INFORMATIQUE

Gestion d'un distributeur de boissons

On considère une machine automatique distribuant des boissons et rendant la monnaie. Cette machine permet à un utilisateur de choisir une boisson parmi un choix d'un maximum de dix variétés de boissons. Chaque boisson disponible a un prix qui ne dépasse pas cinq francs et qui est un multiple de dix centimes. La machine accepte les pièces de dix, vingt, cinquante centimes et d'un, deux et cinq francs. Dans tout le problème, les variables qui indiqueront des prix ou des sommes versées les exprimeront en centimes et seront donc toujours des variables entières.

On souhaite réaliser un programme qui gère les fonctionnalités suivantes :

- affichage des boissons,
- saisie du choix d'une boisson avec vérification de ce choix,
- paiement (saisie des pièces),
- rendu de la monnaie,
- fourniture éventuelle de la boisson et mise à jour des stocks.

Pour cela, on utilisera les déclarations suivantes :

```
const NombreDeBoissons = 10;
        {indique le nombre maximum de variétés de boissons qu'un distributeur peut offrir}

type UneBoisson = record {décrit une variété de boisson du distributeur}
                    Nom : string[12];    {indique le nom de la boisson}
                    Prix : integer;      {indique le prix en centimes de la boisson}
                    Quantite : integer
                    {indique le nombre de doses encore disponibles pour cette boisson}
                    end;
     Stock = array[1..NombreDeBoissons] of UneBoisson;
             {décrit le stock de boissons d'un distributeur, un élément par variété de boisson}
     Monnaie = Array[1..6] of integer;
               {décrit un ensemble de pièces de monnaie : dans un tableau de type Monnaie ...
                ... le premier élément indique le nombre de pièces de dix centimes,
                  le deuxième élément indique le nombre de pièces de vingt centimes,
                  le troisième élément indique le nombre de pièces de cinquante centimes,
                  le quatrième élément indique le nombre de pièces de un franc,
                  le cinquième élément indique le nombre de pièces de deux francs,
                  le sixième élément indique le nombre de pièces de cinq francs.}
```

Le programme principal utilisera les deux variables suivantes :

```
var Boissons : Stock;
               {tableau qui décrit les informations sur les boissons disponibles dans le distributeur}
    Caisse   : Monnaie;
               {tableau qui décrit la somme disponible dans le distributeur et sa répartition en pièces}
```

Dans tout ce qui suit, on adoptera les conventions suivantes :

- Un numéro de boisson désignera une boisson en indiquant le numéro de l'élément qui la représente dans un tableau de type `Stock`. Ce numéro sera donc un entier de l'intervalle `1..NombreDeBoissons`.
- Un numéro de pièce est un entier de 1 à 6 qui désigne un type de pièce selon la convention suivante : entier 1 : pièce de dix centimes, entier 2 : pièce de vingt centimes, entier 3 : pièce de cinquante centimes, entier 4 : pièce de un franc, entier 5 : pièce de deux francs, entier 6 : pièce de cinq francs.

Tournez la page S.V.P.

1 Quel est l'intérêt d'utiliser la constante `NombreDeBoissons` ?

2 a) A partir de la variable *Boissons*, comment sont notées les variables représentant le prix de la troisième variété de boissons et la quantité de la cinquième ?

b) Si les éléments du tableau *Caisse* contiennent respectivement les valeurs 10, 20, 12, 4, 7 et 8, quelle est la somme totale disponible dans la machine ?

3 a) Ecrivez une fonction **EnStock** qui prend comme paramètres un tableau de boissons *Tab* du type `Stock`, et un numéro de boisson *Num*. Cette fonction rendra une valeur booléenne égale à « vrai » si et seulement si la boisson *Num* est disponible.

b) Ecrivez une fonction **Valeur** qui prend en paramètre un entier qu'on appellera *Piece* représentant un numéro de pièce et qui rend un entier qui est la valeur en centimes de la pièce désignée. On se reportera au préambule qui précède la question 1 pour la convention sur les numéros de pièce, et on supposera qu'il ne peut pas y avoir d'erreur sur la valeur du paramètre *Piece* qui sera toujours un entier compris entre 1 et 6.

4 **Affichage des boissons**

Ecrivez une procédure **Affichage** qui à partir d'un tableau de boissons *Tab* affiche à l'écran l'ensemble des boissons encore disponibles sous la forme que l'exemple suivant illustre :

```
boisson 1  café          prix : 250 centimes
boisson 2  café au lait  prix : 300 centimes
boisson 5  thé           prix : 300 centimes
boisson 8  orangeade     prix : 220 centimes
```

(Les correcteurs ne pénaliseront pas les mauvais alignements de colonnes)

5 **Paiement (saisie des pièces)**

a) On veut écrire la procédure **EntrePiece** qui gère l'entrée d'une pièce dans le distributeur ou l'annulation d'une commande. On supposera pour cela que le processus qui contrôle le distributeur transmet, en réaction à un tel événement, un entier que le programme lira avec un **readln**. L'entier transmis sera :

- le nombre 0, si l'utilisateur a appuyé sur le bouton d'annulation de la commande
- un entier de 1 à 6 correspondant au numéro de pièce, si l'utilisateur a entré une pièce valide
- un entier extérieur à l'intervalle [0, 6] si l'utilisateur a entré une pièce invalide.

Ecrivez la procédure **EntrePiece** qui a comme paramètres l'entier représentant l'événement et la caisse du distributeur. Cette procédure permet l'initialisation du paramètre représentant l'événement et met à jour la caisse en fonction de cet événement.

b) Ecrivez une procédure **Paiement** qui a comme paramètres le prix d'une boisson, la somme versée et la caisse du distributeur.

Cette procédure saisira les pièces données par l'utilisateur une par une, les rajoutera une par une en caisse et ceci jusqu'à ce que la boisson soit payée ou la commande annulée.

En entrée, les paramètres décrivant le prix d'une boisson et la caisse seront supposés initialisés. Le paramètre décrivant la somme versée par l'utilisateur devra être initialisé à zéro au début de la procédure.

Quand une des deux conditions de sortie sera remplie, la procédure se finira en ayant bien soin de rendre la caisse mise à jour, et d'avoir mis à jour la somme versée en fonction de l'entrée de la monnaie. Dans tous les cas, cette procédure ne s'occupera pas de rendre de la monnaie.

A SUIVRE

6 Rendu de la monnaie

a) Ayant une somme à rendre inférieure à 5 francs, à prendre dans une caisse de type `Monnaie`, et supposant que la répartition des pièces de cette caisse permet de l'obtenir, on applique l'algorithme suivant :

> Tant que l'on n'a pas fini de rendre la monnaie, on rend la pièce de valeur la plus élevée dans la caisse et qui ne dépasse pas la somme qu'il reste à rendre.

Est-on sûr d'obtenir la monnaie ? Et pourquoi ?

b) Vous supposerez que l'on est sûr que la caisse permet de rendre n'importe quelle somme, en particulier `Caisse[1]`$\neq 0$.

Ecrivez une procédure **`Monnaie`** à deux paramètres qui à partir d'une somme à rendre (notée *SommeARendre*), et de la caisse (notée *Caisse*), rend la monnaie. Au sortir de la procédure, la caisse sera mise à jour en tenant compte des pièces qui ont été enlevées. On supposera pour cela que le processus qui contrôle le distributeur réagit à un ordre d'écriture **`writeln`**, ayant un seul paramètre de type entier, en donnant l'ordre à la machine de fournir une pièce dont le numéro est indiqué par ce paramètre suivant la convention décrite au début du problème.

7 Programme principal

On suppose la caisse *Caisse* initialisée ainsi que le tableau des boissons *Boissons*. On suppose également que la caisse contient assez de monnaie pour rendre la monnaie à l'utilisateur quelle que soit la boisson qu'il choisisse et la somme qu'il introduit. On suppose enfin que l'utilisateur ne peut pas introduire de pièces tant que la boisson n'a pas été choisie.

Ecrivez le programme principal du distributeur de boissons. On déclarera les variables dont on a besoin. On pourra, si on le souhaite, écrire des fonctions ou procédures auxiliaires. Ce programme ne sera conçu que pour une utilisation. Il effectuera les opérations suivantes :

- affichage des boissons,
- saisie du choix d'une boisson avec vérification de ce choix,
- paiement (saisie des pièces),
- rendu de la monnaie,
- fourniture éventuelle de la boisson et mise à jour des stocks. On effectuera cela comme suit : si la boisson a été payée, la fournir par une écriture du nom de la boisson voulue (exemple : `writeln('café')` est la fourniture d'un café), décrémenter d'un la quantité disponible de cette boisson.

On se servira de **`Affichage`**, **`EnStock`**, **`Paiement`**, **`Monnaie`**.

8 Le programme risque de ne pas fonctionner si la caisse ne peut pas rendre la monnaie.

On remarque que si la caisse est capable de rendre n'importe quelle somme entre 10 et 490 centimes, on est sûr que l'appareil peut rendre la monnaie lors de sa prochaine utilisation. Proposez une procédure performante pour vérifier qu'une caisse de type `Monnaie` possède cette propriété.

FIN