1. **Write a menu driven program in C to perform Linear Queue operations (Enqueue,Dequeue).**

**Program: prg3a.c**

#include <stdio.h>

#include <stdlib.h>

#define MAX 4

void displayStack();

void push();

void pop();

void peek();

int STACK[MAX], TOP = -1;

int main() {

int choice;

while (1) {

printf("\n\t--: STACK OPERATIONS :--\n\n");

printf(" 1. PUSH\n 2. POP\n 3. PEEK\n");

printf(" 4. DISPLAY\n 0. Exit\n");

printf("\nEnter corresponding numbers of your choice : ");

scanf("%d", &choice);

switch(choice) {

case 0:

printf("\n\tTHANK YOU FOR USING THE PROGRAM\n");

return 1;

case 1:

system("cls");

printf("\n\t-- PUSH ELEMENT IN STACK --\n\n");

push();

break;

case 2:

printf("\n\t-- POP ELEMENT FROM STACK --\n\n");

pop();

break;

case 3:

peek();

break;

case 4:

displayStack();

break;

default:

printf("\n\t!!! ERROR: Wrong Choice !!!\t");

}

printf("\n\nPress Enter to continue.... ");

fflush(stdin);

getchar();

system("cls");

}

printf("\n\nAfter calling all function, returned at main\n");

return 0;

}

void displayStack() {

int i;

if(TOP == -1) {

printf("\n\t!!! STACK IS EMPTY !!!\n\n");

return;

}

printf("\n\nThe STACK is : \n\n");

for(i = TOP; i >= 0; i--) {

printf("\n\t| %d |", STACK[i]);

if(i == TOP) {

printf(" <-- TOP");

}

}

}

void push() {

int i, item;

if(TOP == MAX - 1) {

printf("\n\t!!! STACK FULL. CAN'T INSERT NEW ELEMENT !!!\n\n");

return;

}

printf("Enter a element : ");

scanf("%d", &item);

TOP = TOP + 1;

STACK[TOP] = item;

printf("\n>> New element %d succefully entered in the STACK <<\n", item);

display();

}

void pop() {

int i;

if(TOP == - 1) {

printf("\n !!! STACK EMPTY. CAN'T DELETE ANY ELEMENT !!!\n\n");

return;

}

TOP = TOP - 1;

printf("\n>> Top element %d succefully deleted from the STACK <<\n", STACK[TOP + 1]);

display();

}

void peek() {

if(TOP == - 1) {

printf("\n !!! STACK EMPTY. CAN'T SHOW THE PEEK !!!\n\n");

return;

}

printf("\n >> THE TOP OF THE STACK IS = %d at INDEX = %d << \n", STACK[TOP], TOP);

}

**OUTPUT:**

Enter corresponding numbers of your choice : 1

-- PUSH ELEMENT IN STACK --

Enter a element : 30

>> New element 30 succefully entered in the STACK <<

The STACK is :

| 30 | <-- TOP

| 20 |

| 10 |

Press Enter to continue.…

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 1

-- PUSH ELEMENT IN STACK --

Enter a element : 40

>> New element 40 succefully entered in the STACK <<

The STACK is :

| 40 | <-- TOP

| 30 |

| 20 |

| 10 |

Press Enter to continue.…

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 1

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 1

-- PUSH ELEMENT IN STACK --

Enter a element : 10

>> New element 10 succefully entered in the STACK <<

The STACK is :

| 10 | <-- TOP

Press Enter to continue....

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 1

-- PUSH ELEMENT IN STACK --

Enter a element : 20

>> New element 20 succefully entered in the STACK <<

The STACK is :

| 20 | <-- TOP

| 10 |

Press Enter to continue.…

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

The STACK is :

| 30 | <-- TOP

| 20 |

| 10 |

Press Enter to continue....

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 2

-- POP ELEMENT FROM STACK --

>> Top element 30 succefully deleted from the STACK <<

The STACK is :

| 20 | <-- TOP

| 10 |

Press Enter to continue....

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 2

-- POP ELEMENT FROM STACK --

>> Top element 20 succefully deleted from the STACK <<

The STACK is :

| 10 | <-- TOP

Press Enter to continue.…

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

-- PUSH ELEMENT IN STACK --

!!! STACK FULL. CAN'T INSERT NEW ELEMENT !!!

Press Enter to continue.…

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 2

-- POP ELEMENT FROM STACK --

>> Top element 40 succefully deleted from the STACK <<

The STACK is :

| 30 | <-- TOP

| 20 |

| 10 |

Press Enter to continue.…

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 3

>> THE TOP OF THE STACK IS = 30 at INDEX = 0 <<

Press Enter to continue.…

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 4

Enter corresponding numbers of your choice : 2

-- POP ELEMENT FROM STACK --

>> Top element 10 succefully deleted from the STACK <<

Press Enter to continue....

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 2

-- POP ELEMENT FROM STACK --

!!! STACK EMPTY. CAN'T DELETE ANY ELEMENT !!!

Press Enter to continue.…

--: STACK OPERATIONS :--

1. PUSH

2. POP

3. PEEK

4. DISPLAY

0. Exit

Enter corresponding numbers of your choice : 0

THANK YOU FOR USING THE PROGRAM

1. **Write a menu driven program in C to perform Circular Queue operations (Enqueue, Dequeue).**

**Program: prg3b.c**

**OUTPUT:**