Quatrième partie

Examens de l'année précédente

ULP DEUG STU/SM 2^{ieme} année : Automne 2003
Programmation Scientifique Resp. : Christian Boily

Examen d'évaluation final
Durée : 2 heures

Appel le jeudi 29 janvier 2004 à 8 :30h
Amphithéatre 6, Bâtiment LeBel Campus de l'Esplanade

Les notes personnelles, soit les notes de cours, TD et TP, sont permises. Aucun livre de référence ou support électronique (téléphone cellulaire, calculette, etc) n'est authorisé, à l'exception des dictionnaires de langues qui devront être déclarés au début de l'épreuve.

Logique (maximum 10 pts)

Une variable X doit être modifiée pour prendre l'une de quatre valeurs a, b, c ou d connues, selon certaines règles. Ces règles sont représentées dans le tableau suivant pour deux cas différents, avec leur transcription en C.

1. Expliquez pourquoi la transcription du cas #1 est correcte, mais pas celle du cas #2.
2. Pour le cas #2 : donnez un exemple explicite de ce qui ne va pas, puis corrigez la transcription C pour faire ce que demande la règle de modification de x.

Cas	Règle	Transcription en C

1	$x = \begin{cases} a \text{ si i} > 0 \\ b \text{ si i = 0} \\ c \text{ si i = -1 ou -2} \\ d \text{ sinon} \end{cases}$	x = d ;
		if (i > 0) x = a ;
		if (i==0) x = b ;
		if ((i==-1) || (i==-2)) x = c ;

2	$x = \begin{cases} a \text{ si x} > 0 \\ b \text{ si x = 0} \\ c \text{ si x = -a ou -b} \\ d \text{ sinon} \end{cases}$	x = d ;
		if(x > 0) x = a ;
		if (x==0) x = b ;
		if((x==-a) || (x==-b)) x = c ;

Déclarations (5pts)

Dites si les déclarations suivantes sont valides ou non. Corrigez les erreurs au besoin.

a) float R1, u1, u2 ; b) dfloat x,y,z ; c) int If, Then, else ;

d) char r2-d2 ; e) double D1 ; D2 ; d3 ; f) int j = 2, I = j ;

g) unsigned char C = 'c' ;

Fractal : second regard (minimum 12 pts)

Déterminez si un point c du plan complexe, $c = x + iy$, appartient à un ensemble, défini par la propriété suivante : Soit la relation de récurrence

$$z_{n+1} = z_n^2 + c \; ; \text{ ou } n = 0, 1, 2...$$

entre les nombres complexes définissant la suite $\{z_n\} = \{z_0, z_1, ...z_n\}$. Le premier élément de la suite $z_0 = 0$ dans tous les cas. On dira que le point c appartient à l'ensemble M_m si après m itérations $z_m \cdot z_m^\dagger < 4$.

Rappel : La norme d'un nombre est définie par le produit $z \cdot z^\dagger$; la notation du complexe conjugé ici est $c^\dagger = x - iy$.

Rappel 2 : la multiplication entre deux nombres complexes $a = x + iy$ et $b = X + iY$ donne $a \cdot b = (xX - yY) + i(xY + yX)$, où $i^2 = -1$ est le nombre imaginaire au carré.

(a) Écrivez la déclaration d'un type composé `complexe` , ayant un champ x et un deuxième champ y, acceptant des réels, ainsi qu' un 3ième champ, de type entier, qu'on nommera *bool*.
(b) Composez une fonction `mulcomp`, acceptant **deux** arguments, a et b, et permettant de récupérer le résultat du produit $a \cdot b$ tel que défini dans le préambule.

Nous supposerons l'existence, dans votre répertoire de travail Unix, d'un fichier nommé `entree.d` *où sont stockées les valeurs de* `N` *nombres complexes* c*. Nous voulons vérifier si chaque nombre appartient à* M_m*, et écrire le résultat dans un fichier nommé* `sortie.d`.

(c) Utilisez une instruction au pré-processeur pour définir une constante symbolique `m` prenant la valeur 100, et une 2ième constante symbolique `N` prenant la valeur 10.
(d) Définissez la fonction `main`. On précisera si oui (`bool = 1`) ou non (`bool = 0 `) le point est inclu dans M_m.
(e) Sauvergardez tous les champs de chaque nombre complexe dans *sortie.d*. Fermez les fichiers encore ouverts avant de quitter le programme.

Boucles

a) Remplacer dans la fonction suivante les boucles *for* par des boucles *do ... while*.

```
int strifor ( char s[], char t[] ) {
int i,j ;
```

```
    for( j=0 ; s[0] != '\0' ; j++ ){
for( i=0 ; ( s[i] == t[i] ) && ( s[i] != '\0' ) ; i++) ;
if( t[i] == '\0' ) return j ;
j++, s++ ;
}
    return(-1) ; }
```

b) Remplacer les tableaux *s* et *t* de la fonction strifor par des pointeurs. Vous récrirez la fonction pour plus de clareté.

ULP DEUG STU/SM 2^{ieme} année : Automne 2004
Programmation Scientifique Resp. : Christian Boily

Examen de reprise
Durée : 2 heures

Appel le vendredi 3 septembre 2004 à 13 :00h
Amphithéatre 3, Bâtiment LeBel Campus de l'Esplanade

Les notes personnelles, soit les notes de cours, TD et TP, sont permises. Aucun livre de référence ou support électronique (téléphone cellulaire, calculette, etc) n'est authorisé, à l'exception des dictionnaires de langues qui devront être déclarés au début de l'épreuve.

Fractal : encore et toujours .. (minimum 10 pts)

Déterminez si un point c du plan complexe, $c = x + iy$, appartient à un ensemble, défini par la propriété suivante : Soit la relation de récurrence

$$z_{n+1} = z_n^2 + c \; ; \text{ où } n = 0, 1, 2...$$

entre les nombres complexes définissant la suite $\{z_n\} = \{z_0, z_1, ...z_n\}$. Le premier élément de la suite $z_0 = 0$ dans tous les cas. On dira que le point c appartient à l'ensemble M_m si après m itérations $z_m \cdot z_m^\dagger < 4$.

Rappel : La norme d'un nombre est définie par le produit $z \cdot z^\dagger$; la notation du complexe conjugé ici est $c^\dagger = x - iy$.

Rappel 2 : la multiplication entre deux nombres complexes $a = x + iy$ et $b = X + iY$ donne $a \cdot b = (xX - yY) + i(xY + yX)$, où $i^2 = -1$ est le nombre imaginaire au carré.

(a) Écrivez la déclaration d'un type composé `complexe` , ayant un champ x et un deuxième champ y, acceptant des réels, ainsi qu' un 3ième champ, de type entier, qu'on nommera $bool$.
(b) Composez une fonction `mulcomp`, acceptant **deux** arguments, a et b, et permettant de récupérer le résultat du produit $a \cdot b$ tel que défini dans le préambule.

Nous supposerons l'existence, dans votre répertoire de travail Unix, d'un fichier nommé `entree.d` *où sont stockées les valeurs de* `N` *nombres complexes* c*. Nous voulons vérifier si chaque nombre appartient à* M_m*, et écrire le résultat dans un fichier nommé* `sortie.d`.

(c) Utilisez une instruction au pré-processeur pour définir une constante symbolique `m` prenant la valeur 100 ; une 2ième constante `N` prenant la valeur 10.
(d) Définissez la fonction `main`. On précisera si oui (`bool = 1`) ou non (`bool = 0 `) le point est inclu dans M_m.
(e) Sauvergardez tous les champs de chaque nombre complexe dans *sortie.d*. Fermez les fichiers encore ouverts avant de quitter le programme.

Itérations 3n + 1 (5pts)

Vous écrirez un programme C réalisant l'algorithme suivant. Note : il est inutile de faire un traitement superflu des entrées/sorties.

1. saisir un entier n
2. imprimer à l'écran la valeur de n
3. si n est impair, remplacer n par $3n + 1$
4. si n est pair, remplacer n par $n/2$.
5. si $n \neq 1$, repremdre à partire de la deuxième instruction.

L'algorithme est-il toujours finie et donc exécutable sur ordinateur ? Discutez, puis donnez un exemple pour $n = 22$.

Trouver la ou les erreurs (5 pts)

Le programme suivant contient une ou plusieurs erreurs.

a) Identifiez les erreurs et b) corrigez-les pour que le programme s'exécute sans faute.

Debut du programme

```
    int calcul ( char op, int x, int y)
{
switch ( op ) {
case '+' : return x+y ;
case '-' : return x-y ;
case '*' : return xy ;
case '/' : return x/y ;
}
}

    void main ( void ) {

    char op ; int a, float b, stdin = 0 ;
```

```
    printf( "Choisir operateur + / * ou - : ") ; scanf( "%c", &op ) ;

    printf ( "Deux operandes ? ") , scanf( "%d %d", &a, &b) ;
resultat = calcul( op, a,b ) printf( "%d\n", resultat ) ;

    }
```

Algorithme sympa (5 pts)

Un étudiant désire effectuer la somme de l'inverse des nombres entiers, c'est-à-dire

$$\sum_i 1/i$$

pour tous les entiers $i > i_o$. Il vous demande votre assistance pour développer un algorithme qui pourra par la suite être transcrit en C.

a) Expliquez de la manière la plus précise possible un algorithme permettant d'obtenir la somme recherchée. On pensera à inclure toutes les opérations à effectuer, y compris les entrées/sorties.

b) Réalisez cet algorithme ; un pseudo-code sera jugé acceptable, un programme complet vous permettra d'obtenir 1 point supplémentaire.