Wprowadzenie

Twoim zadaniem, jest zaimplementowanie ewaluatora wyrażeń logicznych. W naszym systemie wyrażenia logiczne mogą składać się z następujących podstawowych wyrażeń:

- Stałych (prawda, fałsz),
- Iloczynu logicznego (AND)
- Sumy logicznej (OR)
- Zaprzeczenia (NOT)

Dodatkowo, funkcjonalność powinna pozwalać na definiowanie zmiennych, których wartości można w każdym momencie zmienić. Do tej części powinno użyć się klasy ***BooleanContext,*** która zapamiętuję nazwy i wartości zmiennych.

Wyrażenie logiczne podawane jest do systemu w łańcuchu znaków (string) z następującymi zasadami:

- Każde wyrażenie podstawowe ma odstęp 1 spacji: (“ “)
- Iloczyn logiczny oznaczony jest przez string: “AND”
- Suma logiczna oznaczony jest przez string: “OR”
- Zaprzeczenie oznaczony jest przez string: “NOT”
- Stała (prawda,fałsz) oznaczona jest przez string: “1”,”0”
- Zmienna oznaczona jest przez jakikolwiek ciąg znaków (bez spacji), który nie koliduje z poprzednimi oznaczeniami.

Dodatkowo, wyrażenie zapisane jest w notacji prefiksowej (prefix). Notacja prefiksowa wymaga aby wszystkie operatory (np. AND), występowały przed swoimi operandami.
Przykładowy format wejściowy: "AND 1 0" oznacza wyrażenie (1 AND 0).
Wyrażenie w notacji prefiksowej, przetwarzane jest w klasie ***PrefixBooleanParser***, która zwraca wyrażenie gotowe do ewaluacji. Więcej przykładów znajduję sie w klasie ***ExampleBooleanExpressions***.

Twoje zadanie

Używając interface ***IBooleanExpression*** zaimplementuj następujące wyrażenia logiczne:

1) ***ConstantExpression***, reprezentuje stałą wartość (prawda lub fałsz). Użyj wbudowanego typu *bool.*
2) ***AndExpression*** (AND), przyjmuje dwa wyrażenia logiczne i liczy ich iloczyn logiczny
3) ***OrExpression*** (OR), przyjmuje dwa wyrażenia logiczne i liczy ich sumę logiczną
4) ***NotExpression*** (!)*,* przyjmuje jedno wyrażenie logiczne i liczy jego zaprzeczenie
5) ***VariableExpression****,* przyjmuje nazwę zmiennej (*string name*). Użyj funkcji *BooleanContext.GetVariable(name)* do obliczenia wartości danej zmiennej. Wartości zmiennych można zmieniać, nawet po utworzeniu wyrażenia logicznego. Jest to możliwe za pomocą funkcji *BooleanContext.SetVariable(name)*.
6) Przeciążanie funkcji ***ToString()*****,** tak aby każde z wyrażeń zostało wyróżnione swoim operatorem (patrz przykład). W przypadku ***ConstantExpression*** zwróć (“true”, albo “false”). W przypadku ***VariableExpression***, zwróć nazwę zmiennej.

Przetwarzanie notacji prefiksowej:

1) Uzupełnienie klasy ***PrefixBooleanParser*** tak aby zwracała wyrażenie logiczne przetworzone z notacji prefiksowej. Gdy format wyrażenia na wejściu Parsera jest nie poprawny, powinien wystąpić wyjątek ***SyntaxErrorException***.

2) Poprawnie wykonane zadanie, powinno przechodzić wszystkie testy (patrz Main).

Przykładowy output

```
Test Cases for Simple Boolean Expression
Expression: true
Evaluated: true

----------------------------------------
Expression: false
Evaluated: false

----------------------------------------
Expression: (true AND false)
Evaluated: false

----------------------------------------
Expression: (true AND true)
Evaluated: true

----------------------------------------
Expression: (true OR true)
Evaluated: true

----------------------------------------
Expression: (true OR false)
Evaluated: true

----------------------------------------
Expression: (false OR false)
Evaluated: false

----------------------------------------
Expression: !(false)
Evaluated: true

----------------------------------------
Expression: !(true)
Evaluated: false

----------------------------------------
Expression: ((true AND false) OR true)
Evaluated: true

----------------------------------------
Expression: ((true AND false) OR false)
Evaluated: false

----------------------------------------
Expression: ((true AND false) OR !(false))
Evaluated: true

----------------------------------------
Expression: (((true AND false) OR !(false)) AND true)
Evaluated: true
```

```
Test Cases for Invalid Prefix format
Corrently detected invalid format: AND
Corrently detected invalid format: 1 AND
Corrently detected invalid format: OR
Corrently detected invalid format: AND 1
Corrently detected invalid format: OR 1
Corrently detected invalid format: 1 2
Corrently detected invalid format: 1     2
Test Cases for Variable Boolean Expression
Setting Variable 'X' with value: True
Expression: X
Evaluated: true

----------------------------------------
Setting Variable 'X' with value: False
Expression: X
Evaluated: false

----------------------------------------
Setting Variable 'X' with value: False
Expression: ((X OR false) AND !(false))
Evaluated: false

----------------------------------------
Setting Variable 'X' with value: True
Expression: ((X OR false) AND !(false))
Evaluated: true

----------------------------------------
Setting Variable 'X' with value: True
Setting Variable 'Y' with value: True
Setting Variable 'Z' with value: True
Setting Variable 'W' with value: True
Expression: (((X AND Y) OR !(Z)) AND W)
Evaluated: true

----------------------------------------
Setting Variable 'X' with value: True
Setting Variable 'Y' with value: True
Setting Variable 'Z' with value: True
Setting Variable 'W' with value: False
Expression: (((X AND Y) OR !(Z)) AND W)
Evaluated: false
```