Нам необходим цикл, чтобы читать и оценивать выражения. Он простой в использовании заявления BREAK чтобы закончить цикл в нужное время:

**Повторение до бесконечности:**

Получить ввод данных пользователя.

если первое число 0:

Вырваться из цикла

Найти значение данного выражения

Выведите на экран значения.

Получение введенных пользователем данных включает в себя чтение трех значений данных. Нам необходимо три переменные для хранения этих значений. Лучше всего проверить, является ли первое число 0 сразу после того мы читаем его, поэтому пользователю будет оставаться только набрать 0 для завершения программы, а не полное выражение, например, 0 + 0. "Repeat indefinitely" can be written as "while (true)":

while (true):

Let firstNum = TextIO.getDouble()

if firstNum is 0:

Break out of the loop

Let operator = TextIO.getChar()

Let secondNum = TextIO.getlnDouble()

Find the value of the expression

Display the value.

To evaluate the user's expression, we have to test the operator to find out which operation to compute. We can do this with either a multi-way if statement or with a switch statement. In the program below, I use a switch. The if statement would be:

if ( operator == '+' )

value = firstNum + secondNum;

else if ( operator == '-' )

value = firstNum - secondNum;

else if ( operator == '\*' )

value = firstNum \* secondNum;

else if ( operator == '/' )

value = firstNum / secondNum;

else {

TextIO.putln("Unknown operator: " + operator);

continue; // back to start of loop

}

The computer won't let you get away without the else part of the if statement or the default case in the switch, since that would leave a possibility that the variable, value, is not assigned a value before it is printed out.

This program could be improved by having it print out an error message if the user tries to divide by zero.

**The Solution**

public class SimpleCalculator {

/\* This program evaluates simple expressions such as 2 + 2

and 34.2 \* 7.81, consisting of a number, an operator,

and another number. The operators +, -, \*, / are allowed.

The program will read and evaluate expressions until

the user inputs a line that starts with the number 0.

\*/

public static void main(String[] args) {

double firstNum; // First number in the expression.

double secondNum; // Second number in the expression.

char operator; // The operator in the expression.

double value; // The value of the expression.

TextIO.putln("Enter expressions such as 2 + 2 or 34.2 \* 7.81");

TextIO.putln("using any of the operators +, -, \*, /.");

TextIO.putln("To end, enter a 0.");

TextIO.putln();

while (true) {

/\* Get user's input, ending program if first number is 0. \*/

TextIO.put("? ");

firstNum = TextIO.getDouble();

if (firstNum == 0)

break;

operator = TextIO.getChar();

secondNum = TextIO.getlnDouble();

/\* Compute the value of the expression. \*/

switch (operator) {

case '+':

value = firstNum + secondNum;

break;

case '-':

value = firstNum - secondNum;

break;

case '\*':

value = firstNum \* secondNum;

break;

case '/':

value = firstNum / secondNum;

break;

default:

TextIO.putln("Unknown operator: " + operator);

continue; // Back to start of loop!

} // end switch

/\* Display the value. \*/

TextIO.putln("Value is " + value);

TextIO.putln();

} // end while

TextIO.putln("Good bye");

} // end main()

} // end class