

Examen intermédiaire, Programmation I

Sciences et Technologies du Vivant, Semestre 1

Mercredi 14 novembre 2007

1. N'oubliez pas de mettre vos NOM et PRÉNOM sur toutes vos copies.
2. Merci de garder la donnée.
3. Vous pouvez rédiger vos réponses au crayon. Cependant, prenez soin d'écrire proprement. Les copies illisibles ne recevront pas de points.
4. Vous n'avez pas à écrire un programme entier avec `#include`, etc. Rédigez uniquement la partie demandée. Dans le cas d'une fonction, écrivez l'en-tête et le corps de la fonction uniquement.
5. Pour les exercices 3 et 4, le nombre de lignes de code entre crochets [] au début des questions est donné à titre indicatif. Vous n'êtes pas obligés de fournir une réponse comportant *exactement* le même nombre de lignes. Par contre, si votre réponse est *beaucoup plus longue* que celle du corrigé, il est certainement possible de faire plus simple.
6. Les réponses qui font ce qui est demandé mais qui sont trop compliquées n'obtiendront pas le maximum des points.
7. Les nombres de points pour chaque question sont donnés dans la marge de droite, et sont à titre indicatif. Le total de points est sur 100.
8. La question 2.c est une question bonus (au-delà des 100 points).

Question 1 – Commandes Unix 15 points

On suppose qu'on possède dans notre répertoire *home* un unique répertoire nommé `temp`, dans lequel sont stockées une série de photos (fichiers dont le nom se termine par `.jpg`) et de vidéos (fichiers dont le nom se termine par `.avi`). Un listing du répertoire *home* nous donne :

```
drwxr-xr-x 2 jerome users 4096 2007-11-06 13:52 temp
```

- (a) Écrivez la commande permettant d'afficher le nom et la taille des vidéos contenues dans `temp`, dont le nom commence par `a` ou par `b`. (3)
- (b) On aimerait classer correctement nos fichiers images et vidéos. Depuis le répertoire *home*, écrivez la/les commande(s) nécessaire(s) à la création de deux nouveaux sous-répertoires appelés `videos` et `images`. (3)
- (c) Écrivez maintenant la commande permettant de *déplacer* toutes les vidéos contenues dans `temp` vers le nouveau répertoire `videos`. Indiquez également la commande pour déplacer les images vers `images`. (3)
- (d) Le répertoire `temp` est désormais vide et ne sert plus à rien. Indiquez la commande permettant de le supprimer. (3)
- (e) Vous n'avez pas envie que d'autres utilisateurs que vous puissent regarder (lire) ou modifier (écrire) vos images. Ecrivez la/les commande(s) nécessaire(s) pour vous assurer que ça n'arrivera pas. (3)

Question 2 – Pointeurs 15+8 points

(a) Indiquez ce qu'affiche le code suivant :

(8)

```
int i = 10;
int j[5] = {1, 3, 5, 7, 9};
int *p = &i;
int *q = j;

cout << "1. " << *p << " " << *q << endl;

*p = *p + 2;
q = q + 2;

cout << "2. " << *p << " " << *q << endl;

*q = *j;
*p = 2;

cout << "3. " << i << " " << *q << endl;

q[i] = 8;

cout << "4. ";
for (int k=0; k<5; k++)
    cout << j[k] << " ";
cout << endl;
```

(b) Indiquez ce qu'affiche le code suivant :

(7)

```
int t[5] = {2, 4, 6, 8, 10};

cout << "1. " << *(t + 1) << " " << t[2] << endl;

int *p = &t[3];

p--;

cout << "2. " << *p << endl;

(*p)--;

cout << "3. " << *p << endl;

cout << "4. " << int(p) - int(t) << endl;
```

(c) Indiquez ce qu'affiche le code suivant :

(8)

```
int i = 5;
int *p = &i;

cout << "1. " << *p << endl;

p = p + 1;

cout << "2. " << i << endl;

int **q = &p;
```

```

p = &i;

**q = 12;

int j = 10 + i;

cout << "3. " << j << endl;

i = int(p);

int *r = (int *) (**q);
*q = &j;

i = i - int(r) + 35;

cout << "4. " << *r - *p << endl;

```

Question 3 – Implantation de racines 40 points

Le but de cet exercice est de comprendre, à travers les fonctions racine carrée et sinus, comment les fonctions mathématiques peuvent être calculées par un ordinateur.

- (a) [4 lignes] La Méthode Babylonienne utilise une suite qui converge en quelques itérations vers la racine carrée du nombre donné. Les termes de cette suite sont définis par : (5)

$$U_{i+1} = \frac{1}{2} \left(U_i + \frac{x}{U_i} \right)$$

où x représente le nombre dont on veut calculer la racine. Pour pouvoir calculer cette suite, il faut encore définir son premier terme U_0 . Plus U_0 est proche de la racine carrée de x , plus la suite converge rapidement. Pour commencer, nous prendrons simplement :

$$U_0 = x$$

mais d'autres solutions sont possibles.

Écrivez la fonction d'en-tête

```
float mon_sqrt(float x, int n)
```

qui retourne la valeur de \sqrt{x} approximée par le $n^{\text{ème}}$ terme de la suite U_i . On supposera que x est strictement positif.

- (b) [4 lignes] Pour accélérer le calcul de la racine carrée, nous allons utiliser un tableau qui contiendra les valeurs précalculées des racines carrées des valeurs entières de 0 à $m-1$, où m sera défini comme paramètre. (5)

Écrivez la fonction d'en-tête

```
float * cree_tableau_pour_sqrt(int m)
```

qui alloue un tableau de m floats, le remplit avec les racines carrées des nombres entiers de 0 à $m-1$ en utilisant la fonction `mon_sqrt`, et retourne le pointeur sur ce tableau.

- (c) [3 lignes] Écrivez maintenant la fonction : (7)

```
float mon_sqrt_avec_tableau_precalcule(float x, float * tab, int m)
```

où `tab` est le tableau calculé avec la fonction précédente, et m la taille de `tab`. Cette fonction calculera la racine carrée de x en initialisant la suite U_i avec la racine carrée de la partie entière de x , racine qu'on obtiendra grâce au tableau `tab`. Quand la suite est initialisée aussi près de la valeur correcte, une seule itération suffit.

Comment gérer correctement le cas où x est supérieur ou égal à m ?

- (d) [5 lignes] Considérons maintenant la fonction sinus. Une façon de la calculer est d'utiliser son développement de Taylor : (7)

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} + \dots$$

Écrivez la fonction d'entête :

```
float mon_sinus(float x, int n)
```

qui retournera la valeur de $\sin(x)$ en considérant les n premiers termes de son développement de Taylor. Évitez de calculer pour chacun des termes la puissance et la factorielle en remarquant qu'un terme peut être calculé facilement à partir du précédent.

- (e) [2 lignes] En pratique, la fonction `mon_sinus` ne fonctionne pas bien pour de grandes valeurs de x : Bien que le développement de Taylor de $\sin(x)$ est mathématiquement valide quelle que soit la valeur de x , cette fonction risque de renvoyer des valeurs erronées pour de grandes valeurs de x . En effet, les termes de haut rang risquent alors de prendre des valeurs incorrectes à cause de la précision limitée des types `float` et `double`. De plus, même quand le résultat est correct, plus la valeur de x est grande, plus il faut faire d'itérations (c'est-à-dire prendre une valeur grande pour n) pour obtenir une valeur correcte, ce qui prend du temps de calcul. (8)

Une solution est d'utiliser un changement de variable en posant :

$$x = m\frac{\pi}{2} + y$$

où m est une valeur entière et y une valeur réelle comprise entre 0 et $\frac{\pi}{2}$, et d'utiliser les relations suivantes :

- Si m est divisible par 4, $\sin(x) = \sin(y)$;
- Si le reste de la division de m par 4 vaut 1, $\sin(x) = \cos(y)$;
- Si le reste de la division de m par 4 vaut 2, $\sin(x) = -\sin(y)$;
- Si le reste de la division de m par 4 vaut 3, $\sin(x) = -\cos(y)$;

POUR SIMPLIFIER, NOUS NE CONSIDERERONS QUE LES CAS OÙ m EST POSITIF.

Commencez par écrire la fonction `trouve_m_et_y` qui permet de retrouver m et y pour x donné. On pourra utiliser la constante `M_PI` qui contient la valeur de π .

- (f) [17 lignes] Finalement, écrivez la fonction d'en-tête : (8)

```
float mon_sinus2(float x)
```

qui retrouve d'abord les valeurs de m et y en utilisant la fonction `trouve_m_et_y`, puis retourne la valeur de $\sin(x)$ calculée grâce aux relations données ci-dessus.

Pour calculer $\sin(y)$ on utilisera la fonction `mon_sinus` pour calculer la somme des 10 premiers termes du développement de Taylor. On supposera qu'on aura également défini la fonction `mon_cosinus` équivalente pour pouvoir calculer $\cos(y)$.

Question 4 – Tableaux30 points

- (a) [5 lignes] Écrivez une fonction `copie_tableau` qui duplique un tableau d'entiers. La fonction prendra pour arguments un tableau d'`int` ainsi que la taille du tableau. Elle devra allouer de la mémoire pour le nouveau tableau et y copier les valeurs de l'ancien. La fonction retournera un pointeur sur le nouveau tableau. (5)
- (b) [7 lignes] Écrivez une fonction `melange_tableau` qui mélange aléatoirement les éléments d'un tableau en les permutant 2 à 2. Par exemple, en permutant le 2^e et le 5^e élément du tableau suivant (10)

3	8	4	0	1	2	7
---	---	---	---	---	---	---

on obtient

3	1	4	0	8	2	7
---	---	---	---	---	---	---

Votre fonction devra prendre en paramètres un tableau d'`int`, sa taille, ainsi que le nombre de permutations à effectuer pour mélanger le tableau. Pour chaque permutation, il faudra sélectionner aléatoirement deux éléments du tableau et les échanger.

- (c) [21 lignes] Veuillez écrire une fonction **alphabetique** qui vérifie si deux chaînes de caractères sont placées dans l'ordre alphabétique. La fonction prendra en paramètres deux chaînes de caractères `c1` et `c2` et retournera `true` si `c1` est avant `c2` dans l'ordre alphabétique et `false` sinon. Pour cet exercice, vous pouvez supposer que les chaînes sont toujours en minuscules et qu'elles ne contiennent pas de caractères spéciaux ni de chiffres. Vous n'êtes pas autorisés à utiliser la fonction `strcmp`, qui présente un comportement très similaire. (15)

On rappelle que si deux mots commencent par la même lettre, il faut regarder le deuxième caractère pour déterminer l'ordre alphabétique, et ainsi de suite. Si deux mots possèdent les mêmes premiers caractères, c'est le plus court qui passe le premier dans l'ordre alphabétique. Par exemple : *table* vient avant *tableau*.

Indice : la fonction `strlen` permet d'obtenir la taille d'une chaîne de caractères.