



Université
de Limoges

FACULTÉ
DES SCIENCES
ET TECHNIQUES

Département d'Informatique

Devoir Blanc Décembre 2012

Durée : 1 heure 30 Cours, documents autorisés

NB : une rédaction soignée des algorithmes est exigée.

Question 1 (4 points)

(inutile de réécrire les sous-programmes donnés, les numéros suffiront)

1) Écrire sous forme de fonctions les en-tête des procédures suivantes, puis pour chacune écrire un exemple d'utilisation du sous-programme sous la forme procédure et sous la forme fonction. S'il n'est pas possible de transformer une (ou plusieurs) procédure(s) en fonction(s) de façon naturelle, expliquez pourquoi :

1. procédure premiere (E A: réel, E B: réel, S C: entier)
2. procédure seconde (E V: vecteur de réels, E/S N: entier)
3. procédure troisieme (S Moy: réel, E/S W: vecteur de 10 entiers)
4. procédure quatrieme (E E: entier, S C: caractère, S R : réel)
5. procédure cinquieme (E U: réel)

2) Même exercice dans le sens inverse.

6. fonction une (E VC : vecteur de 40 caractères) : entier
7. fonction deux (E/S X : entier, E B : réel):réel
8. fonction trois (S T:booléen): booléen

Question 2 (6 points)

On veut construire un « démineur linéaire », autrement dit placer sur un vecteur de **MAX** éléments initialisé à 0 un nombre **N** donné de 1, qui seront les « mines ».

On suppose connues deux procédures **InitVecteur** et **AfficherVecteur**, qui l'une initialise à 0 tous les éléments d'un vecteur, l'autre affiche les éléments du vecteur. On suppose également que on suppose que l'on dispose d'une fonction **hasard (E N: entier) : entier**, qui retourne un nombre entier aléatoire compris entre 0 et N-1.

Vous écrirez :

1. une procédure **Tirage** ayant pour premier paramètre le vecteur **V** en entrée/sortie et pour second paramètre sa taille. Cette procédure consiste à tirer au hasard un entier **x** entre 0 et taille-1 jusqu'à ce que **V[x]** soit égal à 0. On met alors **V[x]** à 1
2. une procédure **Remplissage** ayant pour paramètres le vecteur **V**, sa taille et un entier **N**, qui réalise le placement des mines

- un algorithme d'application, qui initialise le vecteur de MAX entiers à 0, demande à l'utilisateur de choisir le nombre N de mines (que l'on imposera $<MAX/2$), place les mines et affiche le vecteur résultat

Exemple (pour $MAX=10$) : si on a 4 mines à placer et qu'on en a déjà placé 3, disposées comme suit : {0, 1, 0, 0, 0, 1, 0, 0, 1, 0}, alors on réalisera le tirage jusqu'à ce que l'on obtienne un nombre différent de 1, 5 et 8

Certaines procédures dépendent les unes des autres, mais en cas de difficulté on peut se contenter d'utiliser les en-têtes (par exemple pour écrire l'algorithme)

Question 3 (10 points)

On admettra dans un premier temps que l'on dispose de deux fonctions

fonction **Place** (E V : vecteur de entier, E N : entier, E nbCherche : entier) : entier

qui retourne :

- le premier indice i ($0 \leq i < N$) tel que $V[i] = nbCherche$
- 1 si nbCherche n'a pas été trouvé parmi les N premiers éléments de V

Exemple: $V = \{2, 6, 2, 7, 1, 2, 7, 5, 2, 5, 8, 4, 6\}$

$N=10$

pour nbCherche=7, la fonction **Place** retourne l'indice 3

pour nbCherche=9 ou nbCherche=4, elle retourne -1

On veut à partir d'un vecteur V d'entiers de MAX éléments construire deux vecteurs VElem et VFreq tels que VElem contiennent tous les éléments distincts de V et VFreq leurs fréquences.

Pour tout indice i , VFreq[i] sera le nombre de fois où VElem[i] figurait dans V.

Exemple avec le même vecteur :

$V = \{2, 6, 2, 7, 1, 2, 7, 5, 2, 5, 8, 4, 6\}$

On obtiendra $VElem = \{2, 6, 7, 1, 8, 5, 4\}$

$VFreq = \{4, 2, 2, 1, 1, 2, 1\}$ (2 figure quatre fois dans V, 5, 6 et 7 y figurent deux fois, etc.)

- Écrire une procédure qui admet pour paramètre d'entrée le vecteur V et sa taille et pour paramètres de sortie les deux vecteurs VElem et VFreq, ainsi que le nombre d'éléments distincts de V (*).
- Écrire une procédure de tri croissant du vecteur VElem tout en gardant la cohérence avec le vecteur VFreq (qui doit donc lui aussi être organisé en fonction des opérations de tri) :
On obtiendrait :

$VElem = \{1, 2, 4, 5, 6, 7, 8\}$

$VFreq = \{1, 4, 1, 2, 2, 2, 1\}$

- Écrire une procédure d'affichage des deux vecteurs VElem et VFreq comme suit :

1	1
2	4
4	1

5	2
6	2
7	2
8	1

4. Écrire un algorithme d'application qui saisit un vecteur de `MAX` éléments, réalise la construction de `VElem` et `VFreq`, puis affiche les résultats comme ci-dessus
5. Écrire le corps de la fonction `Place`

(*) **Indication** : Pour chaque élément de `V`, rechercher s'il est déjà dans le vecteur `VElem`. Dans ce cas, incrémenter l'élément correspondant de `VFreq` d'une unité. Dans le cas contraire, ajouter cet élément dans `VElem` avec la fréquence 1. Vous utiliserez bien sûr la fonction **Place**.

Remarque : il est très possible de traiter les questions 2 et 3 même si vous n'avez pas su répondre à la question 1. La question 4 (presque une question de cours) est indépendante des autres.

Corrige

Question 1 :

- 1) fonction premiere (E A : réel , E B : réel) : entier
- 2) fonction seconde (E V : vecteurs de réel, E N:entier) : entier
- 3) fonction troisieme (E/S W : vecteur de 10 entier) : réel
- 4) 2 sorties donc pas souhaitable. Mais si vraiment on veut, on peut faire :
fonction quatrieme (E E:entier, S C:caractère):réel
ou
fonction quatrieme (E E:entier, S R:réel):caractère
(voilà ce qu'il faudrait qu'ils répondent)
- 5) impossible car pas de sortie

- 6) procédure une (E VC : vecteur de 40 caractère , S A : entier)
- 7) procédure deux (E/S X : entier , E B : réel , E C : réel)
- 8) procédure trois (S T:booléen, S U:booléen)

Question 2

Déjà connues les deux premières procédures et la première fonction

procédure InitVecteur(E/S V : vecteur d'entier, E taille : entier) qui initialise tous les éléments du vecteur à 0

procédure AfficherVecteur(E V : vecteur d'entier, E taille : entier) qui affiche les éléments du vecteur

fonction hasard(E N:entier):entier qui retourne un nombre aléatoire entre 0 et N-1.

procédure Tirage(E/S V :vecteur d'entier, E taille : entier)

```
{
    variable alea : entier

    Répéter
    {
        alea <-- hasard(taille)
    } jusqu'à (V[alea]=0)
    V[alea] <-- 1
}
```

procédure Remplissage(E/S V :vecteur d'entier, E taille : entier, E N :entier)

```
{
    variable i : entier

    Pour i de 1 à N
    {
        Tirage(V, taille)
    }
}
```

Algorithme Demineur

```
{
    constante entier MAX <--- 10
    variable vect : vecteur de MAX entier
    nbmines : entier

    InitVecteur(vect, MAX)

    Répéter
    {
        Afficher("Veuillez saisir le nombre de mines souhaitées (< ", MAX/2, ").\n")
        Saisir(nbmines)
    } jusqu'à (nbmines < MAX/2)

    Remplissage(vect, MAX, nbmines)
    AfficherVecteur(vect, MAX)
}
```

Question 3 :

```
fonction Place(E V : vecteur d'entier, E N: entier, E nbCherche: entier):entier
{
    variable i: entier
        position : entier

    i <-- 0
    position <-- -1
    Tant que ((i < N) ET (position = -1))
    {
        Si ((V[i]=nbCherche) alors
        {
            position <-- i
        }
        i <-- i+1
    }

    retourne position;
}

procédure CalculFrequences(E V : vecteur d'entier, E taille: entier, S VElem : vecteur d'entier, S VFreq :
vecteur d'entier, S NbEltsDistincts: entier)
{
    variable i,j : entier

    NbEltsDistincts <-- 0

    Pour i de 0 à taille - 1
    {
        j <-- Place(VElem,NbEltsDistincts, V[i])
        Si (j!=-1) alors
        {
            VFreq[j] <-- VFreq[j] + 1
        }
        sinon
        {
            (VElem[NbEltsDistincts] <-- V[i]
            VFreq[j] <-- 1
            NbEltsDistincts <-- NbEltsDistincts + 1
        }
    }
}

procédure Tri(E/S VElem : vecteur d'entier, E/S VFreq : vecteur d'entier, E taille : entier)
{
    variable i_etape, i, indice_du_max: entier
        temp : entier

    /* utilisation du tri par recherche du maximum*/

    Pour i_etape de 1 à (taille - 1) /* une étape */
    {
        /* Recherche du max */
        indice_du_max <-- 0
        Pour i de 1 à (taille - i_etape)
            Si (VElem[i] > VElem[indice_du_max]) alors
                indice_du_max <-- i

        /* permutation */
        temp <-- VElem[indice_du_max]
        VElem[indice_du_max] <-- VElem[taille - i_etape]
        VElem[taille - i_etape] <-- temp
        temp <-- VFreq[indice_du_max]
        VFreq[indice_du_max] <-- VFreq[taille - i_etape]
        VFreq[taille - i_etape] <-- temp
    }
}
```

```

procédure AfficherFrequences(E/S VElem : vecteur d'entier, E/S VFreq : vecteur d'entier, E taille : entier)
{
    variable i : entier

    Pour i de 0 à taille-1
    {
        Afficher(VElem[i], "          ", VFreq[i], "\n")
    }
}

```

Algorithme Frequences

```

{
    constante entier MAX <-- 10
    variable      V, VElem, VFreq : vecteur de MAX entier
                  i, nbElem : entier

    Pour i de 0 à MAX-1
    {
        Afficher("Veuillez saisir l'élément ", i, " : ")
        Saisir(V[i])
    }

    CalculFrequences(V, MAX, VElem, VFreq, nbElem)
    Tri(VElem, VFreq, nbElem)
    AfficherFrequences(VElem, VFreq, nbElem)
}

```