Test Expressió Booleana

Objecte prova

És un test unitari per testejar si la classe *ExpressióBooleana* funciona correctament.
Aquesta és una classe que fem servir per representar uns consulta del tipus expressió booleana. Els tests de la classe comproven que cadascuna de les funcions funcioni correctament.

Altres elements integrats a la prova

L'altre element integrat a la prova és l'estructura de dades *Tree*, que té un format molt simple, i el test del qual no explicarem, perquè és una classe que només conté *getters* i *setters*. Fem servir el *Tree* per estructurar l'expressió booleana de manera de fer-la facilment avaluable. Hi ha una funció que també treballa amb el *getter* de *Document*.

Valors estudiats

Totes les proves realitzades en aquest test són de caixa blanca. És a dir, la persona que ha programat la classe és la mateixa que la que l'ha testejat. Per tant, el programador coneix l'estructura interna del codi i sap tots els casos possibles que existeixen.
En aquest cas, la classe Expressió Booleana són

- *tradueixExpressio*

  L'objectiu d'aquest test és comprovar si la funció *tradueixExpressio* funciona correctament. Per fer-ho simplement s'ha introduït a la funció un cas de possible paràmetre que l'usuari podria introduir, i compara la traducció amb la traducció esperada. L'expressió d'entrada conté tots aquells elements que ens podem trobar dins una expressió (parèntesis,subconjunts,cometes,connectives lògiques i variables).
- *construeixArbre*

  L'objectiu d'aquest test és comprovar que un arbre es crea correctament. Per fer-ho crearem una nova expressió i en construirem l'arbre. Després cridarem la funció *getArbre*, que ens retorna un *string* amb l'arbre en preordre. El *string* es

compara amb el resultat esperat, que és un *string* que conté l'arbre correcte en preordre.

El cas escollit per construir l'arbre conté tots aquells elements que ens podem trobar dins una expressió (parèntesis,subconjunts,cometes,connectives lògiques i variables).

- *buscarInici*

  L'objectiu d'aquest test és comprovar que *buscarInici* trobi l'operador de meys prioritat dins una expressió. Per fer-ho crearem una nova expressió i en buscarem l'operador menys prioritari. Després, en comproovarem el resultat amb l'enter esperat.

  El cas escollit conté tots aquells elements que ens podem trobar dins una expressió (parèntesis,subconjunts,cometes,connectives lògiques i variables).

- *buscarIniciNonHiHa*

  L'objectiu d'aquest test és comprovar que *buscarInici* NO trobi l'operador de meys prioritat dins una expressió, ja que l'expressió només conté una variable, per tant *buscarInici* ha de retornar -1. Per fer-ho crearem una nova expressió i en buscarem l'operador menys prioritari. Després, en comprovarem el resultat amb l'enter esperat (-1).

- *evaluateRec*

  L'objectiu d'aquest test és comprovar que la funció *evaluateRec* avalui bé una expressió *booleana* donat el contingut d'un document. Per fer-ho s'ha creat una expressio i un *array* de *String* (que correspondria al contingut d'un document), seguidament s'ha avaluat el document segons l'expressió i s'ha comparat el booleà resultant amb el resultat esperat. S'ha fet testos a part per analitzar cadascuna de les casuístiques:

  - *evaluateRec1* Assegura que una expressió amb tots els possibles elements que aquesta pot contenir sigui *false*.
  - *evaluateRec2* Assegura que una expressió entre cometes s'avalui correctament.
  - *evaluateRec3* Assegura que una expressió en un subconjunt s'avalui correctament i sigui falsa.
  - *evaluateRec4* Assegura que una expressió en un subconjunt s'avalui correctament i sigui certa.

- *evaluateRec5* Assegura que una expressió entre parèntesis tingui en compte els parèntesis. Per fer-ho avaluarem una expressió per un document que sigui falsa però que sense els parèntesis seria certa.

- *getArbre*

  L'objectiu d'aquest test és comprovar que *getArbre* retorni l'arbre de l'*atribut* expressió correctament en preordre. Per fer-ho crearem una nova expressió, en crearem l'arbre i cridarem a *getArbre*. Després, en comprovarem el resultat amb un *string* que contindrà l'arbre correcte en preordre.