UNIVERSITÉ PAUL CÉZANNE
BIBLIOTHÈQUE UNIVERSITAIRE

FACULTÉ DES SCIENCES ET TECHNIQUES DE SAINT JÉRÔME

SUJETS D'EXAMEN LMD

Cycle **LICENCE**

Série **I1 - Algorithmique et Programmation**

Années **2005-**

Attention : un sujet peut comporter plusieurs pages

© 2005 - Université Paul Cézanne Aix-Marseille 3 - FST - BU
Toute reproduction, totale ou partielle, de ce document, est interdite sans le consentement express de son auteur

Université Paul Cézanne
Licence Mathématique et Informatique (M.I.)

Examen de I1
Session de janvier 2007

Date : le lundi 8 janvier
Durée : 1heure 30

EPREUVE SANS DOCUMENTS
SANS CALCULATRICES

1. Qu'est-ce qu'une "**variable**" ? Soient x et y deux variables de type "entier", que réalise l'instruction suivante :

   x = y;

   (que se passe-t-il exactement en mémoire d'ordinateur ?).

2. Soient x et y deux variables de type "int", Les instructions suivantes sont-elles légales (possibles) et pourquoi?:
   - `x = -x;`
   - `2x = y;`
   - `y = 2*x;`
   - `2 = 2-x;`
   - `y-2 = x-1;`
   - `x = 2;`

3. Qu'appelle-t-on type composé par opposition à la notion de type simple? Donnez 2 exemples de variables de type composé.

4. Sachant que les variables a, b et c ont respectivement pour valeur 2, 8 et 6, déterminer si les tests suivants sont vrai ou faux :
   - `a == b`
   - `!(b != 0)`
   - `(b - a) == c`
   - `((2*a + c) > = b) && (a < = (c - b))`
   - `((c - a) < b ) || (( a + c) > b)`
   - `!(!(b < c) && ((a < c) || (a > b)))`

5. Traduire en langage C les déclarations des variables suivantes:
   - x : entier long
   - mot : "chaîne" de 30 caractères,
   - liste : tableau de 15 entiers courts,
   - matrix :tableau de 10 tableaux de 20 réels,

6. La fonction C suivante est censée afficher la somme s des carrés de tous les diviseurs d'un entier n (sauf lui-même). Pourtant, l'exécution montre des résultats erronés. Trouvez et corrigez les 4 erreurs de logique qui se sont glissées dans cet algorithme:

```
int somdiv2(int n)
{
   int i, s,
   s = 0;
   for (i=2; i<=n  ; i=i-1 )
     if ((n%i)!=  0) s=s+i*i;
   return s;
}
```

7. Que réalise la fonction C suivante :

```
void faitquoi(int k);
{   int i, j;
    for (j = 2; j <=k ; j = j+1)
      {i=2;
       while ((i<j) &&(j%i!=0))
            i = i+1;
       if (i == j)printf ("%d \n",j);
      }
}
```

8. Soit les instructions suivantes :

```
#define N 50
typedef int VECTEUR [N];
```

8.a. Ecrire la fonction C permettant de tester si son paramètre de type VECTEUR est trié ou pas. La fonction retournera 1 si le vecteur est trié et 0 sinon.

8.b. Soit un VECTEUR v trié par ordre croissant (et ne possédant pas de doublons), écrire la fonction C de recherche dichotomique d'un entier m (fourni en paramètre) dans ce VECTEUR v (en paramètre également). La fonction retournera:

- Soit l' indice de position de m dans le vecteur, si m $\in$ au vecteur
- Soit -1 si m $\notin$ au vecteur

8.c. Expliquer le principe général du tri par sélection appliqué sur un VECTEUR v.

Université d'Aix-Marseille III
Faculté des Sciences et Techniques de Saint-Jérôme
I1 - Informatique - Janvier 2005

Problème 1 – Notions de base - 5 points

Question 1. Expliquez en quelques phrases ce que signifient les termes « *déclaration de type*», « *déclaration de fonction*» et « *passage de paramètres*». Donnez pour chaque cas un exemple.

Question 2. Donnez les valeurs des variables à l'issue de l'exécution du programme C :

```
main()
{    int a,b,c,x,y,z;
     x = 3 ; y = 2; z = 5; a = 1 ; b = 2; c = a != b;
     a = !c ; b = a && c; c = a || b; x = x-z+y ; y = y+z; z = x-x;
}
```

Question 3. Donnez la « trace d'exécution » du programme C :

```
main()
{    int x,y,z;
     x = 1 ; y = 5; z = 2;
     while ( x < y ) { z = 2*z; x = x+1; }
}
```

Question 4. Que peut-on dire des valeurs des variables x, y et z en fin l'exécution du programme C :

```
main()
{    int x,y,z;
     x = 1 ; y = 50; z = 0;
     while ( x != y ) { z = z+x; x = x+2; }
}
```

Problème 2 – Boucles - 3 points

Question 1. Ecrivez un programme C qui réalise la saisie d'un entier n puis affiche les n premières valeurs de la suite u_n définie par $u_0 = 1$ et $u_n = u_{n-1}.u_{n-1} + 1$ pour $n > 0$. Vous utiliserez obligatoirement une itération **`for`**. Par contre, l'emploi d'un tableau n'est pas autorisé.

Question 2. Ecrivez un programme C qui réalise d'abord la saisie de trois valeurs entières x, y et z. Le programme devra ensuite réaliser la saisie d'entiers et s'arrêtera en affichant le message « FIN DE SAISIE » dès que l'entier qui vient d'être saisi aura une valeur égale à la différence (en valeur absolue) des valeurs de `(x+y)` et z. Il faudra aussi afficher le nombre d'entiers saisis (hormis x, y et z). Vous utiliserez une itération **`do while`**.

Problème 3 – Tableaux et fonctions - 12 points

Question 1. On considère maintenant des tableaux d'entiers sur l'intervalle [0,999] définis dans le type C suivant :

```
typedef  int  tableau[1000] ;
```

Ecrivez un programme C qui, étant donné un tableau `t` de type `tableau`, affecte les éléments du tableau `t` de telle sorte que pour tout i, $0 \leq i \leq 999$, on ait `t[i] = 2*i + 3`.

Question 2. Ecrivez un programme C qui, étant donnés un tableau `t` de type `tableau` initialisé comme à la question 1, réalise la saisie d'une valeur `x`, et vérifie s'il existe un indice `i` du tableau `t` tel que `t[i] = x`. Dans ce cas, le programme affichera `VRAI` et `FAUX` sinon.

Question 3. Définissez une fonction C, appelée `F3`, correspondant au traitement réalisé par le programme de la question 2. Toutefois, plutôt que de procéder à un affichage, il faudra que la fonction ait pour résultat l'indice de la case du tableau où la valeur `x` est mémorisée quand `VRAI` était affiché, et -1 quand `FAUX` était affiché.

Question 4. On dit qu'un tableau `t` est trié dans l'ordre croissant si pour tout indice `k`, $0 \leq k < 999$, on a bien `t[k] ≤ t[k+1]`. On dit qu'une section du tableau `t` allant des indices `i` à `j` est triée dans l'ordre croissant si pour tout indice `k`, avec $i \leq k \leq j$, on a bien `t[k] ≤ t[k+1]`. Ecrivez un programme C qui, étant donnés un tableau `t` de type `tableau`, et deux variables entières `i` et `j` dont on suppose que l'on a : $0 \leq i \leq j \leq 999$, affiche `VRAI` si la section de tableau `t` allant des indices `i` à `j` est triée dans l'ordre croissant et `FAUX` sinon. Le cas pour lequel `i = j` correspond au cas où la section du tableau `t` est réduite à une seule du tableau : `t[i] = t[j]`.

Question 5. Définissez une fonction C, appelée `F5`, correspondant au traitement réalisé par le programme de la question 4. Toutefois, plutôt que de procéder à un affichage, il faudra que la fonction ait pour résultat 1 quand `VRAI` était affiché, et 0 quand `FAUX` était affiché.

Question 6. Définissez une fonction C, qui prend en entrée un tableaux `t` de type `tableau`, et renvoie 1 si le tableau `t` est trié dans l'ordre croissant, et renvoie 0 sinon. Vous utiliserez la fonction `F5`. Indiquez les avantages et inconvénients qu'il peut y avoir à utiliser ici la fonction `F5`.

Question 7. Pour le cas où un tableau `t` n'est pas trié, on dit que le tableau `t` est une concaténation de sections triées s'il existe une suite d'indices de la forme $(i_1, i_2, \ldots, i_{k-1}, i_k)$, où $k > 1$ tels que :

- $i_1 = 0 < i_2 < i_3, \ldots < i_{k-1} < i_k \leq 999$ et
- pour tout j tel que $1 \leq j \leq k-1$, la section du tableau `t` allant des indices i_j à $i_{j+1}-1$ est triée dans l'ordre croissant. De plus, la section allant de l'indice i_k à 999 est triée.

Définissez une fonction C, qui prend en entrée un tableau `t` de type `tableau`, et renvoie `k` si le tableau `t` est une concaténation de `k` sections triées.

Université d'Aix-Marseille III
Faculté des Sciences et Techniques de Saint-Jérôme

I1 - Informatique - Juin 2005

Problème 1 - Notions de base

Question 1. Expliquez en quelques phrases ce que signifie le terme « *affectation de variable* ». Vous donnerez pour cela un exemple et vous décrirez les mécanismes qui sont mis en œuvre au niveau de la mémoire.

Question 2. Donnez les valeurs des variables x, y et z à l'issue de l'exécution du programme C :

```
main()
{    int x,y,z;
     x = 5 ; y = 7; z = 9;
     x = x+y+z; y = x-y-z; z = x-y-z;
}
```

Question 3. Donnez les valeurs des variables x, y et z à l'issue de l'exécution du programme C :

```
main()
{    int x,y,z;
     x = 2 ; y = 4; z = ( x != y );
     x = !z ; y = ( x && z ); z = (x || y);
}
```

Question 4. Donnez les valeurs des variables x, y et z à l'issue de l'exécution du programme C :

```
main()
{    int x,y,z;
     x = 4 ; y = 5;
     if (5*x != 4*y) { y = 3; z = 5; } else x = y*y ;
}
```

Question 5. Donnez la « trace d'exécution » du programme C :

```
main()
{    int x,y,z;
     x = 1 ; y = 5; z = 2;
     while ( x <= y ) { z = z*z; x = x+2; }
}
```

Question 6. Donnez les valeurs des variables x, y et z à l'issue de l'exécution du programme C :

```
main()
{    int x,y,z;
     x = 1 ; y = 100; z = 0;
     while ( x < y) { z = z+2; x = x+1; }
}
```

Problème 2 – Boucles, tableaux et fonctions.

Question 1. Ecrivez un programme C qui réalise la saisie d'un entier k puis affiche les k premières valeurs de la suite u_n définie par $u_0 = 1$ et $u_n = u_{n-1}. u_{n-1} + 2. u_{n-1} + 3$ pour $n > 0$. Vous utiliserez obligatoirement une itération **`for`**. L'emploi d'un tableau n'est pas autorisé.

Question 2. Ecrivez un programme C qui réalise d'abord la saisie de deux valeurs entières `x` et `y`, en vérifiant que `x` est strictement inférieur à `y`. Le programme devra ensuite réaliser la saisie d'entiers et s'arrêtera en affichant le message « FIN DE SAISIE » dès que l'entier qui vient d'être saisi aura une valeur située dans l'intervalle `[x,y]`. Il faudra aussi afficher la somme des entiers saisis (hormis `x`, et `y`). Vous utiliserez une itération **`do while`**.

Question 3. On considère maintenant des tableaux d'entiers sur l'intervalle [0,999] définis dans le type C suivant :

```
typedef  int  tableau[1000] ;
```

Ecrivez un programme C qui, étant donné un tableau `t` de type `tableau`, affecte les éléments du tableau `t` en saisissant leur valeur au clavier.

Question 4. Ecrivez un programme C qui, étant donnés deux tableaux `t1` et `t2`, de type `tableau`, et une variable entière `i` telle que $0 \leq i \leq 999$, vérifie si `t1[i] = 2*t2[i]` et affiche `VRAI` si c'est le cas, et `FAUX` sinon.

Question 5. Ecrivez un programme C qui, étant donnés deux tableaux `t1` et `t2` de type `tableau`, affiche `VRAI` si `t1[i] = 2*t2[i]` pour tout i, $0 \leq i \leq 999$ et `FAUX` sinon.

Question 6. Définissez une fonction C, qui prend en entrée un tableau `t` de type `tableau`, et renvoie 1 si la plus petite valeur mémorisée dans les cases du tableau `t` est située en `t[0]` et la plus grande en `t[999]`. Dans le cas contraire (une des deux conditions au moins n'est pas vérifiée), la fonction C renverra 0.

Question 7. Définissez une fonction C, appelée `F7`, correspondant au traitement réalisé par le programme de la question 4. Toutefois, plutôt que de procéder à un affichage, il faudra que la fonction ait pour résultat 1 quand `VRAI` était affiché, et 0 quand `FAUX` était affiché.

Question 8. Définissez une fonction C, appelée `F8`, correspondant au traitement réalisé par le programme de la question 5. Toutefois, plutôt que de procéder à un affichage, il faudra que la fonction ait pour résultat 1 quand `VRAI` était affiché, et 0 quand `FAUX` était affiché. Vous utiliserez la fonction `F7`. Indiquez les avantages et inconvénients qu'il peut y avoir à utiliser ici la fonction `F7`.