**WEEK 4**

**STACK**

#include <stdio.h>

#include <stdlib.h>

struct Node {

int data;

struct Node\* next;

};

struct StackLL {

struct Node\* top;

};

struct StackArray {

int\* array;

int top;

int capacity;

};

struct StackLL\* createStackLL() {

struct StackLL\* stack = (struct StackLL\*)malloc(sizeof(struct StackLL));

stack->top = NULL;

return stack;

}

struct StackArray\* createStackArray(int capacity) {

struct StackArray\* stack = (struct StackArray\*)malloc(sizeof(struct StackArray));

stack->capacity = capacity;

stack->top = -1;

stack->array = (int\*)malloc(stack->capacity \* sizeof(int));

return stack;

}

int isEmptyLL(struct StackLL\* stack) {

return stack->top == NULL;

}

int isEmptyArray(struct StackArray\* stack) {

return stack->top == -1;

}

void pushLL(struct StackLL\* stack, int data) {

struct Node\* newNode = (struct Node\*)malloc(sizeof(struct Node));

newNode->data = data;

newNode->next = stack->top;

stack->top = newNode;

}

void pushArray(struct StackArray\* stack, int data) {

if (stack->top == stack->capacity - 1) {

printf("Stack Overflow\n");

return;

}

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

}

int popLL(struct StackLL\* stack) {

if (isEmptyLL(stack)) {

printf("Stack Underflow\n");

return -1;

}

struct Node\* temp = stack->top;

int data = temp->data;

stack->top = stack->top->next;

free(temp);

return data;

}

int popArray(struct StackArray\* stack) {

if (isEmptyArray(stack)) {

printf("Stack Underflow\n");

return -1;

}

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

}

int peekLL(struct StackLL\* stack) {

if (isEmptyLL(stack)) {

printf("Stack is empty\n");

return -1;

}

return stack->top->data;

}

int peekArray(struct StackArray\* stack) {

if (isEmptyArray(stack)) {

printf("Stack is empty\n");

return -1;

}

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

}

void displayLL(struct StackLL\* stack) {

if (isEmptyLL(stack)) {

printf("Stack is empty\n");

return;

}

struct Node\* temp = stack->top;

printf("Elements in stack: ");

while (temp != NULL) {

printf("%d ", temp->data);

temp = temp->next;

}

printf("\n");

}

void displayArray(struct StackArray\* stack) {

if (isEmptyArray(stack)) {

printf("Stack is empty\n");

return;

}

printf("Elements in stack: ");

for (int i = stack->top; i >= 0; i--) {

printf("%d ", stack->array[i]);

}

printf("\n");

}

int main() {

// Test linked list implementation

struct StackLL\* stackLL = createStackLL();

pushLL(stackLL, 1);

pushLL(stackLL, 2);

pushLL(stackLL, 3);

displayLL(stackLL);

printf("Top element: %d\n", peekLL(stackLL));

printf("Popped element: %d\n", popLL(stackLL));

displayLL(stackLL);

struct StackArray\* stackArray = createStackArray(5);

pushArray(stackArray, 4);

pushArray(stackArray, 5);

pushArray(stackArray, 6);

displayArray(stackArray);

printf("Top element: %d\n", peekArray(stackArray));

printf("Popped element: %d\n", popArray(stackArray));

displayArray(stackArray);

return 0;

}