Package
sf_ia

sf_ia
Class Clause

```
java.lang.Object
   |
   +-sf_ia.Clause
```

public class **Clause**
extends java.lang.Object

Cette classe met en oeuvre une clause, qui est une disjonction de littéraux.
Elle est caractérisée par un ArrayList de littéraux **litterals** .
See Also:
`Litteral`

Field Summary

private	litterals

Constructor Summary

public	Clause() Constructeur par défaut
public	Clause(java.util.ArrayList litterals) Constructeur avec un paramètre

Method Summary

void	duplicateLitteralsAvoidance() Méthode permettant de supprimer les littéraux dupliqués dans la clause courante.
boolean	equal(Clause cl2) Méthode permettant de vérifier si la clause courante et la clause passée en paramètre sont égales.
java.util.ArrayList	getLitterals() Getter
boolean	paireReductible(Clause cl2) Méthode permettant de vérifier si la clause courante et la clause passée en paramètre sont une paire réductible.
Clause	resolvant(Clause cl2) Méthode permettant de trouver le résolvant de deux clauses (qui sont une paire réductible)
void	setLitterals(java.util.ArrayList litterals) Setter
java.lang.String	toString() Méthode de description d'une Clause

Methods inherited from class `java.lang.Object`

clone, equals, finalize, getClass, hashCode, notify, notifyAll, registerNatives, toString, wait, wait, wait

Fields

litterals

private java.util.ArrayList **litterals**

Constructors

Clause

public **Clause**()

Constructeur par défaut

Clause

public **Clause**(java.util.ArrayList litterals)

Constructeur avec un paramètre

Parameters:
litterals - ArrayList de littéraux

Methods

paireReductible

public boolean **paireReductible**(Clause cl2)

Méthode permettant de vérifier si la clause courante et la clause passée en paramètre sont une paire réductible.

Parameters:
cl2 - une Clause.

Returns:
true si les deux clauses sont une paire réductible et false sinon.

resolvant

public Clause **resolvant**(Clause cl2)

Méthode permettant de trouver le résolvant de deux clauses (qui sont une paire réductible)

Parameters:
cl2 - une clause

Returns:
la clause résolvante

duplicateLitteralsAvoidance

```
private void duplicateLitteralsAvoidance()
```

Méthode permettant de supprimer les littéraux dupliqués dans la clause courante.

getLitterals

```
public java.util.ArrayList getLitterals()
```

Getter

Returns:
un ArrayList des littéraux composant la clause.

setLitterals

```
public void setLitterals(java.util.ArrayList litterals)
```

Setter

Parameters:
`litterals` - un ArrayList de littéraux.

equal

```
public boolean equal(Clause cl2)
```

Méthode permettant de vérifier si la clause courante et la clause passée en paramètre sont égales.

Parameters:
`cl2` - une Clause.

Returns:
true si les deux clauses sont égales et false sinon.

toString

```
public java.lang.String toString()
```

Méthode de description d'une Clause

Returns:
une description de la clause courante

sf_ia
Class Graphe

```
java.lang.Object
   ┗-sf_ia.Graphe
```

public class **Graphe**
extends java.lang.Object

Une classe matérialisant la stucture de données adoptée pour notre TP.
Il s'agit d'un graphe non orienté. On a décidé de le représenter grâce aux
listes d'adjacence. Pour son implémentation en java, on a opté pour la structure
LinkedHashMap adjs , car il permet une gestion d'ordre des clés, qui est
l'ordre de leur insertion. La clé est **Sommet** et la valeur un **ArrayList de Sommets**.
See Also:
`java.util.LinkedHashMap`, `Sommet`

Field Summary

private	adjs

Constructor Summary

public	Graphe() Constructeur par défaut
public	Graphe(java.util.LinkedHashMap adjs) Un constructeur à un paramètre

Method Summary

void	addArc(Sommet s1, Sommet s2) Ajout d'une arrête reliant les sommets s1 et s2
void	addSommet(Clause clause) Méthode d'ajout d'un sommet au graphe
void	addSommet(Sommet s) Méthode d'ajout d'un sommet au graphe
java.util.Map	getAdjs() Getter
java.util.ArrayList	getAdjSommets(Sommet s) Méthode permettant de récupérer l'ensemble des sommets adjacents ,de adjs, au Sommet s passé en paramètre (la clé)
void	setAdjs(java.util.LinkedHashMap adjs) Setter
boolean	verifDoublons(Clause clause) Méthode permettant de vérifier si la clause passé en paramètre fait déjà parti des sommets du graphe.

Methods inherited from class `java.lang.Object`
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, registerNatives, toString, wait, wait, wait`

Fields

adjs

`private java.util.LinkedHashMap` **`adjs`**

Constructors

Graphe

`public` **`Graphe`**`()`

Constructeur par défaut

Graphe

`public` **`Graphe`**`(java.util.LinkedHashMap adjs)`

Un constructeur à un paramètre

Parameters:
`adjs` - un LinkdedHashMap

See Also:
`java.util.LinkedHashMap`

Methods

addSommet

`public void` **`addSommet`**`(`Clause `clause)`

Méthode d'ajout d'un sommet au graphe

Parameters:
`clause` - une Clause

See Also:
`Clause`

addSommet

`public void` **`addSommet`**`(`Sommet `s)`

Méthode d'ajout d'un sommet au graphe

Parameters:
`s` - un Sommet

(continued from last page)

See Also:
Sommet

---

addArc

```
public void addArc(Sommet s1,
          Sommet s2)
```

Ajout d'une arrête reliant les sommets s1 et s2

Parameters:
`s1` - un Sommet
`s2` - un Sommet

See Also:
Sommet

---

getAdjSommets

```
public java.util.ArrayList getAdjSommets(Sommet s)
```

Méthode permettant de récupérer l'ensemble des sommets adjacents ,de adjs, au Sommet s passé en paramètre (la clé)

Parameters:
`s` - un Sommet

Returns:
un arrayList de Sommet

See Also:
Sommet

---

getAdjs

```
public java.util.Map getAdjs()
```

Getter

Returns:
un Map

See Also:
`java.util.Map`

---

verifDoublons

```
public boolean verifDoublons(Clause clause)
```

Méthode permettant de vérifier si la clause passé en paramètre fait déjà parti des sommets du graphe.

Parameters:
`clause` - une Clause

Returns:
true si la clause est déjà existante dans le graphe et false sinon

See Also:

(continued from last page)

Clause

setAdjs

```
public void setAdjs(java.util.LinkedHashMap adjs)
```

Setter

Parameters:

`adjs` - un LinkedHashMap

sf_ia
Class Litteral

```
java.lang.Object
   |
   +-sf_ia.Litteral
```

public class **Litteral**
extends java.lang.Object

Cette classe représente un litteral, c'est à dire une proposition ou sa négation.
Elle est caractérisée par un tableau de char **l[]** de taille 2.

Field Summary

private	l

Constructor Summary

public	Litteral(char a) Constructeur à un paramètre, il s'agit ici d'une proposition.
public	Litteral(char a, char b) Constructeur à deux paramètres, il s'agit ici de la négation d'une proposition.

Method Summary

boolean	compLit(Litteral l2) Méthode permettant de vérifier si deux littéraux sont complémentaires.
boolean	equal(Litteral l2) Méthode permettant de vérifier si le Litteral courant et celui passé en paramètre sont égaux.
char[]	getL() Getter
void	setL(char[] l) Setter
java.lang.String	toString() Méthode de description d'un littéral.

Methods inherited from class `java.lang.Object`

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, registerNatives, toString, wait, wait, wait`

Fields

(continued from last page)

l

```
private char l
```

Constructors

Litteral

```
public Litteral(char a)
```

Constructeur à un paramètre, il s'agit ici d'une proposition.

Parameters:
a - un char

Litteral

```
public Litteral(char a,
                char b)
```

Constructeur à deux paramètres, il s'agit ici de la négation d'une proposition.

Parameters:
a - un char
b - un char

Methods

compLit

```
public boolean compLit(Litteral l2)
```

Méthode permettant de vérifier si deux littéraux sont complémentaires.

Parameters:
l2 - un Litteral

Returns:
true si les deux littéraux sont complémentaires et false sinon

---

getL

```
public char[] getL()
```

Getter

Returns:
le tableau l[] caractérisant le littéral.

---

setL

```
public void setL(char[] l)
```

Setter

(continued from last page)

Parameters:

`l` - un tableau permettant de mettre à jour (modifier) le littéral.

equal

```
public boolean equal(Litteral l2)
```

Méthode permettant de vérifier si le Litteral courant et celui passé en paramètre sont égaux.

Parameters:

`l2` - un littéral

Returns:

true si les deux littéraux sont égaux et false sinon.

toString

```
public java.lang.String toString()
```

Méthode de description d'un littéral.

Returns:

un description du littéral courant.

sf_ia
Class Sommet

```
java.lang.Object
   |
   +-sf_ia.Sommet
```

public class **Sommet**
extends java.lang.Object

Cette classe matérialise un sommet, dans la structure de données de graphe.
Caractérisée par une clause.
See Also:
Clause

Field Summary

private	cl

Constructor Summary

public	Sommet(Sommet s) Constructeur à un paramètre
public	Sommet(Clause clause) Constructeur à un paramètre

Method Summary

Clause	getCl() Getter
void	setCl(Clause cl) Setter

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, registerNatives, toString, wait, wait, wait

Fields

cl

```
private sf_ia.Clause cl
```

Constructors

(continued from last page)

Sommet

```
public Sommet(Sommet s)
```

Constructeur à un paramètre

Parameters:
`s` - un sommet

See Also:
`Sommet`

Sommet

```
public Sommet(Clause clause)
```

Constructeur à un paramètre

Parameters:
`clause` - une Clause

See Also:
`Clause`

Methods

getCl

```
public Clause getCl()
```

Getter

Returns:
Clause

See Also:
`Clause`

---

setCl

```
public void setCl(Clause cl)
```

Setter

Parameters:
`cl` - une Clause

See Also:
`Clause`

sf_ia
Class StrategieLineaire

```
java.lang.Object
   |
   +-sf_ia.StrategieLineaire
```

public class **StrategieLineaire**
extends java.lang.Object

Cette classe met en oeuvre la stratégie de résolution par réfutation linéaire.

Field Summary

static	b
	graphe

Constructor Summary

public	StrategieLineaire(Graphe graphe)

Method Summary

void	enregistrerClauses(java.util.ArrayList clauses) Enregistre les clauses de la KB et de !
static void	main(java.lang.String[] args)
void	resolutionLineaire() Méthode permettant de réaliser l'algorithme de la résolution par réfutation, avec stratégie de résolution linéaire.

Methods inherited from class `java.lang.Object`

`clone, equals, finalize, getClass, hashCode, notify, notifyAll, registerNatives, toString, wait, wait, wait`

Fields

graphe

`sf_ia.Graphe` **`graphe`**

b

`static boolean` **`b`**

(continued from last page)

Constructors

StrategieLineaire

```
public StrategieLineaire(Graphe graphe)
```

Methods

enregistrerClauses

```
public void enregistrerClauses(java.util.ArrayList clauses)
```

Enregistre les clauses de la KB et de !alpha en tant que clé de la hashMap

Parameters:

`clauses` - une ArrayList de Clause

See Also:

`Clause`

resolutionLineaire

```
public void resolutionLineaire()
```

Méthode permettant de réaliser l'algorithme de la résolution par réfutation,
avec stratégie de résolution linéaire.

main

```
public static void main(java.lang.String[] args)
```