Algorithmique - TD3

IUT 1ère Année

10 octobre 2012

Les exercices marqués avec une étoile sont un peu plus difficiles. Il est recommandé de prendre le temps de travailler sur ces exercices.

1 Les boucles (suite)

Exercice 1. Ecrire un algorithme qui reçoit en entrée un nombre entier de 1 à 10 et affiche en sortie la table de multiplication de ce nombre. Par exemple, si l'algorithme reçoit le nombre 7, il affichera la table :

- $1 \times 7 = 7$
- $2 \times 7 = 14$
- ...
- $10 \times 7 = 70$

Exercice 2. A la naissance de Marie, son grand-père Nestor, lui ouvre un compte bancaire. Ensuite, à chaque anniversaire, le grand père de Marie verse sur son compte 100 €, auxquels il ajoute le double de l'âge de Marie. Par exemple, lorsqu'elle a deux ans, il lui verse 104 €. Ecrire un algorithme qui permette de determiner quelle somme aura Marie lors de son n-ième anniversaire.

Exercice 3. La population des Sims Alpha est de 10,000,000 d'habitants et elle augmente de 500,000 habitants par an. Celle des Sims Beta est de 5,000,000 habitants et elle augmente de 3% par an. Ecrire un algorithme permettant de déterminer dans combien d'années la population de Sims Beta dépassera celle des Sims Alpha.

Exercice 4. Corriger le programme C++ suivant afin de résoudre le problème suivant :

- Données : un nombre entier positif n
- Résultat : le résultat de la suite harmonique : $\sum_{i=1}^{n} \frac{1}{i}$

Listing 1 – suiteHarmonique

```
#include <iostream>
using namespace std;

int main()
{
int i,n,somme;
cout << "Entrer le nombre entier: ";
cin >> n;
for(i = 0; i < n; i++)
    somme = somme + 1/i;
cout << "Le résultat est: " << somme << endl;
return 0;
}
```

Exercice 5. Construire un algorithme permettant d'évaluer vos chances de gagner dans l'ordre ou dans le désordre au tiercé, quarté ou quinté. De manière formelle, le problème est le suivant :

- Données : un nombre p de chevaux partants et un nombre $j \in \{3,4,5\}$ de chevaux joués
- Résultat : la probabilité de gagner au jeu dans l'ordre, et la probabilité de gagner au jeu dans le désordre

Rappel : les formules habituelles de comptage sont données dans la table ci-jointe.

Nombre de possibilités de construire une liste ordonnée, avec répétitions, de j éléments parmi p	p^j
Nombre de possibilités de construire une liste ordonnée, sans répétition, de j éléments parmi p	$\frac{p!}{(p-j)!}$
Nombre de possibilités de construire un ensemble non ordonné, sans répétition, de j éléments parmi p	$\frac{p!}{(p-j)!j!}$

2 Les tableaux

Exercice 6. Corriger l'algorithme en pseudo-code suivant afin de résoudre le problème suivant :
- Données : deux vecteurs **p** et **q** dans un espace (Euclidien) à 3 dimensions
- Résultat : la somme des vecteurs **p** + **q**

Algorithme 1: SommeDeVecteurs

```
variables
    réel p[3]
    réel q[3]
    réel r[3]

début
    pour i ← 1 à 3 faire
        pour j ← 1 à 3 faire
            r[j] ← p[i] + q
        fin
    fin
fin
```

Exercice 7. Ecrire un algorithme permettant de résoudre le problème suivant :
- Données : deux vecteurs **p** et **q** dans un espace (Euclidien) à 3 dimensions
- Résultat : le produit scalaire de **p** et **q**

Exercice 8. Pour sa naissance, la grand-mère de Gabriel place une somme de 1000 € sur son compte épargne rémunéré au taux de 2.25% (chaque année le compte est augmenté de 2.25%). Développer un algorithme permettant d'afficher un tableau sur 20 ans associant à chaque anniversaire de Gabriel la somme acquise sur son compte.

Exercice 9. Felix est un fermier qui dispose d'un couple de shadoks capables de se reproduire à vitesse phénoménale. Un couple de shadocks met deux mois pour grandir ; à partir du troisième mois, le couple de shadocks engendre une paire de nouveaux shadocks (qui mettront deux mois pour grandir et donc trois mois pour engendrer une nouvelle paire, etc.). Et surtout, les shadoks ne meurent jamais !

D'après cet exercice le nombre de couples de shadoks F_n à chaque mois n obéit à la loi :
- $F_1 = 1$
- $F_2 = 1$
- $F_n = F_{n-1} + F_{n-2}$

Développer un algorithme permettant de construire le tableau des couples depuis le premier jusqu'au 20ème mois.

Exercice 10. Corriger le programme C++ suivant afin de résoudre le problème suivant :
- Données : un tableau de 100 entiers, une valeur entière x
- Résultat : le nombre d'occurrences de x dans le tableau

Listing 2 – nombreOccurrences

```
#include <iostream>
using namespace std;

int main()
{
int tableau[100];
int i,x,occurrences;

cout << "Entrer votre valeur: ";
cin >> x;
i = 0;
occurrences = 0;
while(tableau[i] != x) i++;
cout << occurrences << endl;
return 0;
}
```

Exercice 11. Nous souhaitons développer un algorithme permettant de rechercher un élément dans un tableau de 100 entiers en partant des deux extrémités. Dans cette perspective, corriger le programme C++ suivant.

Listing 3 – rechercheBipolaire

```
#include <iostream>
using namespace std;

int main()
{
int tableau[100];
int i,j,x;
bool trouve;

cout << "Entrer votre valeur: ";
cin >> x;
i = 0;
j = 100;
trouve = 0;
do
    {
    trouve = (tableau[i] == x) && (tableau[j] == x);
    i++;
    j--;
    }
while(!trouve);
cout << trouve << endl;
return 0;
}
```

Exercice 12. Ecrire un algorithme permettant de résoudre le problème suivant :
- Données : un tableau *tableau* contenant 100 entiers
- Résultat : “vrai” si les entiers sont consécutifs et “faux” sinon

Rappel : deux entiers x et y sont consécutifs si et seulement si $y = x + 1$.

Exercice 13. Ecrire un algorithme permettant de résoudre le problème suivant :
- Données : un tableau *tableau* contenant 100 entiers
- Résultat : “vrai” si le tableau est trié du plus petit au plus grand et “faux” sinon

Exercice 14. Ecrire un algorithme permettant de saisir 100 valeurs et qui les range au fur et à mesure dans un tableau.

Exercice 15. Ecrire un algorithme qui inverse l'ordre d'un tableau trié d'entiers. En d'autres termes, si le tableau est trié du plus petit au plus grand, alors l'algorithme retourne le tableau trié du plus grand au plus petit ; réciproquement, si le tableau est trié du plus grand au plus petit, alors l'algorithme retourne le tableau trié du plus petit au plus grand.

Exercice 16. Ecrire un algorithme qui calcule le plus grand écart dans un tableau d'entiers.
Rappel : l'écart entre deux entiers x et y est la valeur absolue de leur différence $|x-y|$.

Exercice 17 (*). Ecrire un algorithme de recherche dichotomique permettant de résoudre le problème suivant :
- Données : un tableau *tableau* contenant 1000 entiers (avec répétitions possibles) triés du plus petit au plus grand, ainsi qu'un entier x
- Résultat : l'index de la première occurrence de x dans le tableau s'il est présent, et -1 sinon.

Exercice 18 (*). L'algorithme suivant *prétend* trier un tableau. Pensez-vous qu'il termine ? S'il termine effectivement, pensez-vous que le tableau final est bien trié ?

Algorithme 2: triBizzaroide

```
variables
    entier tableau[100]
    entier i, t
    booléen permuté

début
    répéter
        permuté ← faux
        pour i ← 1 à 99 faire
            si tableau[i − 1] > tableau[i] alors
                t ← tableau[i − 1]
                tableau[i − 1] ← tableau[i]
                tableau[i] ← t
                permuté ← vrai
            fin
        fin
    jusqu'à permuté = faux
fin
```

Exercice 19 (*). Ecrire un algorithme permettant de résoudre le problème suivant :
- Données : un tableau f représentant une courbe : chaque couple $(x, f[x])$ du tableau correspond à l'abscisse et l'ordonnée de la courbe.
- Résultat : la surface de la courbe par la méthode des rectangles

La méthode des rectangles est illustrée dans la figure ci-dessous. Pour chaque entier impair t, les coordonnées du $t$ème rectangle sont données par $(t-1,0),(t+1,0),(t-1,f[t]),(t+1,f[t])$.