TP 7 : boucles `while`

Informatique Fondamentale (IF1)

Semaine du 3 décembre 2007

1 Boucles simples

1. Écrivez un programme `Encore` qui demande à l'utilisateur " encore ? " , et qui continue de lui poser la question tant que celui-ci lui répond " oui ".

2. La commande Unix " `yes` " affiche indéfiniment sur la console des lignes contenant le caractère " `y` ". Écrivez un programme Java `Yes` qui a le même comportement.

Vous pouvez interrompre ce programme en tapant `^C` (tenez la touche *Control* enfoncée pendant que vous tapez un C).

2 Algorithme d'Euclide

3. L'algorithme dit d'Euclide permet de calculer le pgcd de deux entiers positifs non nuls. Il est basé sur les propriétés suivantes :

- si $a < b$, alors $\text{pgcd}(a, b) = \text{pgcd}(b, a)$;
- si b divise a, alors $\text{pgcd}(a, b) = b$;
- sinon, $\text{pgcd}(a, b) = \text{pgcd}(b, r)$, où r est le reste de la division euclidienne de a par b.

Écrivez une fonction `pgcd` qui calcule le pgcd de ses deux arguments à l'aide de l'algorithme d'Euclide. Comme d'habitude, écrivez une fonction `main` qui vous permette de la tester.

3 Des approximations numériques

4. Étant donné un réel positif a, on définit la suite réelle $(x_n)_{n \in \mathbf{N}}$ de la manière suivante :

$$\begin{array}{rcl} x_0 & = & a \\ \text{pour tout } i \in \mathbf{N}, \quad x_{i+1} & = & \dfrac{x_i^2 + a}{2x_i} \end{array}$$

On admet que cette suite converge vers $\sqrt{a}$.

Écrivez un programme qui lit un réel positif a et un entier n et qui renvoie une valeur approchée b de $\sqrt{a}$ en utilisant l'approximation x_n. Affichez aussi b^2 pour vérifier.

5. Un étudiant place 1 zł[1] dans une banque. Cette somme sera rémunérée au taux de 100% à la fin de l'année[2], l'étudiant se retrouvera donc en possession de 2 zł. Un deuxième étudiant choisit de placer son Złoty dans une banque lui offrant un taux de rémunération de 50% tous les six mois. Ce dernier se retrouvera en possession de $2,25$ zł à la fin de l'année.

Écrivez un programme calculant ce que l'étudiant n, qui a placé son złoty dans une banque lui offrant un taux de rémunération de $1/n$ toutes les $1/n$ années, possède à la fin de l'année.

Votre programme calcule des valeurs approchées de e. Estimez-vous que la convergence est rapide ?

4 Recherche d'un entier dans un tableau (suite)

Pour répondre à cette section, reprennez la classe `Recherche` du TP 6.

Pour rechercher un entier dans un tableau `a`, lorsque ce tableau est trié, il est plus efficace de procéder en utilisant un algorithme appelé *recherche par dichotomie* que de procéder par une recherche linéaire. En effet, avec la méthode linéaire, pour savoir si aucun des éléments du tableau `a` n'est égal à `valeur` il faut comparer `valeur` successivement à tous les éléments du tableau. La méthode par dichotomie permet de réduire ce nombre de comparaisons.

La recherche par dichotomie manipule trois entiers p, m et g (pour " petit ", " moyen " et " grand "). À tout moment, on choisira ces entiers de tel sorte que $p \leq m \leq g$ et `a[`p`]` $\leq$ `valeur` $\leq$ `a[`g`]`.

Initialement, p vaut 0 et g la taille du tableau moins un. À chaque étape de calcul, m vaut $(p+g)/2$; si `a[`m`]` $\leq$ `valeur`, alors p devient m, sinon c'est g qui devient m. Le calcul s'arrête lorsque `a[`m`]` $=$ `valeur` ou lorsque $p = m$. Dans le second cas, aucun des éléments du tableau ne vaut `valeur`.

6. Écrivez une fonction :

```
int rechercheDichotomie(int valeur, int[] a)
```

qui utilise une recherche par dichotomie pour trouver un indice i tel que `a[`i`]` $=$ `valeur`, et retourne -1 si un tel indice n'existe pas.

7. Reprennez la fonction `main`, de telle sorte que si le tableau `a` est trié, elle utilise la méthode par dichotomie au lieu de la recherche linéaire.

[1] *Złoty*, l'unité monétaire polonaise.
[2] Exceptionnel, ou alors donnez-moi l'adresse de votre banque.