**package** stack;//delete this line if your files are in the default package

//import Scanner class for string inputs from the user

**import** java.util.Scanner;

**import** java.util.Stack;

//Infix2Postfix class inherits Stack methods so you can use push(), pop() and peek()

**public** **class** Infix2Postfix **extends** Stack {

**public** String InToPost(String infixString) {

//declare and initialize postfixString

String postfixString = " ";

//now parse the infixString character by character

**for** (**int** index = 0; index < infixString.length(); ++index) {

**char** parseString = infixString.charAt(index);

/\* complete this code

\* if parseString is left brace, then push it to stack

\* else if the character is right brace:

\* a. Declare a variable of type char called topStack to store the top of stack

\* you might need to cast the top of stack to char using (char)

\* b. Write a while loop. while topStack is not left brace AND stack is not empty

\* 1- add topStack to postfixString

\* 2- pop the stack

\* 3- set topStack to store the new top of stack

\* exit the while then pop the stack

\* else if parseString is '+' or '-'

\* if the stack is empty, push parseString to stack

\* else declare topStack, store the top stack in topStack variable

\* while the stack is not empty OR topStack is not left brace OR topStack is not right brace

\* pop the stack

\* add topStack to postfixString

\* exit the while then push parseString to stack

\* else if parseString is '\*' or '/'

\* if the stack is empty, push parseString to stack

\* else declare topStack then store the top stack in topStack variable

\* while the stack is not empty AND topStack is not '+' AND topStack is not '-'

\* pop the stack

\* add topStack to postfixString

\* exit the while then push parseString to stack

\* else add parseString to postfixString

\*

\* while stack is empty

\* declare topStack variable to store top stack

\* if topStack is not left brace then pop the stack and add topStack to postfixString

\* exit the while loop

\*return postfixString. This is the return of InToPost method

\*

\*

\*/

**public** **static** **void** main(String[] args) {

Infix2Postfix mystack = **new** Infix2Postfix();

System.***out***.println("Type in an infix string followed by key");

Scanner scan = **new** Scanner(System.***in***);

String str = scan.next();

System.***out***.println("The Expression you have typed in infix form :\n"+str);

System.***out***.println("The Equivalent Postfix Expression is :\n"+mystack.InToPost(str));

}

}