# Taaaaaaaa, ta, taaaa, taaaa, taaa, ta, tatatatataaaaaaaaaaa

Marin Bougeret

Septembre 2011

Le but de ce TP est d'écrire le "jeu des allumettes" d'une émission hautement culturelle que je ne citerai pas. Pour être plus sérieux : regardez bien le "découpage" proposé (c'est à dire comment on casse un "gros TP" en plein de "petites" procédures séparées), car plus tard cela sera à votre tour de le faire !

Le plateau est modélisé par un entier, qui correspond au nombre d'allumettes restantes. Chaque joueur prend à son tour 1, 2, ou 3 allumettes, et le but du jeu est de ne pas prendre la dernière (rmq, si il reste 2 ou 3 allumettes, et que le joueur les prend toutes, il perd !).

## Q1

Ecrire une procédure `affiche_plateau(etat_plateau : in integer; hauteur_dessin : in integer)` ayant les spécifications suivantes :

Prérequis : aucun.

Action : affiche le plateau courant, en représentant les allumettes verticalement par `hauteur_dessin` symboles '*', et en séparant les allumettes par 2 espaces.

Par exemple, `affiche_plateau(5,3)` doit afficher :

```
* * * * *
* * * * *
* * * * *
```

## Q2

Ecrire une procédure `inviter_joueur(flag : in boolean)` ayant les spécifications suivantes :

Prérequis : aucun

Action : affiche "Joueur 1, à toi !" si `flag = TRUE`, et "Joueur 2, à toi !" sinon. De plus, cette procédure doit afficher un retour à la ligne (instruction new_line).

## Q3

Ecrire une procédure `saisir_coup (...)` ayant les spécifications suivantes :

Prérequis : aucun

Action : Fait saisir à l'utilisateur un entier représentant le prochain coup à jouer, et s'assure que le coup est légal, c'est à dire que le joueur ne prend pas strictement plus d'allumettes que le nombre restantes, et qu'il en prend au plus 3

Réfléchissez vous même à la liste des paramètres (et à leur modes).

## Q4

Ecrire une procédure `effectue_coup (etat_plateau : in out integer ; coup : in integer)` ayant les spécifications suivantes :

Prérequis : $coup \leq min(3, etat\_plateau)$

Action : Diminue le nombre d'allumettes restantes en fonction du paramètre `coup`.

**RMQ : notez l'intérêt de la spécification "dure" de la question précédente : cela nous donne une prérequis confortable. Autrement dit, il faut décider où on fait les vérifications, et ne les faire qu'une fois !**

## Q5

Ecrire une procédure `changer_joueur(flag : in out boolean)` ayant les spécifications suivantes :

Prérequis : aucun

Action : change le joueur courant

## Q6

Utiliser toutes ces procédures pour faire programme qui demande un nombre d'allumettes initial (on suppose que l'utilisateur rentre un nombre au moins égal à 1), et qui fait jouer alternativement les joueurs (en leur demandant "joueur $i$, à toi !") jusqu'à ce qu'un joueur perde. Le programme affiche alors "Bravo joueur $i$", où $i$ est le joueur gagnant.

## Q7 (bonus)

Modifier le programme précédent afin qu'il humilie tous ces concurrents de fort boyard[1] qui ne savent pas compter modulo 4, c'est à dire que l'ordinateur joue à la place d'un des deux joueurs, et gagne dès que cela est possible.

[1] Oups, je l'ai dit !

# Commentaires

(Bonne habitude) Remarquez que l'on essaye d'écrire de "petites" procédures (c'est à dire qui font peu de choses), elles sont alors faciles à spécifier. De plus, on sépare autant que possible l'affichage du reste. On évite par exemple une procédure qui demande au joueur de jouer, vérifie son coup, et l'effectue (mais, bien sûr, cela reste tout de même une question de goût).

De plus, remarquez que l'on n'a pas utilisé de variables globales (on aurait pu mettre *etat_plateau* et *joueur_courant* en variables globales..).