Stack in array

#include <stdio.h>

#include<stdlib.h>

#define size 5

int stack[size],top= -1;

void push(int val){

if(top>=size -1)

printf("Stack is full.\n");

else{

top++;

stack[top]=val;

printf("Inserted element: %d\n",stack[top]);

}

}

void pop(){

if(top<= -1)

{

printf("No element to delete.\n");

}

else

{

printf("The popped element = %d\n",stack[top]);

top--;

}

}

void show(){

if(top>=0){

printf("Stack Elements are:\n");

for(int i=0;i<=top;i++)

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

}

else

{

printf("Stack is Empty\n");

}

}

int main(int argc, char const \*argv[]) {

int ch,val;

do{

printf("\nMenu");

printf("\n1. PUSH");

printf("\n2. POP ");

printf("\n3. SHOW STACK");

printf("\n4. Exit");

printf("\nEnter your choice 1 to 4=");

scanf("%d",&ch);

switch (ch)

{

case 1:

printf("Enter the value to be pushed=");

scanf("%d",&val);

push(val);

break;

case 2:

pop();

break;

case 3:

show();

break;

case 4:

exit(0);

break;

default:

printf("Invalid choice!");

break;

}

}while (ch<=3);

return 0;

}

Stack in Link Singly list

#include <stdio.h>

#include <stdlib.h>

#include <limits.h> // For INT\_MIN

#define CAPACITY 10000 // Stack maximum capacity

// Define stack node structure

struct stack

{

int data;

struct stack \*next;

} \*top;

// Stack size

int size = 0;

/\* Function declaration to perform push and pop on stack \*/

void push(int element);

int pop();

int main()

{

int choice, data;

while(1)

{

/\* Menu \*/

printf("------------------------------------\n");

printf(" STACK IMPLEMENTATION PROGRAM \n");

printf("------------------------------------\n");

printf("1. Push\n");

printf("2. Pop\n");

printf("3. Size\n");

printf("4. Exit\n");

printf("------------------------------------\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch(choice)

{

case 1:

printf("Enter data to push into stack: ");

scanf("%d", &data);

// Push element to stack

push(data);

break;

case 2:

data = pop();

// If stack is not empty

if (data != INT\_MIN)

printf("Data => %d\n", data);

break;

case 3:

printf("Stack size: %d\n", size);

break;

case 4:

printf("Exiting from app.\n");

exit(0);

break;

default:

printf("Invalid choice, please try again.\n");

}

printf("\n\n");

}

return 0;

}

/\*\*

\* Functiont to push a new element in stack.

\*/

void push(int element)

{

// Check stack overflow

if (size >= CAPACITY)

{

printf("Stack Overflow, can't add more element to stack.\n");

return;

}

// Create a new node and push to stack

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

// Assign data to new node in stack

newNode->data = element;

// Next element after new node should be current top element

newNode->next = top;

// Make sure new node is always at top

top = newNode;

// Increase element count in stack

size++;

printf("Data pushed to stack.\n");

}

/\*\*

\* Function to pop element from top of stack.

\*/

int pop()

{

int data = 0;

struct stack \* topNode;

// Check stack underflow

if (size <= 0 || !top)

{

printf("Stack is empty.\n");

// Throw empty stack error/exception

// Since C does not have concept of exception

// Hence return minimum integer value as error value

// Later in code check if return value is INT\_MIN, then

// stack is empty

return INT\_MIN;

}

// Copy reference of stack top to some temp variable

// Since we need to delete current stack top and make

// Stack top its next element

topNode = top;

// Copy data from stack's top element

data = top->data;

// Move top to its next element

top = top->next;

// Delete the previous top most stack element from memory

free(topNode);

// Decrement stack size

size--;

return data;

}

Stack in character

#include <stdio.h>

#include<stdlib.h>

char stack[5];

int top= -1;

void push(char val){

if(top>=5-1)

printf("Stack is full.\n");

else{

top++;

stack[top]=val;

printf("Inserted element: %c",stack[top]);

}

}

void pop(){

if(top<=-1)

{

printf("No element to delete.\n");

}

else

{

printf("The popped element = %c",stack[top]);

top--;

}

}

void show(){

if(top>=0){

printf("Stack Elements are:\n");

for(int i=0;i<=top;i++)

printf("%c ",stack[i]);

}

else

{

printf("Stack is Empty\n");

}

}

int main() {

int ch;

char val;

do{

printf("\nMenu");

printf("\n1. PUSH");

printf("\n2. POP ");

printf("\n3. SHOW STACK");

printf("\n4. Exit");

printf("\nEnter your choice 1 to 4=");

scanf("%d",&ch);

switch (ch)

{

case 1:

printf("Enter a character: ");

scanf(" %c",&val);

// printf("You entered %c", val);

push(val);

break;

case 2:

pop();

break;

case 3:

show();

break;

case 4:

exit(0);

break;

default:

printf("Invalid choice!");

break;

}

}while (ch<=3);

return 0;

}