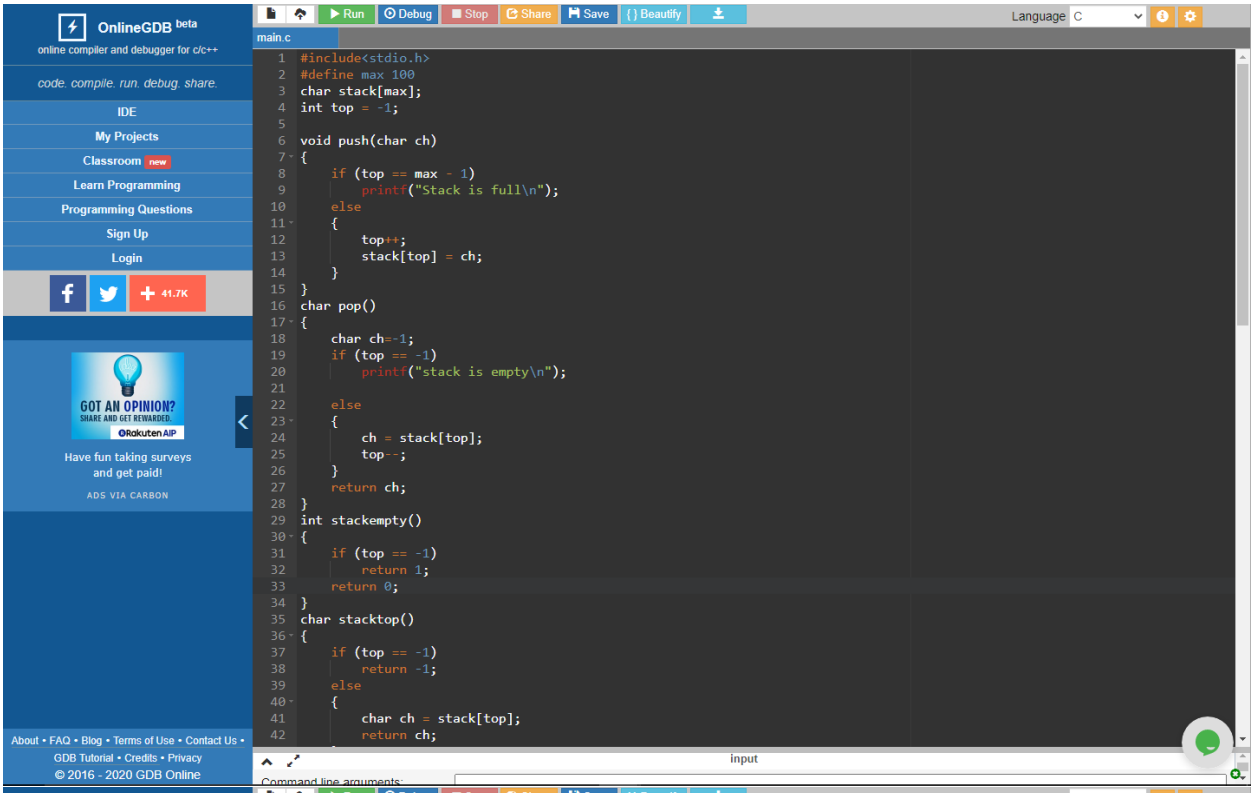
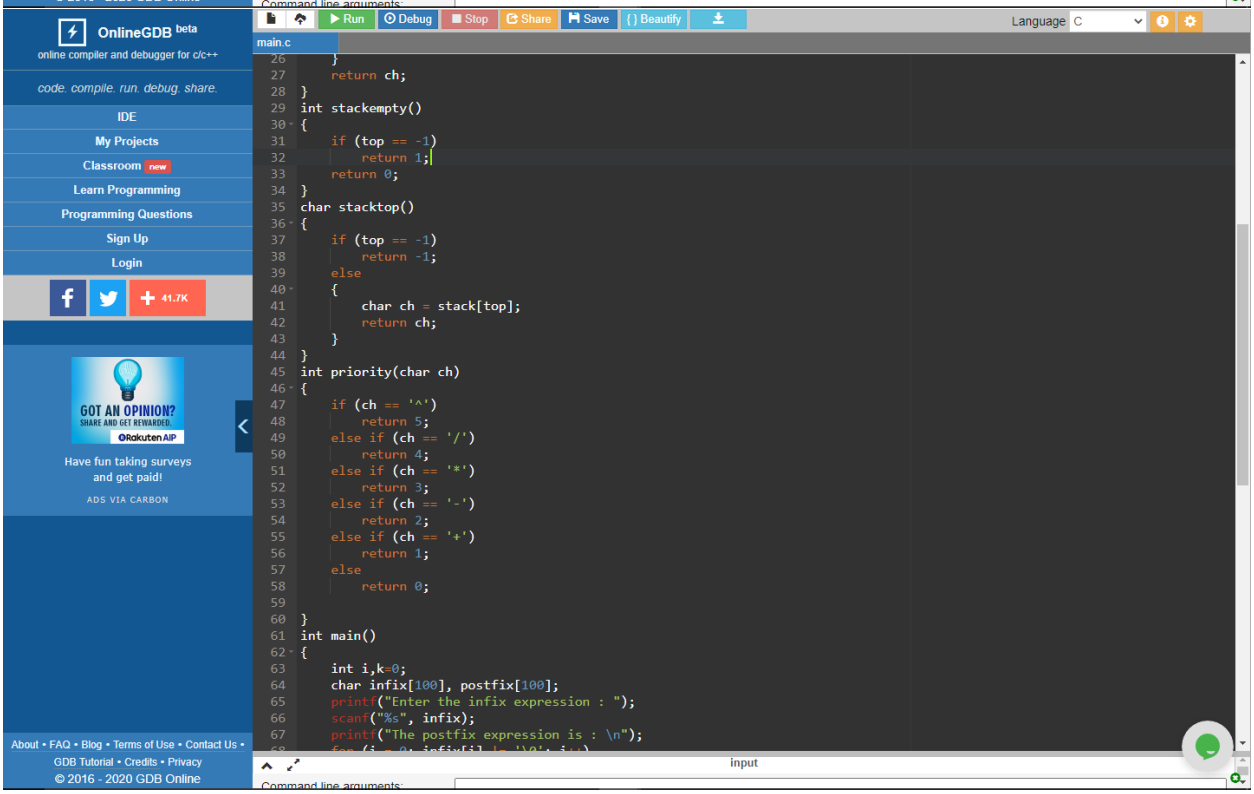

OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
41.7K
GOT AN OPINION?
SHARE AND GET REWARDED.
Rakuten AIP
Have fun taking surveys and get paid!
ADS VIA CARBON
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2020 GDB Online
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
main.c
#include<stdio.h>
#define max 100
char stack[max];
int top = -1;
void push(char ch)
{
if (top == max - 1)
printf("Stack is full\n");
else
{
top++;
stack[top] = ch;
}
}
char pop()
{
char ch=-1;
if (top == -1)
printf("stack is empty\n");
else
{
ch = stack[top];
top--;
}
return ch;
}
int stackempty()
{
if (top == -1)
return 1;
return 0;
}
char stacktop()
{
if (top == -1)
return -1;
else
{
char ch = stack[top];
return ch;
input
Command line arguments:


OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
Sign Up
Login
41.7K
GOT AN OPINION?
SHARE AND GET REWARDED.
Rakuten AIP
Have fun taking surveys and get paid!
ADS VIA CARBON
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits • Privacy
© 2016 - 2020 GDB Online
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
main.c
}
return ch;
}
int stackempty()
{
if (top == -1)
return 1;
return 0;
}
char stacktop()
{
if (top == -1)
return -1;
else
{
char ch = stack[top];
return ch;
}
}
int priority(char ch)
{
if (ch == '^')
return 5;
else if (ch == '/')
return 4;
else if (ch == '*')
return 3;
else if (ch == '-')
return 2;
else if (ch == '+')
return 1;
else
return 0;
}
int main()
{
int i,k=0;
char infix[100], postfix[100];
printf("Enter the infix expression : ");
scanf("%s", infix);
printf("The postfix expression is : \n");
input
Command line arguments:

Run | Debug | Stop | Share | Save | { } Beautify | Language C

main.c

```c
    char infix[100], postfix[100];
    printf("Enter the infix expression : ");
    scanf("%s", infix);
    printf("The postfix expression is : \n");
    for (i = 0; infix[i] != '\0'; i++)
    {
        if (infix[i] == '(')
        {
            push(infix[i]);
        }
        else if (infix[i] == ')')
        {
            while (stacktop() != '(')
            {
                postfix[k] = pop();
                k++;
            }
            pop();
        }
        else if (infix[i] == '^' || infix[i] == '/' || infix[i] == '*' || infix[i] == '-' || infix[i] == '+')
        {
            while(stackempty() == 0 && priority(stacktop()) >= priority(infix[i]))
            {
                postfix[k] = pop();
                k++;
            }
            push(infix[i]);
        }
        else
        {
            postfix[k] = infix[i];
            k++;
        }

    }
    while(!stackempty())
    {
        postfix[k] = pop();
                k++;
    }
    for(i=0;i<k;i++)
    {
```

input

Command line arguments:

Run | Debug | Stop | Share | Save | { } Beautify | Language C

main.c

```c
    printf("The postfix expression is : \n");
    for (i = 0; infix[i] != '\0'; i++)
    {
        if (infix[i] == '(')
        {
            push(infix[i]);
        }
        else if (infix[i] == ')')
        {
            while (stacktop() != '(')
            {
                postfix[k] = pop();
                k++;
            }
            pop();
        }
        else if (infix[i] == '^' || infix[i] == '/' || infix[i] == '*' || infix[i] == '-' || infix[i] == '+')
        {
            while(stackempty() == 0 && priority(stacktop()) >= priority(infix[i]))
            {
                postfix[k] = pop();
                k++;
            }
            push(infix[i]);
        }
        else
        {
            postfix[k] = infix[i];
            k++;
        }

    }
    while(!stackempty())
    {
        postfix[k] = pop();
                k++;
    }
    for(i=0;i<k;i++)
    {
        printf("%c",postfix[i]);
    }
}
```

input

Command line arguments: