Lab 2

1) Stack Implementations

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

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

#define MAX_SIZE 100

typedef struct {
    int items[MAX_SIZE];
    int top;
} stack;

// Initialising empty stack
void initStack(stack *stack) {
    stack->top = -1;
}

// checking if array is empty
int isEmpty(stack *stack) {
    return stack->top == -1;
}

// checking if stack is full
int isFull(stack *stack) {
    return stack->top == MAX_SIZE-1;
}

// pushing element onto the stack
void push(stack *stack, int value){
    if(isFull(stack)) {
        printf("stack overflow\n");
        return;
    }
    stack->items[++stack->top] = value;
}
```

```
// popping element from stack
int pop(stack * stack) {
    if (isEmpty(stack)){
        printf("stack underflow\n");
        exit(1);
    }
    return stack->items[stack->top--];
}

// peeking
int peek(stack * stack){
    if(isEmpty(stack)){
        printf("stack is empty");
        exit(1);
    }
    return stack->items[stack->top];
}

// displaying
void display(stack* stack) {
    if (isEmpty(stack)){
        printf("stack is empty\n");
        return;
    }
    printf("stack:");
    for(int i = stack->top; i >=0; i--){
        printf("%d", stack->items[i]);
    }
    printf("\n");
}
```

```
int main() {
    Stack stack;
    initStack(&stack);
    push(&stack, 10);
    push(&stack, 20);
    push(&stack, 30);
    display(&stack);
    printf("Peeked element : %d\n", peek(&stack));
    printf("Popped element : %d\n", pop(&stack));
    printf("Popped element : %d\n", pop(&stack));
    display(&stack);
    return 0;
}
```

O/P:

```
Stack : 30 20 10
Peeked element : 30
Popped element : 30
Popped element : 20
Stack : 10
```

⑧ Queue Implementation

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_SIZE 100
#define n 5
int front = -1, rear = -1;
int q[n];
void enqueue () {
    int d;
    if (rear == n-1) {
        printf ("overflow\n");
        return;
    }
    if (front == -1) {
        front = 0;
    }
    printf ("Enter value :");
    scanf ("%d", &d);
    q[++rear] = d;
}
void dequeue () {
    if (front == -1 || front > rear) {
        printf ("underflow\n");
        return;
    }
    int val = q[front++];
    printf ("Value dequeued is %d\n", val);
}
void display () {
    if (front == -1) {
        printf ("Queue empty\n");
        return;
    }
```

```
int i;
for(i= front; i<= rear; i++) {
        printf ("%d\n", q[i]);
}
}
void main () {
    enqueue();
    enqueue();
    enqueue();
    enqueue();
    enqueue();
    display();
    dequeue();
    dequeue();
    dequeue();
    dequeue;
    dequeue;
    display();
}
```

9/8/24

o/p: Enter value: 1
" : 2
" : 3
" : 4
" : 5

Overflow

1
2
3
4
5

Value dequeued is 1
" " 2
" " 3
" " 4
" " 5

Underflow Queue is empty.