Exercice 1 — Branchements conditionnels : Location de vélos

Le but de cet exercice est de permettre à un service de location de vélos (online, tournant 24 heures sur 24) de facturer ses clients.

Le programme demandera à l'utilisateur d'entrer les heures de début et de fin de location sous la forme d'entiers (on ne se préoccupe pas des minutes pour simplifier).

Les tarifs de location sont définis comme suit :

- 1€ par heure si le vélo est loué entre 0h et 7h ou entre 17h et 24h ;
- 2€ par heure si le vélo est loué entre 7h et 17h.

Votre programme demandera à l'utilisateur de quelle heure à quelle heure se fait la location et calculera le prix de la location en conséquence.

Vous adopterez les simplifications suivantes :

- les heures de début et fin de location sont des entiers (pas de demi ni de quart, toute heure entamée est due) ;
- l'heure du début de la location est toujours inférieure à l'heure de la fin de la location; cela implique que la location ne peut pas se faire sur plus de 24 heures ; elle doit se faire dans la même journée.

Si les données introduites sont correctes, votre programme affichera simplement le coût de la location en respectant les formats donnés dans les exemples de déroulement ci-dessous.

En cas de donnée incorrecte, votre programme devra afficher un message d'erreur et s'arrêter.

Utilisez les messages suivants :

- « Les heures doivent être comprises entre 0 et 24 ! », si une des heures introduites par l'utilisateur n'est pas comprise entre 0 et 24 (inclus) ;
- « Bizarre, vous n'avez pas loué votre vélo bien longtemps ! », si les heures de début et fin de location sont identiques ;
- « Bizarre, le début de la location est après la fin ... » si l'heure de début de la location est supérieure à l'heure de fin.

<u>Exemples de déroulement :</u>

Donnez l'heure de début de la location (un entier) : 10
Donnez l'heure de fin de la location (un entier) : 19
Vous avez loué votre vélo pendant
2 heure(s) au tarif horaire de 1.0 €
7 heure(s) au tarif horaire de 2.0 €
Le montant total à payer est de 16.0 €.

Donnez l'heure de début de la location (un entier) : 18
Donnez l'heure de fin de la location (un entier) : 20
Vous avez loué votre vélo pendant
2 heure(s) au tarif horaire de 1.0 €
Le montant total à payer est de 2.0 €.

Exercice 2 - Branchements conditionnels : Identification de champignons

1. Introduction

Le but de cet exercice est d'écrire un programme Java posant des questions à l'utilisateur pour deviner (parmi une liste connue à l'avance) à quel champignon pense l'utilisateur. Pour deviner un champignon, le programme ne peut poser que trois questions au maximum[1], dont la réponse est soit oui, soit non (l'utilisateur répondra aux questions du programme par false pour non, et par true pour oui; voir l'exemple de déroulement fourni plus bas).

Les 6 champignons possibles sont :

- l'agaric jaunissant;
- l'amanite tue-mouches;
- le cèpe de Bordeaux;
- le coprin chevelu;
- la girolle;
- et le pied bleu.

Seul le cèpe de Bordeaux possède des tubes, les autres champignons ayant des lamelles.
Le coprin chevelu et l'agaric jaunissant poussent dans les prés, les autres dans la forêt.
Les seuls à avoir un chapeau convexe sont l'agaric jaunissant, l'amanite tue-mouches et le pied bleu.
Enfin, les seuls à avoir un anneau sont l'agaric jaunissant, l'amanite tue-mouches et le coprin chevelu.

Commentaire: clavier.nextBoolean() permet de lire les booléens.

2. Instructions

Ecrire le programme de sorte à ce que le programme puisse trouver, en trois questions maximum le champignon auquel pense l'utilisateur (dans le cadre décrit plus haut).

Une des difficultés de cet exercice consiste à trouver quelles questions poser et dans quel ordre. Tous les ordres ne sont pas équivalents et ne conduisent pas à la solution en trois questions maximum.

Note: On suppose que l'utilisateur respecte les règles. Si les réponses de l'utilisateur sont incohérentes ou incorrectes, l'affichage du programme n'est pas spécifié, c.-à-d. qu'il peut être n'importe quoi suivant votre choix.

[1] Mais ce ne sont pas forcément les trois mêmes questions à chaque fois !

3. Exemples de déroulement

Attention, ces exemples ne sont là que pour donner une idée de ce qui se passe. Il n'est pas dit que les questions posées ici soient pertinentes ni dans le bon ordre pour garantir trois questions maximum! Il n'est pas dit non plus que ces deux exemples proviennent du même programme. Ce sont juste des exemples possibles d'interactions avec l'utilisateur.

Exemple1:

Pensez à un champignon : amanite tue mouches, pied bleu, girolle, cèpe de Bordeaux, coprin chevelu ou agaric jaunissant.

Est-ce que votre champignon a des lamelles (true : oui, false : non) ? true
Est-ce que votre champignon a un anneau (true : oui, false : non) ? false
Est-ce que votre champignon a un chapeau convexe (true : oui, false : non) ? false
==> Le champignon auquel vous pensez est la girolle.

Exemple2:

Pensez à un champignon : amanite tue mouches, pied bleu, girolle, cèpe de Bordeaux, coprin chevelu ou agaric jaunissant.

Est-ce que votre champignon vit en forêt (true : oui, false : non) ? true
Est-ce que votre champignon a des lamelles (true : oui, false : non) ? false
==> Le champignon auquel vous pensez est le cèpe de Bordeaux.

Exercice 3 – Boucles et itérations : Saut en parachute

1. Introduction

On s'intéresse ici à écrire un programme permettant de calculer les paramètres de la chute d'un parachutiste.

1) commencez par déclarer dans la méthode main 2 variables de type double :
   - masse pour la masse du parachutiste initialisée à 80.0
   - et h0 pour la hauteur de départ du parachutiste initialisée à 39000.0
2) Demander à l'utilisateur de saisir la masse du parachutiste tant que celle-ci n'est pas supérieure ou égale à 40.
3) Demander à l'utilisateur de saisir la hauteur de départ du parachutiste tant que celle-ci n'est pas supérieure ou égale à 250.

Exemple de déroulement :

masse du parachutiste (>=40) ?
30
masse du parachutiste (>=40) ?
50
hauteur de depart du parachutiste (>=250) ?
50
hauteur de depart du parachutiste (>=250) ?
300

2. Modélisation du parachutiste

Le parachutiste sera représenté dans le programme par sa masse, sa vitesse de chute, son accélération, son altitude et la surface de son corps exposée aux frottements de l'air (celle-ci variera lorsque le parachute s'ouvrira).

1) commencez par déclarer dans la méthode main une constante g de valeur 9.81, et deux variables de type double (qui seront modifiées lors de l'ouverture du parachute) : v0, initialisée à 0 et t0, initialisée à 0.
2) définissez ensuite les variables nécessaires à la description du parachutiste telle que donnée ci-dessus, autre que sa masse que nous avons déjà définie (voir le début de la méthode main dans le fichier téléchargé) : vitesse pour sa vitesse, hauteur pour son altitude, accel pour son accélération et t pour le temps. On initialisera la surface du parachutiste à 2.0 m², son altitude avec la valeur de h0, sa vitesse avec celle de v0 et son accélération avec celle de g. Définissez enfin une variable t initialisée à la valeur de t0.
3) Affichez les valeurs initiales telles que définies ci-dessus en utilisant la ligne suivante :
   System.out.printf("%.0f, %.4f, %.4f, %.5f\n", t, hauteur, vitesse, accel);
   Avec les valeurs initiales données ci-dessus, une masse de 80 kg et une altitude de départ de 39'000 m, le programme affichera à ce stade : 0, 39000.0000, 0.0000, 9.81000

3. Chute libre

Pour calculer l'évolution du sportif en chute libre nous aurons besoin des deux expressions suivantes:

- s qui est la surface du sportif divisée par sa masse;
- un « terme » noté q et valant q = exp(−s × (t − t0)), où t représente le temps courant et t0 le temps initial de la chute, initialisé à 0 dans la question précédente.
  Note: la fonction exp s'écrit simplement Math.exp en Java, par exemple: Math.exp(x).
  L'évolution du sportif en chute libre s'exprime alors comme suit :

$$v(t) = \frac{g}{s} \times (1 - q) + v0 \times q$$

$$h(t) = h0 - \frac{g}{s} \times (t - t0) - \frac{v0 - g/s}{s} \times (1 - q)$$

$$a(t) = g - s \times v(t)$$

où v est la vitesse du sportif, h son altitude, a son accélération, g = 9.81,et v0, h0 et t0 correspondent aux trois variables définies ci-dessus. On vous demande de compléter votre programme précédent de sorte à calculer l'évolution de la chute du sportif tel qu'initialisé dans la question précédente : faites le calcul, de seconde en seconde (c'est-à-dire ajouter à chaque fois 1 au temps t), tant que le sportif n'atteint pas le sol, c'est-à-dire tant que son altitude h est positive. Affichez les caractéristiques du sportif à chaque seconde en respectant le format de la question précédente. Testez votre programme avec une masse de 80 kg et une altitude de départ de 39'000 m; il devrait donner les résultats suivants :

0, 39000.0000, 0.0000, 9.81000
1, 38995.1356, 9.6884, 9.56779
2, 38980.7030, 19.1376, 9.33156
3, 38956.9382, 28.3535, 9.10116
...
137, 426.3065, 379.6277, 0.31931
138, 46.5205, 379.9430, 0.31142

4. Vitesse du son et vitesse limite

On vous demande maintenant d'étendre votre programme précédent de sorte que:

dès que la vitesse du sportif dépasse la vitesse du son (343 m/s), le programme affiche (en plus, mais qu'une seule fois) le message suivant : ## Felix depasse la vitesse du son Ce message doit s'afficher AVANT les informations de temps, altitude, vitesse et accélération :

...
82, 20498.5770, **341.8844**, 1.26289
Felix depasse la vitesse du son
83, 20156.0663, **343.1317**, 1.23171
...

dès que son accélération est inférieure à 0.5 m/s2, le programme affiche (en plus, mais qu'une seule fois) le message suivant : ## Felix a atteint sa vitesse maximale
Ce message doit s'afficher AVANT les informations de temps, altitude, vitesse et accélération :

...
119, 7199.1595, 372.3690, **0.50078**
Felix a atteint sa vitesse maximale
120, 6826.5422, 372.8636, **0.48841**
...

Pour tester : avec les valeurs précédentes (80 kg et 39'000 m), la vitesse du son est atteinte au bout de 83 s et la vitesse maximale (' 372 m/s) au bout de 120 s comme montré dans les deux exemples ci-dessus.

5. Ouverture du parachute

On vous demande finalement d'étendre une dernière fois votre programme précédent de sorte que dès que l'altitude du sportif est plus petite que 2500 m, le programme change la valeur de la surface du sportif de 2.0 m2 (avant l'ouverture du parachute) à 25.0 m2 (après l'ouverture du parachute). Il faut aussi changer les «conditions initiales» t0, v0 et h0 avec les valeurs actuelles du sportif (de sorte que les équations d'évolution soient correctes pour la suite de la chute). De plus, le programme doit afficher le message suivant : ## Felix ouvre son parachute

Ce message doit s'afficher AVANT les informations de temps, altitude, vitesse et accélération :

...
131, **2698.0264**, 377.5607, 0.37098
Felix ouvre son parachute
132, **2320.2818**, 377.9270, 0.36182
133, 1991.2751, 284.9225, -79.22827
...

Notez que donc l'accélération devient négative deux lignes après l'affichage de ce message.
Pour tester : avec les valeurs précédentes (80 kg et 39'000 m), le parachute est ouvert au bout de 132 s et la simulation se termine au bout de 170 s :

...
131, 2698.0264, 377.5607, 0.37098
Felix ouvre son parachute
132, 2320.2818, 377.9270, 0.36182
133, 1991.2751, 284.9225, -79.22827
...
170, 18.4814, 31.3944, -0.00075

Enfin, réfléchissez bien au moment où il faut signaler l'ouverture du parachute; par exemple, on ne saute pas avec un parachute déjà ouvert!..

Exercice 4 – Tableaux : Elément le plus fréquent dans un tableau

Le but de cet exercice est d'écrire un programme permettant d'identifier l'élément apparaissant le plus fréquemment dans un tableau d'entiers.

Ce programme devra également afficher le nombre d'occurrences dans le tableau de cet élément le plus fréquent. Par exemple, pour le tableau suivant :

{2, 7, 5, 6, 7, 1, 6, 2, 1, 7}

votre programme devra indiquer que l'élément le plus fréquent est le 7 et que sa fréquence d'apparition est 3.

Votre programme devra produire l'affichage suivant :

Le nombre le plus frequent dans le tableau est le :
7 (3 x)

Le code que vous écrirez devra pouvoir s'appliquer à n'importe quel tableau, mais vous pourrez supposer que ces tableaux sont toujours non vides.

Notez à ce propos que si dans un tableau donné il y a plus d'un nombre ayant le plus grand nombre d'occurrences, alors votre programme ne retiendra que celui qui apparaît en premier dans le tableau.

Par exemple, pour le tableau tab1 = {2, 7, 5, 6, 7, 1, 6, 2, 1, 7, 6}, où 6 et 7 sont tous deux les nombres les plus fréquents (les deux apparaissant 3 fois), votre programme ne retiendra que le 7 et affichera :

Le nombre le plus frequent dans le tableau est le :
7 (3 x)

Exercice 5 —Chaînes de caractères : Cryptage

Jules César utilisait un système de codage très simple, qui consiste à remplacer chaque lettre d'un message par la lettre placée plusieurs rangs après dans l'ordre alphabétique. Par exemple, pour un décalage de 4, A devient E, B devient F, jusque Z qui devient D.

Il s'agit ici d'appliquer cette technique pour coder une chaîne de caractère. Vous écrirez pour cela un programme qui met en œuvre les traitements décrits ci-dessous.

Pour cet exercice utilisez une chaîne de caractère constante ALPHABET contenant toutes les lettres de l'alphabet latin, en minuscule.

Le programme demande à l'utilisateur de saisir une chaîne de caractères s.

Le but de l'exercice est de construire une nouvelle chaîne aCoder comme suit :

- aCoder contient, dans l'ordre, tous les caractères de s qui sont présents dans ALPHABET ou qui correspondent au caractères espace (' '). Tous les autres caractères sont ignorés.

La chaîne aCoder sera codée au moyen de la technique de Jules César. Elle devra être affichée. Si elle ne contient aucun caractère, un message indiquant que la chaîne à coder est vide sera affiché.

Lorsque la chaîne à coder est non vide, votre programme devra la coder avec un décalage fixe donné par la constante fournie DECALAGE.

Codez la chaîne aCoder en remplaçant chaque caractère par celui situé DECALAGE lettres plus loin dans la chaîne de caractères ALPHABET, et cela de façon cyclique ('z' sera remplacé par 'd' par exemple). Les espaces seront maintenus tels quels. Il ne feront l'objet d'aucun codage mais resteront présents à l'endroit qu'ils occupaient.

La chaîne ainsi codée sera affichée.

L'exécution de votre programme devra strictement se conformer aux exemples suivants :

Veuillez entrer une chaine de caracteres :
fuyez manants
La chaine initiale etait : 'fuyez manants'
La chaine a coder est : 'fuyez manants'
La chaine codee est : 'jycid qererxw'

Veuillez entrer une chaine de caracteres :
avez-vous vu mes 3 chats et mes 2 chiens ?
La chaine initiale etait : 'avez-vous vu mes 3 chats et mes 2 chiens
La chaine a coder est : 'avezvous vu mes chats et mes chiens '
La chaine codee est : 'ezidzsyw zy qiw glexw ix qiw glmirw '

Veuillez entrer une chaine de caracteres :
93589()çç%&=+12AD
La chaine initiale etait : '93589()çç%&=+12AD'
La chaine a coder est vide.