**Data structures using c**

A data structure is a specialized format for organizing, processing, retrieving, and storing data. It defines the relationships among data items and the operations that can be performed on the data.

**Stacks:**

**Definition**: A stack is a linear data structure that follows the Last-In-First-Out (LIFO) principle.

**Characteristics**:

* Elements can only be added or removed from the top.
* Uses push (add) and pop (remove) operations.

**Basic Operations**:

**Push**: Add an element to the top.

**Pop**: Remove the top element.

**Peek**: View the top element without removing it.

**IsEmpty**: Check if the stack is empty.

**Syntax:**

#define MAX 1000

struct Stack {

int top;

int arr[MAX];

};

// Initialize stack

void initStack(struct Stack\* stack) {

stack->top = -1;

}

// Push operation

void push(struct Stack\* stack, int data) {

if (stack->top == MAX - 1) return; // Stack overflow

stack->arr[++stack->top] = data;

}

// Pop operation

int pop(struct Stack\* stack) {

if (stack->top == -1) return -1; // Stack underflow

return stack->arr[stack->top--];

}

// Peek operation

int peek(struct Stack\* stack) {

if (stack->top == -1) return -1;

return stack->arr[stack->top];

}

**Queues**

**Definition**: A queue is a linear data structure that follows the First-In-First-Out (FIFO) principle.

**Characteristics**:

Elements are added at the rear and removed from the front.

Uses enqueue (add) and dequeue (remove) operations.

**Basic Operations**:

* **Enqueue**: Add an element to the rear.
* **Dequeue**: Remove an element from the front.
* **Peek/Front**: View the front element without removing it.
* **IsEmpty**: Check if the queue is empty

**Syntax:**

#include <stdio.h>

#include <stdlib.h>

#define MAX 100

struct Queue {

int front, rear, size;

int arr[MAX];

};

// Function to initialize queue

void initQueue(struct Queue\* queue) {

queue->front = 0;

queue->rear = -1;

queue->size = 0;

}

// Function to check if queue is empty

int isEmpty(struct Queue\* queue) {

return queue->size == 0;

}

// Function to add an element

void enqueue(struct Queue\* queue, int data) {

if (queue->size == MAX) {

printf("Queue overflow\n");

return;

}

queue->rear = (queue->rear + 1) % MAX;