

DIALLO  
Alpha Oumar

PL1  
2013

## Devo

Le Vendredi 23 mai 2014

Durée : 2h

Sans documents, sans calculatrice

Avant de débiter le DE :

- Prenez le temps de bien lire les énoncés des exercices. Vous répondrez dans les espaces laissés libres à cet effet dans le sujet. Aucune autre copie ne sera prise en compte.
- N'oubliez pas de reporter vos nom, prénom, groupe et école dans les espaces ci-dessous
- **La qualité de la rédaction de vos réponses entrera en compte pour la notation des exercices**

NOM

DIALLO

PRENOM

Alpha Oumar

GROUPE

PL1

Promotion

L1/PL1



CPI1



## Thème 1 : Structures

a) En langage C, que signifie **typedef** ?

*0,5*  
Typedef sert à nommer une structure, elle nous permet d'appeler une structure créée par son nom, ça évite de préciser à chaque fois que la nouvelle structure créée est une structure. Par exemple :  
typedef structure coordonne coordonne { }; Permet de dire que la structure coordonne s'appelle coordonne, donc dans la fonction principale on utilisera : coordonne nom\_variable au lieu de faire structure coordonne nom\_variable.

b) Définissez une structure permettant de définir une personne dans le cadre d'une application bancaire.

*1*  
structure personne  
{  
  char nom [30];  
  char prenom [50];  
  date naissance;  
  coordonnées adresse;  
  int num\_compte;  
}  
  
typedef structure date\_date  
{  
  int jour;  
  int mois;  
  int année;  
}  
  
structure coordonne  
{  
  int num;  
  char typevoie [20];  
  char voie [50];  
  char ville [50];  
  int BP;  
}

c) Soit le programme suivant, à gauche duquel on a porté les numéros des lignes :



```

1  structure t_complexe
2  {
3      reel re, im;
4  };
5  fonction principale()
6  {
7      t_complexe z1, z2;
8
9      z1 ← 1+3i; X
10     z2.re ← 5.2;
11     t_complexe.im ← 4; X
12     z1.re ← z2.re;
13     z1.im ← 1;
14
15     z2 ← z1+z2;
16
17     afficher(z2);
18     retourner;
19 }
```

Indiquez les lignes incorrectes de ce programme en expliquant pourquoi elles sont incorrectes (réponse page suivante)

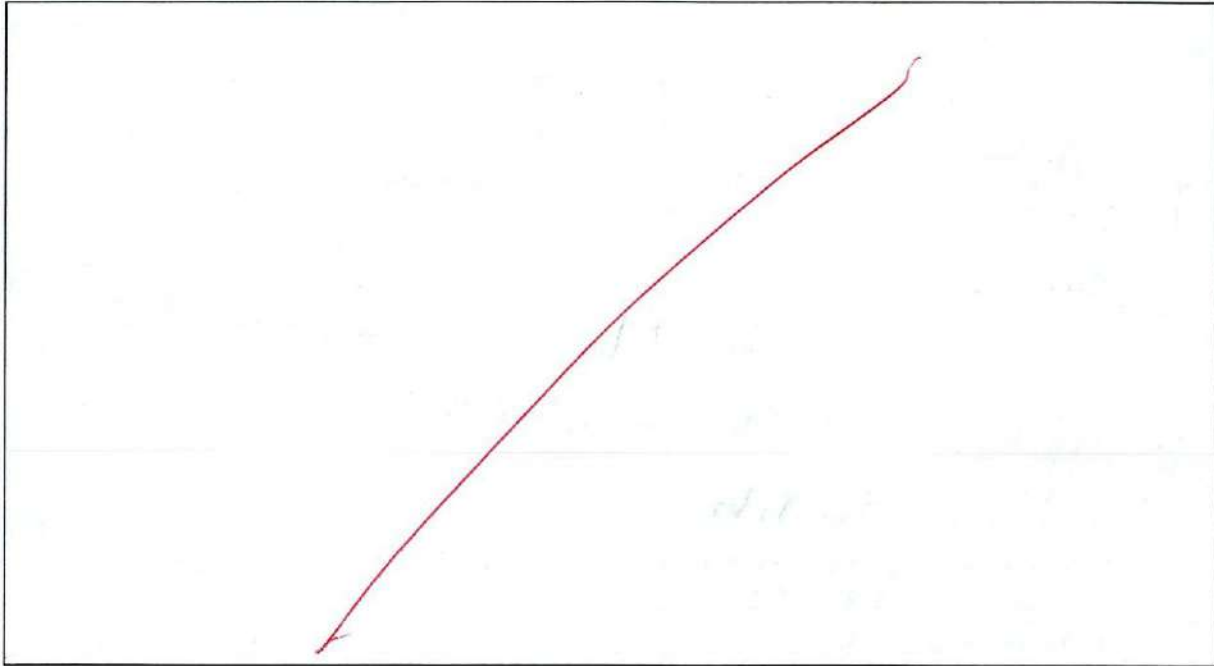
- La ligne 9 est incorrecte, l'ordinateur ne sait pas faire l'addition d'un nombre avec un chiffre multiplié par une lettre, 3i contient deux valeurs de types différents, un entier 3 et un caractère i. L'ordinateur ne sait pas non plus deviner quelle partie est imaginaire ou réelle. Ce qu'on aurait dû faire c'est:  $Z_{1.re} \leftarrow 1$  et  $Z_{2.im} \leftarrow 3$ .

- La ligne 11 n'est pas correct non plus, t.complexe est une structure et non une variable, donc il sert juste à définir une variable et non à attribuer une valeur. au lieu de  $t.complexe.im \leftarrow 4$  on aurait dû juste mettre  $Z_{2.im} \leftarrow 4$ .









①

d) Soit la structure suivante décrivant une heure

```
structure t_heure
{
    entier hh, mm, ss;
};
```

e) Ecrivez les fonctions suivantes :

- Affichage d'une heure
- Saisie d'une heure
- Comparaison de deux heures
- Addition de deux heures

Affichage :

Fonction Affiche\_heure (entier h) : t\_heure h

```
{
    Afficher
    printf ("Il est : ", h.hh, "h", h.mm, "mn", h.ss, "s");
}
```

①

Saisie :

fonction saisir\_h ( )  
{  
  Afficher ("Veuillez saisir l'heure") ;  
  saisir(  
fonction saisir\_h (t-heure h) partie t-heure)  
{  
  Afficher ("Veuillez saisir l'heure") ;  
  saisir (h.h);  
  saisir (h.m);  
  saisir (h.s);  
  retourner h;  
}

Ok - inutile de la mettre en entrée du coup...

Comparaison :

fonction -comparaison-h (t-heure a; t-heure b)  
{  
  si (a.h > b.h)  
  {  
    Afficher ("L'heure a plus grande que la b") ;  
  }  
  sinon  
  {  
    Afficher ("L'heure b plus grande que la a") ;  
  }  
}

minute / seconde ?

k) Ecrire une fonction récursive qui résout le problème des tours de Hanoi pour N disques (vu en cours) – vous pouvez faire des schémas si vous le souhaitez pour vous aider à écrire cette fonction

fonction déplacer (nb-disque, d  
fonction déplacer (entrée: entier nb, départ, intermédiaire, arrivée)  
{  
  si (nb = 0)  
  { afficher (départ, "→", arrivée);  
  }  
  sinon  
  { déplacer (nb - 1, départ, intermédiaire, arrivée);  
    déplacer (1, départ, arrivée, intermédiaire);  
    déplacer (nb - 1, intermédiaire, arrivée, départ);  
  }  
  si .

2



**Thème 3 : VRAI/FAUX**

j) VRAI/FAUX : répondez par *vrai* ou par *faux* aux affirmations suivantes. Une bonne réponse rapporte 1 point.

Une fonction récursive sans condition d'arrêt peut générer un plantage du programme.

☐ VRAI

Une structure contient des champs.

☐ VRAI

Un champ est identifié par son type et son nom.

☒ FAUX

On utilise la notation '.' pour accéder au champ d'une structure si ce champ est un pointeur.

☒ FAUX

Une variable de type structure n'a pas d'adresse.

☐ FAUX

~~1~~ 1,5



Addition (attention aux retenues en additionnant les secondes et les minutes)

fonction addition-h (t-heure a, t-heure b → sortie t-heure)  
{ t-heure X; %: module  
(X.hh = (a.m  
$$X.hh = a.hh + b.hh + ((a.mm + b.mm) / 60)$$
  
$$- \%((a.mm + b.mm) / 60)$$
  
$$X.mm = \%((a.mm + b.mm) / 60) + (a.ss + b.ss) / 60$$
  
$$- \%((a.ss + b.ss) / 60)$$
  
$$X.ss = \%((a.ss + b.ss) / 60)$$
  
renvoyer X;  
}

Ex) Soit la structure suivante décrivant un événement :

```
structure t_evt
{
    entier jour,mois,annee;
    caractere *quoi;
    entier importance;
};
```

Ecrivez la fonction de saisie d'un événement pour laquelle on vous fournit l'entête suivante :

```
fonction saisieEvt(entree : t_evt *evt)
```

note : cette fonction est appelée dans la fonction principale de la manière suivante :

```
fonction principale()
{
    t_evt e;
    saisieEvt(&e);
    ...
}
```

①

```

fonction saisirEvt(entree : t_evt * evt)
{
    (" Veuillez entrer un événement!");
    saisir(*evt, jour);
    saisir(*evt, mois);
    saisir(*evt, annee);
    saisir(*evt, *quoi); erreur seg!
    saisir(*evt, importance);
    retourner *evt;
}
    
```

g) Soit **tabag** un tableau (statique ou dynamique, peu importe) de **t\_evt** dont la taille utile est connue. Ecrire une fonction qui trie un tableau de **t\_evt** par ordre chronologique. L'entête de cette fonction est la suivante :

```
fonction triChrono(entree : t_evt *tab, entier util)
```

Note : cette fonction est appelée dans la fonction principale de la manière suivante :

```

fonction principale()
{
    t_evt      *tabag;
    entier     nbEvt;
    entier     cpt;

    afficher("entrez le nombre d'evenements:");
    saisir(nbEvt);

    tabag ← reservation(nbEvt t_evt);
}
    
```

```

pour cpt de 0 à nbEvt-1
{
    saisieEvt(&tabag[cpt]);
}

triChrono(tabaf, nbEvt);
...
}

```

```

fonction triChrono(entree: t_evt * tab, entier util)
{
    int i = 0; int j = 0;
    t_evt * temp = Null;
    pour (i = 0; i < util; i++)
    {
        pour (j = 0; j < util - i; j++)
        {
            si (tab[i].jour > tab[i+1].jour)
            {
                temp = tab[i];
                tab[i] = tab[i+1];
                tab[i+1] = temp;
            }
            si (tab[i].mois > tab[i+1].mois)
            {
                temp = tab[i];
                tab[i] = tab[i+1];
                tab[i+1] = temp;
            }
            si (tab[i].jour > tab[i+1].jour)
            {
                temp = tab[i];
                tab[i] = tab[i+1];
                tab[i+1] = temp;
            }
        }
    }
    retourner *tab;
}

```



## Thème 2 : un peu de récursivité

h) Qu'est-ce qu'une fonction récursive ?

Une fonction récursive est une fonction qui s'appelle elle-même.

i) Soit la fonction récursive suivante : que fait-elle si on l'appelle avec une valeur initiale de position égale à 0 ? (vous pouvez choisir un texte à afficher en exemple)

```
fonction devinette(entree : caractere *texte, entier position)
{
    si ( position < longueur_texte(texte) )
    {
        afficher(texte[position]);
        devinette(texte, position+1);
    }

    retourner;
}
```

Si on l'appelle avec une valeur initiale de position égale à 0, cette fonction récursive nous affiche tout le texte. tout le texte.

Par exemple pour afficher le texte : "je suis fatigué" la fonction calcule la longueur du texte qui est à peu près égale à 15. Elle compare ensuite sa longueur à la position initiale de 0 (< 15). ensuite elle affiche la première lettre (texte) puis s'appelle elle-même, augmente la position de 1, puis affiche la seconde lettre ainsi de suite jusqu'à ce que position soit égale à la longueur du texte, et là elle affiche tout le texte.