

Master bio-info : Java

Année 2011-2012

TD2

Exercice 1 [Tamagotchis] Le Tamagotchi est un animal de compagnie virtuel, japonais. Ce nom est un mot-valise créé à partir des mots tamago ("œuf") et de l'abréviation de watchi qui vient du mot anglais watch ("montre"). Le jeu consiste à simuler l'éducation d'un animal à l'aide d'une petite console miniature, de la taille d'une montre, dotée d'un programme informatique.

Un tamagotchi va donc être un objet qui vit (perd de l'énergie) et qui doit être alimenté pour survivre (augmenter son énergie). Le but est donc ici de réaliser un petit programme qui consiste à faire interagir l'utilisateur dans le but de maintenir en vie une petite colonie de tamagotchis ayant un comportement basique.

1. Écrivez une classe **Tamagotchi** qui contient les attributs **privés** suivants : **age**, **dureeDeVie**, **maxEnergie**, **energie** de type entier et **nom** de type **String**. Le constructeur prend comme paramètre une chaîne pour le nom du tamagotchi et initialise les autres attributs ainsi : **age** à zéro, et de façon aléatoire : **dureeDeVie** entre 9 et 14, **maxEnergie** entre 5 et 9 et **energie** entre 3 et **maxEnergie**.
2. Écrivez une méthode de signature **public void parler()** qui écrit à l'écran le nom du tamagotchi et son état de forme : "heureux", si l'attribut **energie** est supérieur à 5, ou "affamé" dans le cas contraire.
3. Écrivez une méthode de signature **public void manger()** qui augmente d'une valeur aléatoire, comprise entre 1 et 3, la valeur de l'attribut **energie** et affiche ensuite à l'écran un message de satisfaction. Si **energie** est déjà à son niveau maximum (**maxEnergie**) alors le tamagotchi n'a pas faim et un message doit montrer son mécontentement.

Attention à ne pas dépasser **maxEnergie** !

4. Écrivez une méthode de signature **public boolean ageLimite()** qui permet de tester si le tamagotchi a atteint son âge limite (**dureeDeVie**).
5. Écrivez une méthode de signature **public boolean vivre()** qui fonctionne suivant trois cas :
 - (a) ne fait rien et retourne **true** si les valeurs de **age** et de **dureeDeVie** sont égales.
 - (b) si **energie** ≤ 0 : affiche un message de fin ("je meurs") et retourne **false**
 - (c) si **energie** > 0 : augmente **age** de 1, réduit **energie** de 1, puis, si le tamagotchi a rejoint son âge limite, affiche un message de satisfaction ou dans le cas contraire fait simplement parler le tamagotchi (return true).
6. Écrivez un programme (méthode **main** située dans une classe **SimulTamagotchis**) qui crée **n** tamagotchis, la valeur **n** étant fournie par l'utilisateur. Le programme exécute en boucle les actions suivantes :
 - la méthode **vivre()** est appliquée sur tous les tamagotchis.
 - l'utilisateur a la possibilité de choisir le tamagotchi qu'il souhaite nourrir pour ce cycle.On sort de la boucle uniquement dans les deux conditions suivantes :
 - Gagné : tous les tamagotchis ont rejoint leur âge limite.
 - Perdu : un tamagotchi est mort de faim.Exemples d'exécution

```
>java SimulTamagotchis
Quel nom pour le nouveau tamagotchi : Pierre
Quel nom pour le nouveau tamagotchi : Paul
Quel nom pour le nouveau tamagotchi : Jacques
```

```
-----Cycle no 1 -----
Pierre : je suis affamé !
Paul : tout va bien !
Jacques : je suis affamé !
```

```
(0) Pierre (1) Paul (2) Jacques
Nourrir quel tamagotchi ? 1
```

```
Paul : je n'ai pas faim !!
```

```
-----Cycle no 2 -----
Pierre : je suis affamé !
Paul : tout va bien !
Jacques : je suis affamé !
```

```
(0) Pierre (1) Paul (2) Jacques
Nourrir quel tamagotchi ? 0
```

```
Pierre : Merci !
```

```
-----Cycle no 3 -----
Pierre : tout va bien !
Paul : je suis affamé !
Jacques : je suis affamé !
```

```
(0) Pierre (1) Paul (2) Jacques
Nourrir quel tamagotchi ? 1
```

```
Paul : Merci !
```

```
-----Cycle no 4 -----
Pierre : je suis affamé !
Paul : c'est gagné pour moi !
Jacques : je suis affamé !
```

```
(0) Pierre (2) Jacques
Nourrir quel tamagotchi ? 2
```

```
Jacques : Merci !
```

```
-----Cycle no 5 -----
```

```
Pierre : je meurs... Arrrggh !
```

```
PERDU !!
```