Semaine while

Matière

Les boucles *while* et *do while*

Objectifs

Savoir « écrire un programme » simple contenant des boucles *while* et *do while* à partir d'un énoncé en français.
Ecrire des répétitives avec des conditions composées.

Mise en garde :
Vous devez être capable d'écrire des répétitives avec une seule condition de continuation et sortie de boucle prématurée (*break, return*, …) mais aussi des répétitives avec des conditions composées sans sortie de boucle prématurée !

Contrainte :
Pour cette fiche, on vous demande de ne pas utiliser les *break, return*, … pour faire des sorties prématurées d'une boucle.

Pour les exercices de cette semaine, créez un nouveau projet IntelliJ et appelez-le ALGO_while

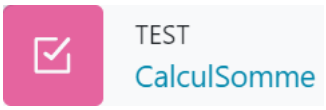
Exercices obligatoires

A « *while* » avec conditions d'arrêt simple

A1 Complétez la classe *CalculMoyenne* qui lit des cotes au clavier, calcule et affiche la moyenne.
Après chaque cote introduite, le programme demande à l'utilisateur s'il y en a une autre.
Si l'utilisateur répond par le caractère 'Y', l'encodage continue, sinon il s'arrête.

Pour lire un caractère au clavier : `scanner.next().charAt(0)`
(On lit une chaîne de caractères et ensuite on en prend le premier caractère. Le premier caractère se trouve à l'index `0`.)

Si vous avez du mal à vous lancer, refaites le test moodle :



(La solution de ce test est une classe qui calcule la somme. Pour la moyenne, il sera encore nécessaire de connaître le nombre de cotes.)

A2 Complétez la classe *MultiplicationAuFinish* qui propose à l'écran une multiplication de 2 nombres entiers tirés au hasard entre 0 et 10. L'utilisateur est invité à donner la réponse. Tant qu'il n'a pas donné la bonne réponse, le programme affiche « C'est faux, recommence » et l'utilisateur donne une autre réponse (toujours pour cette même multiplication).

B « *while* » avec conditions d'arrêt composées

B1 Complétez la classe *Multiplication3Essais* qui propose à l'écran une multiplication de 2 nombres entiers tirés au hasard entre 0 et 10. L'utilisateur est invité à donner la réponse. L'utilisateur a droit à maximum 3 essais pour trouver la bonne réponse.
Si l'utilisateur n'a pas donné la bonne réponse au bout de 3 essais, le programme l'affiche.

B2 Améliorez la classe *CalculMoyenne*. L'utilisateur répond 'Y', 'y', 'O' ou 'o' pour continuer !

B3 On vous demande d'écrire une classe *RecitationTableMultiplication* qui permet à un élève de s'exercer aux tables de multiplication.
Le programme choisit au hasard un entier compris entre 1 et 10. Cet entier correspond à la table de multiplication que l'élève va devoir faire. Le programme doit présenter à l'écran les multiplications à réaliser. Si l'élève se trompe, le programme donne la bonne réponse et s'arrête. Par contre, si l'élève n'a fait aucune erreur, le programme le félicite !
Voici 2 exemples de ce que pourrait être l'affichage à l'écran :

Tu vas donner la table de multiplication par 5 :
1 x 5 = 5
2 x 5 = 10
3 x 5 = 15
4 x 5 = 22
Non c'est faux, la bonne réponse est 20

Tu vas donner la table de multiplication par 3 :
1 x 3 = 3
2 x 3 = 6
3 x 3 = 9
4 x 3 = 12
5 x 3 = 15
6 x 3 = 18
7 x 3 = 21
8 x 3 = 24
9 x 3 = 27
10 x 3 = 30
Félicitation !

C Le jeu du nombre mystère

C1 Programmez un jeu dans lequel l'utilisateur doit deviner un nombre mystère tiré au hasard entre 0 et 100 (l'ordinateur n'affichera évidemment pas ce nombre).
A chaque essai, l'ordinateur demande à l'utilisateur un nombre et lui dit s'il est plus petit ou plus grand que le nombre à deviner. Le nombre d'essais est illimité.
Le jeu s'arrête lorsque l'utilisateur a deviné le nombre. Le programme affiche le nombre d'essais qu'il a fallu à l'utilisateur pour trouver la bonne réponse.
Nommez cette classe *NombreMystere*.

C2 Ecrivez le programme inverse de l'exercice précédent : c'est l'utilisateur qui choisit un nombre à deviner et c'est l'ordinateur qui doit le deviner. L'ordinateur proposera un nombre entre 0 et 100 et l'utilisateur répondra par le caractère '+' si le nombre à deviner est plus grand que le nombre proposé, '-' s'il est plus petit et '=' si c'est le bon nombre. A la fin le programme affichera le nombre d'essais qu'il a fallu à l'ordinateur pour trouver le bon nombre.
Appelez votre classe *NombreMystereInverse*
Conseil : repartez d'un programme vide. N'essayez pas de récupérer des bribes de solution de l'exercice précédent.

D Combat Guerriers : Combat à mort

Complétez la classe *CombatAMort* qui simule un combat entre deux guerriers.
CogneDur reçoit 9 points de vie et FrappeFort 12 points de vie.
A tour de rôle les guerriers se donnent un coup d'épée qui retire entre 1 et 6 points de vie à l'autre. C'est CogneDur qui frappe en premier.
Les dégâts infligés sont calculés dans votre programme à l'aide de la méthode `lancerDe()`.
Les deux guerriers étant entrés en mode Berserk ils vont se battre jusqu'à la mort de l'un d'entre eux.

Voici un exemple d'exécution :

```
Bienvenue au combat entre CogneDur et FrappeFort !

CogneDur inflige 1 points de degats a FrappeFort.
Il reste 11 points de vie a FrappeFort.

FrappeFort inflige 4 points de degats a CogneDur.
Il reste 5 points de vie a CogneDur.

CogneDur inflige 2 points de degats a FrappeFort.
Il reste 9 points de vie a FrappeFort.

FrappeFort inflige 5 points de degats a CogneDur.
CogneDur est mort. Paix a son ame, il est mort en brave.
FrappeFort est victorieux.
```

Exercice défi

C3 Améliorez la classe *NombreMystereInverse*.

- Faites en sorte que l'ordinateur soit le plus malin possible.
- L'ordinateur arrête le jeu s'il détecte que l'utilisateur le trompe !

Exercices supplémentaires

D Dans le projet IntelliJ **CombatGuerriers**, ajoutez la classe *CombatAMort_VI*. Utilisez la classe *FenetreGuerriers* pour rendre votre classe interactive.

A3 Écrivez un programme qui lit des nombres au clavier et s'arrête dès que l'on a entré cinq nombres strictement positifs.

A4 Limitation des dépenses :
Ecrivez un programme qui commence par demander à l'utilisateur quel est le montant maximum qu'il peut dépenser (ce nombre doit être positif). Ensuite le programme demandera à l'utilisateur d'entrer le montant de ses achats un à un. A chaque achat, le programme affichera le montant restant.

Le programme s'arrêtera lorsque la valeur du dernier achat entré au clavier sera supérieure au montant restant. Ce dernier achat ne sera pas pris en compte.
A la fin le programme affichera le montant restant.

Voici un exemple :

Budget : 100
Achats : 50 30 45
Montant restant : 20