1.

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_SIZE 10
int stack[MAX_SIZE], top = -1;
void push(int value) {
  if (top == MAX_SIZE - 1) {
    printf("Stack overflow\n");
  }
  else {
    top++;
    stack[top] = value;
    printf("Pushed %d onto stack\n", value);
  }
}
void pop(){
  if (top == -1){
    printf("Stack underflow");
  }
  else {
    printf("Popped %d from stack\n", stack[top]);
    top--;
  }
}
void display() {
  if (top == -1){
    printf("Stack is empty");
  }
  else {
    printf("Elements:\n");
    for (int i=0; i<=top; i++) {
      printf("%d", stack[i]);
    }
    printf("\n");
  }
```

```
int main() {
  int choice, value;
  while True {
    printf ("Enter corresponding values: ");
    printf ("1. Push \n");
    print ("2. Pop \n");
    print ("3. Display \n");
    printf ("4. Exit \n");
    printf ("Enter your choice: ");
    scanf ("%d", &choice);

    switch (choice) {
      case 1: printf ("Enter value to push: ");
              scanf ("%d", &value);
              void push (int value);
              break;
      case 2: pop();
              break;
      case 3: display();
              break;
      case 4: printf ("Exit!");
              break;
      default: print ("Invalid choice \n");
  } }
  }
  return 0;
}
```

Output:

Enter values:

1. Push
2. Pop
3. Display
4. Exit

Enter your choice: 2

Stack underflow.

Programiz

C Online Compiler

main.c

Save Run

```c
#include <stdio.h>
#include <stdlib.h>
#define MAX 3 // Altering this value changes size of stack created
int st[MAX], top=-1;
void push(int st[], int val);
int pop(int st[]);
int peek(int st[]);
void display(int st[]);
int main(int argc, char *argv[]) {
int val, option;
do
{
printf("\n ***MAIN MENU****");
printf("\n 1. PUSH");
printf("\n 2. POP");
printf("\n 3. PEEK");
printf("\n 4. DISPLAY");
printf("\n 5. EXIT");
printf("\n Enter your option: ");
scanf("%d", &option);
switch(option)
{
case 1:
printf("\n Enter the number to be pushed on stack: ");
scanf("%d", &val);
```

Output

```
/tmp/znhNIvWhSG.o
****MAIN MENU****
 1. PUSH
 2. POP
 3. PEEK
 4. DISPLAY
 5. EXIT
Enter your option: 1
Enter the number to be pushed on stack: 20
****MAIN MENU****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 1
Enter the number to be pushed on stack: 40
****MAIN MENU****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option: 2
The value deleted from stack is: 40
```

2.

```
#include <stdio.h>
#include <string.h>
int prec(char c){
  if (c == '/' || c == '*')
    return 2;
  else if (c == '+' || c == '-')
    return 1;
  else
    return -1;
}
void infixtopostfix (char s[]){
  char stack[100], result[100];
  int top = -1, result_i = 0;
  int len = strlen(s);
  for(int i=0; i<len; i++){
    char c = s[i];
    if (c>='0' && c<='9'){
      result[result_i++] = c;
    }
    else if (c == '('){
      stack[++top] = c;
    }
    else if (c == ')'){
      while (top >= 0 && stack[top] != '('){
        result[result_i++] = stack[top--];
      }
      top--;
    }
    else {
      case '+':
      case '-':
      case '*':
      case '/': while (prec(stack[top]) >= prec(s[i]))
                {
                  top--;
                }
      while (top>=0 && (prec(s[i]) < prec(stack[top])
             || prec(s[i]) == prec(stack[top]))){
        result[result_i++] = stack[top--];
      }
      stack[++top] = c;
    }
  }
```

V. good!

Evaluated
NLP
4/1/2024

```
int main() {
    char exp[] = "a+b*c-d";
    infixtopostfix(exp);
    return 0;
}
```

```
while (top >= 0) {
    result[result_i++] = stack[top--];
}
result[result_i] = '\0';
printf("%s\n", result);
}
```

C Online Compiler

main.c

Save Run

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

// Function to return precedence of operators

int prec(char c) {

    if (c == '^')

        return 3;

    else if (c == '/' || c == '*')

        return 2;

    else if (c == '+' || c == '-')

        return 1;

    else

        return -1;
}

```

Output

```
/tmp/znhNIvWhSG.o
abcd^e-fgh*+^*+i-
```