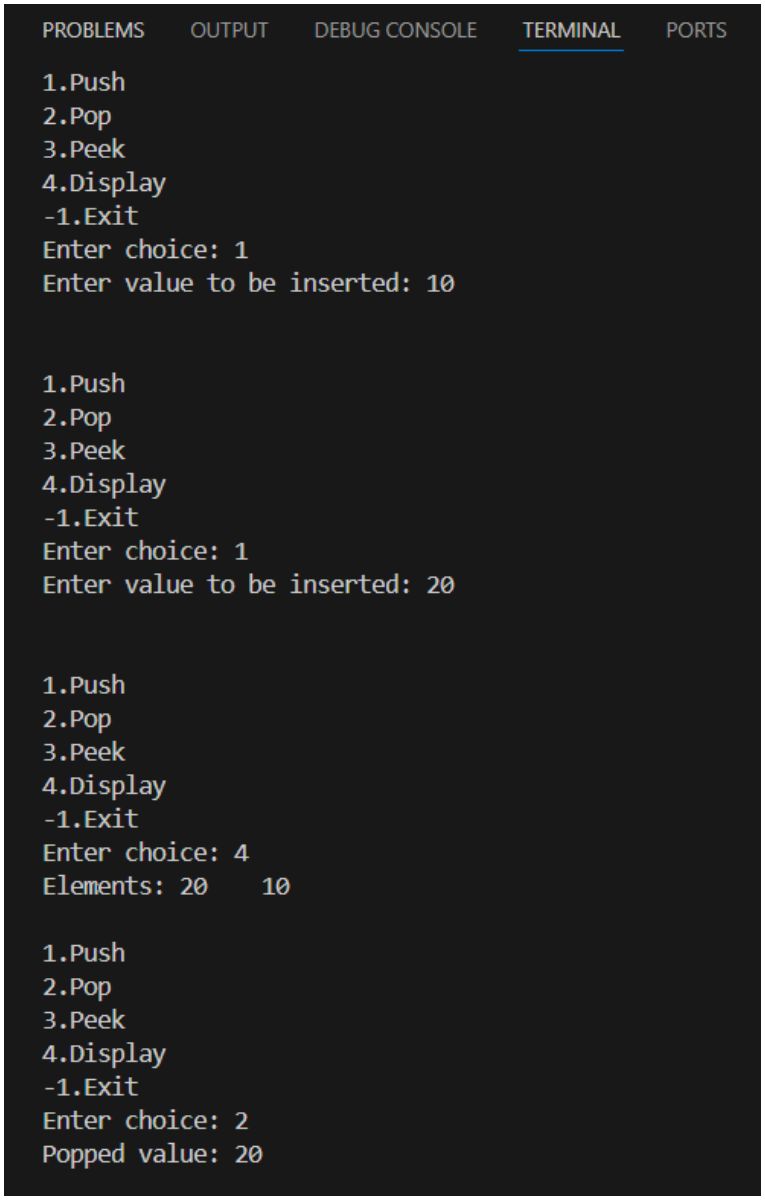
Code:

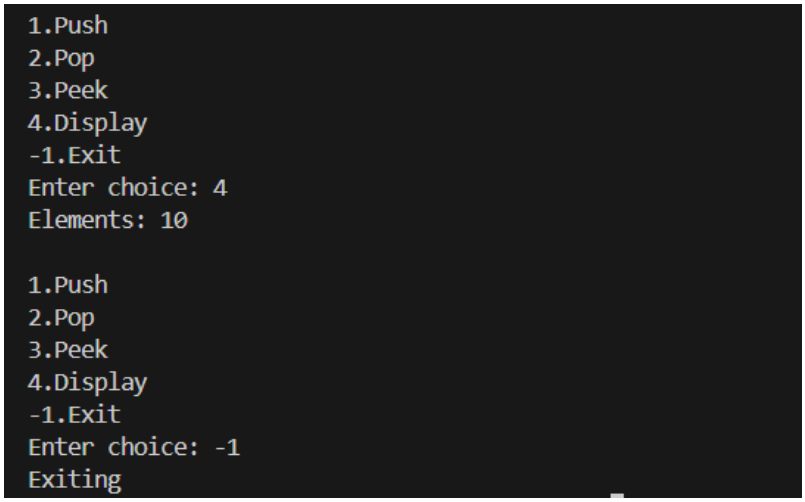
```
#include <stdio.h>
#include <stdlib.h>
struct Node{
    int data;
    struct Node *next;
}*top;
void push(){
    struct Node *nn;
    nn=(struct Node*)malloc(sizeof(struct Node));
    printf("Enter value to be inserted: ");
    scanf("%d",&nn->data);
    nn->next=top;
    top=nn;
}
int pop(){
    if(top==NULL){
        printf("Stack Underflow\n");
        return 0;
    }
    struct Node *temp;
    int val=top->data;
    temp=top;
    top=top->next;
    free(temp);
    return val;
}
int peek(){
    return top->data;
}
```

```
void display(){
    if(top==NULL){
        printf("No Elements to display");
    }
    else{
        printf("Elements: ");
        struct Node *temp;
        temp=top;
        while(temp!=NULL){
            printf("%d\t",temp->data);
            temp=temp->next;
        }
    }
}
void main(){
    int ch=0;
    while(ch!=-1){
        printf("\n\n1.Push\n2.Pop\n3.Peek\n4.Display\n-1.Exit\n");
        printf("Enter choice: ");
        scanf("%d",&ch);
        switch(ch){
            case 1:
            push();
            break;
            case 2:
            printf("Popped value: %d",pop());
            break;
            case 3:
            printf("Top value: %d",peek());
            break;
            case 4: display();
            break;
```

```
            case -1:
            printf("Exiting");
            break;
            default:
            printf("Invalid input");
        }
    }
}
```

Output:





Queue Using Linked List

Code:

```
#include <stdio.h>
#include <stdlib.h>
struct Node {
    int data;
    struct Node* next;
};
struct Node *front = NULL, *rear = NULL;
void enqueue() {
    int value;
    printf("Enter value to enqueue: ");
    scanf("%d", &value);
    struct Node* nn = (struct Node*) malloc(sizeof(struct Node));
    nn->data = value;
    nn->next = NULL;
    if (front == NULL) {
        front = rear = nn;
    } else {
        rear->next = nn;
        rear = nn;
    }
```

```
}
int dequeue() {
    if (front == NULL) {
        printf("Queue underflow! (Empty queue)\n");
        return -9999;
    }
    struct Node* temp = front;
    int value = temp->data;
    front = front->next;
    if (front == NULL)
        rear = NULL;
    free(temp);
    return value;
}
void display() {
    if (front == NULL) {
        printf("Queue is empty.\n");
        return;
    }
    struct Node* temp = front;
    printf("Queue elements: ");
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }
    printf("\n");
}
void main() {
    int ch, val;
    while (ch!=-1) {
        printf("1. Enqueue\n");
        printf("2. Dequeue\n");
```

```
        printf("3. Display\n");
        printf("-1. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &ch);
        switch (ch) {
        case 1:
            enqueue();
            break;
        case 2:
            val = dequeue();
            printf("Dequeued value: %d\n", val);
            break;
        case 3:
            display();
            break;
        case -1:
            printf("Exiting program...\n");
        default:
            printf("Invalid choice\n");
        }
    }
}
```

Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
E:\piyu\Computer Engg\Sem 3\DSA\Linked List>gcc queueLL.c

E:\piyu\Computer Engg\Sem 3\DSA\Linked List>.\a
1. Enqueue
2. Dequeue
3. Display
-1. Exit
Enter choice: 1
Enter value to enqueue: 40
1. Enqueue
2. Dequeue
3. Display
-1. Exit
Enter choice: 3
Queue elements: 40
1. Enqueue
2. Dequeue
3. Display
-1. Exit
Enter choice: 2
Dequeued value: 40
1. Enqueue
2. Dequeue
3. Display
-1. Exit
Enter choice: 3
Queue is empty.
1. Enqueue
2. Dequeue
3. Display
-1. Exit
Enter choice: -1
Exiting program...
Invalid choice
```