# Compilation – TP 9 : Coloriage de graphe

Université Paris Diderot – Master 1

(2015-2016)

L'objectif de ce TP est d'implémenter une première version de l'algorithme de coloriage de graphe : pour le moment, on prend uniquement en compte les relations de conflit (en ignorant les relations de préférence). De plus, le graphe ne contient pour le moment que des nœuds variables et aucun registre physique. Le TP suivant introduira ce raffinement de l'algorithme.

Si vous n'avez pas fini le TP précédent, vous pouvez tout de même travailler sur ce TP car l'algorithme de coloriage de graphe est générique et testable indépendamment du reste du compilateur. Pour cela, il faut avoir compilé le compilateur avec la commande :

```
# make byte
```

Puis à la racine du projet, il faut lancer :

```
# ./flap.top
```

et vous obtiendrez un *toplevel* OCaml avec tous les modules de `flap` chargés par défaut. Dès lors, vous pourrez lancer les fonctions de tests directement depuis ce *toplevel* :

```
# open GraphColoring;;
# test ();;
```

**Exercice 1** *Coloriage de graphe (à la main)*

1. *Rappelez-vous l'algorithme de coloration de graphe vu en cours en coloriant manuellement le graphe suivant à l'aide de 2 couleurs :*

   ```
   a - b, a - c, a - d
   b - c, b - d
   c - d
   ```

2. *Écrire l'algorithme sur papier en pseudo-code.*

□

**Exercice 2** *Structure de données de graphe*

1. *Pourquoi peut-on dire que la structure de données de graphe est persistente (ou, dit autrement, purement fonctionnelle) ? En quoi est-ce intéressant pour notre algorithme de coloriage de graphe ?*
2. *Quelle opération manque-t-il à la structure de graphe fournie par le module* `Graph` *pour implémenter le coloriage de graphe ?*
3. *Implémentez cette opération manquante.*

□

**Exercice 3** *Coloriage de graphe*

1. *Qu'est-ce qu'un coloriage correct ?*
2. *Implémentez la fonction* `check_coloring`*.*
3. *Implémentez la fonction* `pick`*.*
4. *Implémentez la fonction* `colorize`*.*

□