```c
// STACK IMPLEMENTATION USING ARRAY
#include <stdio.h>
#include <stdlib.h>
#define MAX 100

int stack[MAX];
int top = -1;

void push(int x) {
    if (top == MAX - 1) {
        printf("Stack Overflow\n");
    } else {
        stack[++top] = x;
        printf("%d pushed to stack.\n", x);
    }
}

void pop() {
    if (top == -1) {
        printf("Stack Underflow\n");
    } else {
        printf("Popped element: %d\n", stack[top--]);
    }
}

void peek() {
    if (top == -1) {
        printf("Stack is empty\n");
    } else {
        printf("Top element: %d\n", stack[top]);
    }
}

void display() {
    if (top == -1) {
        printf("Stack is empty\n");
    } else {
        printf("Stack elements: ");
        for (int i = top; i >= 0; i--) {
            printf("%d ", stack[i]);
        }
        printf("\n");
    }
}

int main() {
    int choice, value;
    while (1) {
        printf("\n--- Stack using Array ---\n");
        printf("1. Push\n2. Pop\n3. Peek\n4. Display\n5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
```

```
        case 1:
            printf("Enter value to push: ");
            scanf("%d", &value);
            push(value);
            break;
        case 2:
            pop();
            break;
        case 3:
            peek();
            break;
        case 4:
            display();
            break;
        case 5:
            exit(0);
        default:
            printf("Invalid choice\n");
        }
    }
    return 0;
}
```

```
// STACK IMPLEMENTATION USING LINKED LIST
#include <stdio.h>
#include <stdlib.h>

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

struct Node* top = NULL;

void push(int x) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    if (!newNode) {
        printf("Memory allocation failed\n");
        return;
    }
    newNode->data = x;
    newNode->next = top;
    top = newNode;
    printf("%d pushed to stack.\n", x);
}

void pop() {
    if (top == NULL) {
        printf("Stack Underflow\n");
    } else {
        struct Node* temp = top;
        printf("Popped element: %d\n", top->data);
        top = top->next;
        free(temp);
    }
}

void peek() {
    if (top == NULL) {
        printf("Stack is empty\n");
    } else {
        printf("Top element: %d\n", top->data);
    }
}

void display() {
    if (top == NULL) {
        printf("Stack is empty\n");
    } else {
        struct Node* temp = top;
        printf("Stack elements: ");
        while (temp != NULL) {
            printf("%d ", temp->data);
            temp = temp->next;
        }
        printf("\n");
```

```
    }
}

int main() {
    int choice, value;
    while (1) {
        printf("\n--- Stack using Linked List ---\n");
        printf("1. Push\n2. Pop\n3. Peek\n4. Display\n5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
        case 1:
            printf("Enter value to push: ");
            scanf("%d", &value);
            push(value);
            break;
        case 2:
            pop();
            break;
        case 3:
            peek();
            break;
        case 4:
            display();
            break;
        case 5:
            exit(0);
        default:
            printf("Invalid choice\n");
        }
    }
    return 0;
}
```