1. **Write a menu driven program in C to perform Stack operations (Push, Pop, Peek, Display) using user defined functions.**

Program: prg1.c

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

#define MAX_SIZE 4

void display();
void push();
void pop();
void peek();

int STACK[MAX_SIZE], 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 the 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:

                                display();

                                break;

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

                printf("\n\nPress Enter to continue.... ");
                fflush(stdin);
                getchar();
                system("cls");
        }

        return 0;
}


void display() {
        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_SIZE - 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:

```
--: STACK OPERATIONS :--

        1. PUSH
        2. POP
        3. PEEK
        4. DISPLAY
        0. Exit

Enter the corresponding numbers of your choice : 1

        -- PUSH ELEMENT IN STACK --

Enter a element : 10

>> New element 10 successfully 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 the corresponding numbers of your choice : 1

        -- PUSH ELEMENT IN STACK --

Enter a element : 20

>> New element 20 successfully 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

Enter the corresponding numbers of your choice : 1

        -- PUSH ELEMENT IN STACK --

Enter a element : 30

>> New element 30 successfully 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 the corresponding numbers of your choice : 1

        -- PUSH ELEMENT IN STACK --

Enter a element : 40

>> New element 40 successfully 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 the corresponding numbers of your choice : 1

        -- 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 the corresponding numbers of your choice : 2

        -- POP ELEMENT FROM STACK --

>> Top element 40 successfully 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 the 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 the corresponding numbers of your choice : 4

The STACK is :

                | 30 | <-- TOP
                | 20 |
                | 10 |

Press Enter to continue.…

        --: STACK OPERATIONS :--

  1. PUSH
  2. POP
  3. PEEK
  4. DISPLAY
  0. Exit

Enter the corresponding numbers of your choice : 2

        -- POP ELEMENT FROM STACK --

>> Top element 30 successfully 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 the corresponding numbers of your choice : 2

        -- POP ELEMENT FROM STACK --

>> Top element 20 successfully 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

Enter the corresponding numbers of your choice : 2

        -- POP ELEMENT FROM STACK --

>> Top element 10 successfully 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 the corresponding numbers of your choice : 0

        THANK YOU FOR USING THE PROGRAM
```

2. Write a menu driven program in C to perform Stack operations (Push, Pop, Peek, Display) using Structure data type.

Program: prg2.c

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

#define MAX_SIZE 4

void display();
void push();
void pop();
void peek();

struct _STACK_{
	int TOP, ARRAY[MAX_SIZE] ;
}stack;

int main() {
	int choice;

	stack.TOP = -1;

	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:

                        display();

                        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 display() {
        int i;

        if(stack.TOP == -1) {
                printf("\n\t!!! STACK IS EMPTY !!!\n\n");
                return;
        }

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

        for(i = stack.TOP; i >= 0; i--) {
                printf("\n\t| %d |", stack.ARRAY[i]);

                if(i == stack.TOP) {
                        printf(" <-- TOP");
                }
        }
}
```

```
void push() {
	int i, item;

	if(stack.TOP == MAX_SIZE - 1) {
		printf("\n\t!!! STACK FULL. CAN'T INSERT NEW ELEMENT !!!\n\n");
		return;
	}

	printf("Enter a element : ");
	scanf("%d", &stack.ARRAY[++stack.TOP]);

	printf("\n>> New element %d succefully entered in the STACK <<\n",
stack.ARRAY[stack.TOP]);

	display();
}

void pop() {
	int i;

	if(stack.TOP == - 1) {
		printf("\n !!! STACK EMPTY. CAN'T DELETE ANY ELEMENT !!!\n\n");
		return;
	}

	printf("\n>> Top element %d succefully deleted from the STACK <<\n",
stack.ARRAY[stack.TOP--]);

	display();
}

void peek(){
	if(stack.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.ARRAY[stack.TOP], stack.TOP);
}
```

OUTPUT:

```
        --: 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 : 12

>> New element 12 succefully entered in the STACK <<

The STACK is :

                | 12 | <-- 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 : 23

>> New element 23 succefully entered in the STACK <<

The STACK is :

                | 23 | <-- TOP
                | 12 |

Press Enter to continue.…

        --: STACK OPERATIONS :--

  1. PUSH
  2. POP
  3. PEEK
  4. DISPLAY
  0. Exit
```

```
Enter the corresponding numbers of your choice : 1

        -- PUSH ELEMENT IN STACK --

Enter a element : 34

>> New element 34 succefully entered in the STACK <<
The STACK is :

                | 34 | <-- TOP
                | 23 |
                | 12 |

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 : 45

>> New element 45 succefully entered in the STACK <<

The STACK is :

                | 45 | <-- TOP
                | 34 |
                | 23 |
                | 12 |

Press Enter to continue.…

        --: STACK OPERATIONS :--

  1. PUSH
  2. POP
  3. PEEK
  4. DISPLAY
  0. Exit

Enter the corresponding numbers of your choice : 1

        -- PUSH ELEMENT IN STACK --

Enter a element : 56
```

```
!!! 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 45 succefully deleted from the STACK <<
The STACK is :

                | 34 | <-- TOP
                | 23 |
                | 12 |

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 = 34 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

The STACK is :

                | 34 | <-- TOP
                | 23 |
                | 12 |

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 34 succefully deleted from the STACK
<<

The STACK is :

                | 23 | <-- TOP
                | 12 |

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 23 succefully deleted from the STACK
<<
The STACK is :

                | 12 | <-- TOP

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 12 succefully deleted from the STACK
<<
```

```
The STACK is :

                !!! STACK IS EMPTY !!!

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
```

3. Write a menu driven program in C to perform Stack operations (Push, Pop, Peek, Display) using Structure Pointer .

Program: prg3.c

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

#define MAX_SIZE 4

void display();
void push();
void pop();
void peek();

struct _STACK_ {
	int TOP, ARRAY[MAX_SIZE] ;
}*stack;

int main() {
	int choice;

	stack = (struct _STACK_ *) malloc(sizeof(struct _STACK_));

	stack->TOP = -1;

	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:
				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:

                        display();

                        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 display() {
        int i;

        if(stack->TOP == -1) {
                printf("\n\t!!! STACK IS EMPTY !!!\n\n");
                return;
        }

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

        for(i = stack->TOP; i >= 0; i--) {
                printf("\n\t| %d |", stack->ARRAY[i]);
```

```
            if(i == stack->TOP) {
                printf(" <-- TOP");
            }
        }
}

void push() {
        int i, item;

        if(stack->TOP == MAX_SIZE - 1) {
            printf("\n\t!!! STACK FULL. CAN'T INSERT NEW ELEMENT !!!\n\n");
            return;
        }

        printf("Enter a element : ");
        scanf("%d", &stack->ARRAY[++stack->TOP]);

        printf("\n>> New element %d succefully entered in the STACK <<\n", stack-
>ARRAY[stack->TOP]);

        display();
}

void pop() {
        int i;

        if(stack->TOP == - 1) {
            printf("\n !!! STACK EMPTY. CAN'T DELETE ANY ELEMENT !!!\n\n");
            return;
        }

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

        display();
}

void peek() {
        if(stack->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-
>ARRAY[stack->TOP], stack->TOP);
}
```

OUTPUT:

```
        --: 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 : 12

>> New element 12 succefully entered in the STACK <<

The STACK is :

                | 12 | <-- 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 : 23

>> New element 23 succefully entered in the STACK <<

The STACK is :

                | 23 | <-- TOP
                | 12 |

Press Enter to continue.…

        --: STACK OPERATIONS :--

 1. PUSH
 2. POP
 3. PEEK
 4. DISPLAY
 0. Exit
```

```
Enter the corresponding numbers of your choice : 1

        -- PUSH ELEMENT IN STACK --

Enter a element : 34

>> New element 34 succefully entered in the STACK <<
The STACK is :

                | 34 | <-- TOP
                | 23 |
                | 12 |

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 : 45

>> New element 45 succefully entered in the STACK <<

The STACK is :

                | 45 | <-- TOP
                | 34 |
                | 23 |
                | 12 |

Press Enter to continue.…

        --: STACK OPERATIONS :--

 1. PUSH
 2. POP
 3. PEEK
 4. DISPLAY
 0. Exit

Enter the corresponding numbers of your choice : 1

        -- PUSH ELEMENT IN STACK --

Enter a element : 56
```

```
!!! 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 45 succefully deleted from the STACK <<
The STACK is :

                | 34 | <-- TOP
                | 23 |
                | 12 |

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 = 34 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

The STACK is :

                | 34 | <-- TOP
                | 23 |
                | 12 |

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 34 succefully deleted from the STACK
<<

The STACK is :

                | 23 | <-- TOP
                | 12 |

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 23 succefully deleted from the STACK
<<
The STACK is :

                | 12 | <-- TOP

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 12 succefully deleted from the STACK
<<
```

```
The STACK is :

                !!! STACK IS EMPTY !!!

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
```