Université Paris-Nord — L1
Institut Galilée — Année 2008-2009
Licence 1 - 1er semestre

Partiel d'Éléments d'Informatique

Durée : 3 heures.

Documents autorisés : Aucun.

Recommandations : Un barême vous est donné à titre indicatif afin de vous permettre de gérer votre temps. La notation prendra en compte à la fois la syntaxe et la sémantique de vos programmes, c'est-à-dire qu'ils doivent compiler correctement. Une fois votre programme écrit, il est recommandé de le faire tourner à la main sur un exemple pour s'assurer de sa correction.

1 Compilation, édition de liens et exécution d'un programme (*5 points*)

Nous étudions un programme qui travaille sur un tableau d'entiers déjà initialisé. Il supprime de ce tableau tous les zéros et met sa taille à jour pour prendre en compte les suppressions. Le programme est le suivant :

```
1   #include <stdlib.h> /* EXIT_SUCCESS */
2
3   /* déclaration constantes et types utilisateurs */
4
5   /* déclaration de fonctions utilisateurs */
6
7   int main()
8   {
9       /* un tableau et sa taille */
10      int tab[5] = {0,1,0,2,3};
11      int taille = 5;
12      int nouvelle_taille = 0; /* la nouvelle taille du tableau modifié */
13      int i; /* var boucle */
14
15      /* suppression des zéros dans tab */
16      for(i = 0;i < taille;i = i + 1)
17      {
18          if(tab[i] != 0) /* valeur à conserver */
19          {
20              tab[nouvelle_taille] = tab[i];
21              nouvelle_taille = nouvelle_taille + 1;
22          }
```

```
23      }
24      /* i >= taille */
25      /* les zéros ont été supprimés,
26         mise à jour de la taille */
27      taille = nouvelle_taille;
28
29      /* affichage du tableau pour vérifier la suppression */
30      printf("taille : %d\n",taille);
31      for(i = 0;i < taille;i = i + 1)
32      {
33          printf("%d ",tab[i]);
34      }
35      /* i >= taille */
36      printf("\n");
37
38      return EXIT_SUCCESS;
39  }
40
41  /* implantation de fonctions utilisateurs */
```

1. Cependant, ce programme est erroné. Expliquez succintement pourquoi le programme ci-dessus est incomplet et ce qu'il manque pour que la compilation réussisse. Corrigez le programme pour qu'il compile.

   Correction. Le programme utilise la fonction `printf`, pour l'affichage du tableau à l'écran, qui est une fonctionnalité disponible dans la bibliothèque `stdio`. Or, ses fonctionnalités ne sont pas déclarées par la directive du pré-processeur : `#include <stdio.h> /* printf */`. La compilation va échouer à la première occurrence de `printf` dans le programme, c'est-à-dire à la ligne 30.

2. Après la compilation, décrivez succintement, en vos propres termes, ce qu'il se passe lors de la création de l'exécutable par l'éditeur de liens.

   Correction. L'édition de liens transforme le code objet résultant de la compilation en fichier exécutable. Durant cette étape, le code objet des fonctions externes (bibliothèques) est ajouté à l'exécutable et le point d'entrée dans le programme est fixé (sur le main).

3. Afin de bien comprendre la sémantique du programme, simulez l'exécution du programme, en réalisant sa **trace** : l'exécution du programme est représenté par un tableau à $n + 2$ colonnes : la première colonne étant le numéro de la ligne exécutée, les n colonnes suivantes, les colonnes des n variables du programme (n à déterminer) et la dernière colonne, la colonne servant à l'affichage à l'écran du programme. Seules les lignes modifiant l'état mémoire du programme sont à reporter dans le tableau.

   Correction. La correction est donnée dans le tableau 1.

ligne	tab[0]	tab[1]	tab[2]	tab[3]	tab[4]	taille	nouvelle_taille	i	Affichage (sortie écran)
initialisation	0	1	0	2	3	5	0	?	
16								0	
23								1	
20	1								
21							1		
23								2	
23								3	
20		2							
21							2		
23								4	
20			3						
21							3		
23								5	
27						3			
30									taille : 3(passage à la ligne)
31								0	
30									1(espace)
34								1	
30									2(espace)
34								2	
30									3(espace)
34								3	
36									(passage à la ligne)

TAB. 1 – Trace du programme donné à l'exercice 1.

2 Les tableaux (*5 points*)

2.1 Calcul du produit des éléments d'un tableau (*2,5 points*)

Écrire le programme qui demande à l'utilisateur d'entrer les valeurs d'un tableau d'entiers de taille `N` (une constante symbolique), et qui calcule et affiche la valeur du produit des entiers du tableau. Deux exemples d'exécution sont les suivants (pour `N` valant 4) :

```
Saisissez 4 entiers : 2 -3 6 -10
Le produit des éléments du tableau vaut 360.

Saisissez 4 entiers : 3 -3 4 0
Le produit des éléments du tableau vaut 0.
```

Correction.

```
#include <stdlib.h> /* EXIT_SUCCESS */
#include <stdio.h> /* printf, scanf */

/* déclaration constantes et types utilisateurs */
#define TAILLE 4 /* taille du tableau utilisateur */

/* déclaration de fonctions utilisateurs */

int main()
{
    int tab[TAILLE]; /* tableau à initialiser par l'utilisateur */
    int produit = 1; /* élément neutre de la multiplication */
    int i; /* var. de boucle */

    /* demande saisie de TAILLE entiers*/
    printf("Saisissez %d entiers : ",TAILLE);

    /* saisie des elts du tableau (TAILLE entiers) */
    for(i = 0;i < TAILLE;i = i + 1) /* chaque case du tableau */
```

```
    {
        /* saisie valeur */
        scanf("%d",&tab[i]);
    }
    /* i >= TAILLE */

    /* calcul produit */
    for(i = 0;i < TAILLE;i = i + 1) /* chaque case */
    {
        /* multiplie le produit partiel par la valeur de la case */
        produit = produit * tab[i];
    }
    /* i >= TAILLE */

    /* produit contient le produit des éléments du tableau */
    printf("Le produit des éléments du tableau vaut %d.\n",produit);

    return EXIT_SUCCESS;
}

/* implantation de fonctions utilisateurs */
```

2.2 Comptage du nombre d'éléments inférieurs à un seuil donné (*2,5 points*)

Écrire un programme qui demande à l'utilisateur d'entrer les valeurs d'un tableau de réels de taille `N` (une constante symbolique), puis la valeur d'un réel `seuil`, et qui affiche combien d'éléments du tableau sont strictement plus petits que `seuil`. Deux exemples d'exécution sont les suivants (pour `N` valant 4) :

```
Saisissez 4 réels : 2.2 4.3 -1.1 0.33
Entrez le seuil : 1.1
Il y a 2 nombre(s) < seuil.

Saisissez 4 réels : 2.2 4.3 -1.1 0.33
Entrez le seuil : 0.33
Il y a 1 nombre(s) < seuil.
```

Correction.

```
#include <stdlib.h> /* EXIT_SUCCESS */
#include <stdio.h> /* printf, scanf */

/* déclaration constantes et types utilisateurs */
#define TAILLE 4 /* taille du tableau utilisateur */

/* déclaration de fonctions utilisateurs */

int main()
{
```

```
    double tab[TAILLE]; /* tableau à initialiser par l'utilisateur */
    double seuil; /* seuil à saisir par l'utilisateur pour filtrer le tableau */
    int plus_petit = 0; /* nombre d'éléments du tableau < seuil */
    int i; /* var. de boucle */

    /* demande saisie de TAILLE réels */
    printf("Saisissez %d réels : ",TAILLE);

    /* saisie des elts du tableau (TAILLE réels) */
    for(i = 0;i < TAILLE;i = i + 1) /* chaque case du tableau */
    {
        /* saisie valeur */
        scanf("%lg",&tab[i]);
    }
    /* i >= TAILLE */

    /* saisie seuil */
    printf("Entrez le seuil : ");
    scanf("%lg",&seuil);

    /* comptage nombre < seuil */
    for(i = 0;i < TAILLE;i = i + 1) /* chaque case */
    {
        if(tab[i] < seuil) /* trouvé plus petit */
        {
            plus_petit = plus_petit + 1; /* un de plus */
        }
    }
    /* i >= TAILLE */

    printf("Il y a %d nombre(s) < seuil.\n",plus_petit);

    return EXIT_SUCCESS;
}

/* implantation de fonctions utilisateurs */
```

3 Une seconde plus tôt, il était exactement... (*5 points*)

Il faut écrire un programme qui demande à l'utilisateur de saisir l'heure sous la forme de 3 entiers (3 variables pour les heures, `h`, les minutes, `m` et les secondes, `s`) et qui affiche l'heure qu'il était 1 seconde plus tôt. Il faudra envisager tous les cas possibles pour le changement d'heure. Deux exemples de sortie sont :

```
Introduire l'heure puis les minutes puis les secondes : 23 12 0
Une seconde plus tôt, il était exactement : 23h11m59s

Introduire l'heure puis les minutes puis les secondes : 0 0 0
Une seconde plus tôt, il était exactement : 23h59m59s
```

1. En s'aidant des exemples donnés plus haut, écrire l'algorithme correspondant en français.

Correction.

```
- saisie de l'heure
- enlève une seconde
- si tour du cadran à l'envers des secondes alors
    - remise à 59 des secondes
    - il est une minute de moins
    - si tour du cadran à l'envers des minutes alors
        - remise à 59 des minutes
        - il est une heure de moins
        - si tour du cadran à l'envers des heures alors
            - remise à 23 des heures
- affichage de l'heure
```

2. En s'aidant éventuellement de la question précédente, écrire le programme.

Correction.

```
#include <stdlib.h> /* EXIT_SUCCESS */
#include <stdio.h> /* printf, scanf */

/* déclaration constantes et types utilisateurs */

/* déclaration de fonctions utilisateurs */

/* fonction principale */
int main()
{
    /* soit l'heure représentée par 3 entiers */
    int h; /* heures */
    int m; /* minutes */
    int s; /* secondes */

    /* saisie de l'heure */
    printf("Introduire l'heure puis les minutes puis les secondes : ");
    scanf("%d",&h);
    scanf("%d",&m);
    scanf("%d",&s);

    /* calcul de la nouvelle heure */
    s = s - 1; /* une seconde plus tôt */

    if(s == -1) /* tour du cadran à l'envers des secondes */
    {
        /* remise à 59 */
        s = 59;

        /* une minute de moins */
        m = m - 1;

        if(m == -1) /* tour du cadran à l'envers des minutes */
        {
```

```
                /* remise à 59 */
                m = 59;

                /* une heure de moins */
                h = h - 1;

                if(h == -1) /* tour du cadran à l'envers des heures */
                {
                    /* remise à 23 */
                    h = 23;
                }
            }
        }
        /* h,m,s contiennent l'heure une seconde plus tôt */

        /* affichage de l'heure */
        printf("Une seconde plus tôt, il était exactement : %dh%dm%ds\n",h,m,s);

        return EXIT_SUCCESS;
    }

    /* implantation de fonctions utilisateurs */
```

4 Affichage d'un triangle isocèle d'étoiles (*5 points*)

Écrire un programme qui, étant donné un entier naturel impair `base`, affiche un triangle isocèle d'étoiles, ayant pour base, `base` étoiles. La valeur de la `base` sera saisie par l'utilisateur et on considérera qu'il saisit bien un nombre impair. Trois exemples d'exécution sont les suivants :

```
Nombre d'étoiles à la base du triangle (impair) ?
5
  *
 ***
*****

Nombre d'étoiles à la base du triangle (impair) ?
3
 *
***

Nombre d'étoiles à la base du triangle (impair) ?
1
*
```

1. Quelle est la série d'entiers correspondant aux nombres d'étoiles à afficher par ligne ?

   Correction. C'est la série des nombres impairs de 1 à `base`.

2. Combien y a-t-il de blancs à afficher sur une ligne, pour un nombre n d'étoiles sur la ligne ?

Correction. Il y a $(base - n)/2$ blancs à afficher par ligne.

3. Écrire le programme.

Correction.

```
#include <stdlib.h> /* EXIT_SUCCESS */
#include <stdio.h> /* printf, scanf */

/* déclaration constantes et types utilisateurs */

/* déclaration de fonctions utilisateurs */

int main()
{
    int base; /* un nombre impair d'étoiles à la base du triangle, saisi par l'utilisate
    int i; /* var. de boucle */
    int j; /* var. de boucle */

    /* saisie base (impair) */
    printf("Nombre d'etoiles à la base du triangle (impair) ?\n");
    scanf("%d",&base);

    /* affichage triangle isocèle d'étoiles */
    for(i = 1;i <= base;i = i + 2) /* chaque nombre d'étoiles à afficher (base impaire)*
    {
        /* affiche les blancs */
        for(j = 0;j < (base - i) / 2;j = j + 1) /* (base - i) / 2 fois */
        {
            /* affiche un blanc */
            printf(" ");
        }
        /* j >= (base - i) /2 */

        /* affiche les étoiles */
        for(j = 0;j < i;j = j + 1) /* i fois */
        {
            /* affiche une étoile */
            printf("*");
        }
        /* j >= i */

        /* passe à la ligne suivante */
        printf("\n");
    }
    /* i >= base */

    return EXIT_SUCCESS;
}

/* implantation de fonctions utilisateurs */
```

Question bonus

Compléter le programme précédent afin qu'il affiche un losange d'étoiles, `base` étant le nombre impair d'étoiles sur la ligne du milieu. Deux exemple d'exécution sont les suivants :

```
Base du losange ?
5
  *
 ***
*****
 ***
  *

Base du losange ?
7
   *
  ***
 *****
*******
 *****
  ***
   *
```