1. Write a programs for stack using array in c programming language.

```
#include <stdio.h>
#define MAX 100
#define USE_FUNCTIONS 1
#ifdef USE_FUNCTIONS
int stack[MAX];
int top = -1;
void push(int value) {
   if (top >= (MAX - 1)) {
      printf("Stack Overflow\n");
   } else {
      stack[++top] = value;
      printf("%d pushed to stack\n", value);
   }
}
void pop() {
   if (top < 0) {
      printf("Stack Underflow\n");
   } else {
      printf("%d popped from stack\n", stack[top--]);
   }
}
void peek() {
   if (top < 0) {
      printf("Stack is empty\n");
   } else {
      printf("Top element is %d\n", stack[top]);
   }
}
#else
int stack[MAX];
int top = -1;
#endif

int main() {

#ifdef USE_FUNCTIONS
   push(10);
   push(20);
   push(30);
   push(40);
   peek();
   pop();
   pop();
   peek();
   if (top < 0) {
      printf("Stack is empty\n");
   } else {
      printf("Stack contains:\n");
      for (int i = 0; i <= top; i++) {
         printf("%d\n", stack[i]);
      }
   }
#else
   if (top >= (MAX - 1)) {
      printf("Stack Overflow\n");
   } else {
      stack[++top] = 10;
      printf("10 pushed to stack\n");
   }
   if (top >= (MAX - 1)) {
      printf("Stack Overflow\n");
   } else {
      stack[++top] = 20;
```

```
        printf("20 pushed to stack\n");
    }
    if (top >= (MAX - 1)) {
        printf("Stack Overflow\n");
    } else {
        stack[++top] = 30;
        printf("30 pushed to stack\n");
    }
    if (top >= (MAX - 1)) {
        printf("Stack Overflow\n");
    } else {
        stack[++top] = 40;
        printf("40 pushed to stack\n");
    }
    if (top < 0) {
        printf("Stack is empty\n");
    } else {
        printf("Top element is %d\n", stack[top]);
    }
    if (top < 0) {
        printf("Stack Underflow\n");
    } else {
        printf("%d popped from stack\n", stack[top--]);
    }
    if (top < 0) {
        printf("Stack Underflow\n");
    } else {
        printf("%d popped from stack\n", stack[top--]);
    }
    if (top < 0) {
        printf("Stack is empty\n");
    } else {
        printf("Top element is %d\n", stack[top]);
    }
    if (top < 0) {
        printf("Stack is empty\n");
    } else {
        printf("Stack contains:\n");
        for (int i = 0; i <= top; i++) {
            printf("%d\n", stack[i]);
        }
    }
#endif
    return 0;
}
```

Sample ouput:

```
10 pushed to stack
20 pushed to stack
30 pushed to stack
40 pushed to stack
Top element is 40
40 popped from stack
30 popped from stack
Top element is 20
Stack contains:
10
20
```

2. Write a programs for stack using linked list in c programming language.

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

```
typedef struct Node {
    int data;
    struct Node* next;
} Node;
void push(Node** top_ref, int new_data);
int pop(Node** top_ref);
int peek(Node* top);
void display(Node* top);

int main() {
    Node* stack = NULL;
    push(&stack, 10);
    push(&stack, 20);
    push(&stack, 30);
    push(&stack, 40);
    printf("Stack after pushes: ");
    display(stack);
    printf("Top element is %d\n", peek(stack));
    printf("Popped element is %d\n", pop(&stack));
    printf("Popped element is %d\n", pop(&stack));
    printf("Stack after pops: ");
    display(stack);
    while (stack != NULL) {
        pop(&stack);
    }
    return 0;
}
void push(Node** top_ref, int new_data) {
    Node* new_node = (Node*)malloc(sizeof(Node));
    if (new_node == NULL) {
        printf("Memory allocation failed\n");
        exit(1);
    }
    new_node->data = new_data;
    new_node->next = *top_ref;
    *top_ref = new_node;
}
int pop(Node** top_ref) {
    if (*top_ref == NULL) {
        printf("Stack Underflow\n");
        return -1; // Indicate an error
    }
    Node* temp = *top_ref;
    int popped_data = temp->data;
    *top_ref = temp->next;
    free(temp);

    return popped_data;
}
int peek(Node* top) {
    if (top == NULL) {
        printf("Stack is empty\n");
        return -1; // Indicate an error
    }

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

}

Sample ouput:

```
Stack after pushes: 40 30 20 10
Top element is 40
Popped element is 40
Popped element is 30
Stack after pops: 20 10
```