Université Paris-Nord — L1
Institut Galilée — Année 2011-2012
Licence 1 - 1er semestre

Travaux dirigés 7 : lecture de données au clavier.

Correction. Note aux chargés de TD.

- Les traces ne définissent pas une colonne supplémentaire pour les entrées, ce qui ne permet pas de présenter les subtilités de conversion de scanf. Cela sera fait en S2.
- Bien insister sur la procédure permettant d'attaquer un problème de programmation. Il semble qu'ils ne savent toujours pas l'appliquer. La procédure :
  - on se donne des exemples
  - on trouve un algorithme en francais
  - on traduit l'algorithme en C en s'aidant de commentaires
  - on test sur les exemples qu'on s'est donnes
- L'algorithmique des tableaux parle de cases, et un parcours se présente de la façon suivante :

```
- pour chaque case du tableau :
    - affecter 2 a la case
    - afficher la case
    - ...
```

  En C, on peut être plus tenté par : pour chaque indice de case. Cependant, la plupart des langages modernes (Java, Python etc.) ont introduit des conteneurs avec des parcours qui font référence aux éléments. On reste sur les cases pour l'instant et il faut sans doute expliquer la subtilité.
- N.B. : le "programme vide" voit son écriture simplifiée en n'indiquant plus dans le main la déclaration des variables, ni la valeur de retour. Si certains ont encore des problèmes avec ça, il faut repousser.

1 Lecture de données utilisateur entrées au clavier

1. Que fait le programme suivant ?

```
1   /* declaration de fonctionnalites supplementaires */
2   #include <stdlib.h> /* EXIT_SUCCESS */
3   #include <stdio.h> /* printf, scanf */
4
5   /* declarations des constantes et types utilisateurs */
6
7   /* declarations des fonctions utilisateurs */
8
9   /* fonction principale */
10  int main()
11  {
12      int a;
13      double b;
14      char c;
15
16      printf("Entrez un nombre entier puis un nombre réel puis un caractère : ");
```

```
17
18      scanf("%d",&a);
19      scanf("%lg",&b);
20      scanf(" %c",&c);
21
22      printf("Vous avez saisi %d puis %g puis %c.\n",a,b,c);
23
24      return EXIT_SUCCESS;
25  }
26
27  /* definitions des fonctions utilisateurs */
```

Correction. Le programme :
- déclare 3 variables a,b et c, respectivement de type entier, réel (rationnel) et caractère ;
- demande à l'utilisateur de saisir 3 valeurs, respectivement de type entier, réel (rationnel) et caractère ;
- affecte la valeur entière saisie à la variable entière a (même type sinon quelle est la signification ? le compilateur détecte cette erreur sémantique lors de l'analyse sémantique mais il acceptera de compiler en faisant une conversion automatique de type => source de bugs difficiles à détecter)
- affecte la valeur réelle saisie à la variable réelle/rationnelle b ;
- affecte le caractère saisie à la variable caractère c ;
- affiche les valeurs pour montrer les affectations réalisées (vous pouvez utiliser un tel programme pour vérifier que les représentations sont bornées, cf. cours et TP)

2. Faire la trace du programme en considérant que l'utilisateur saisit au clavier : 1 puis "entrée", 12.2 puis "entrée" et 'c' puis "entrée" .

Correction.

```
ligne          | a      | b      | c      | affichage (sortie/écriture à l'écran
-------------------------------------------------------------------------------
initialisation | ?      | ?      | ?      |
16             |        |        |        | Entrez un nombre entier puis un nomb
18             | 1      |        |        |
19             |        | 12.2   |        |
20             |        |        | 'c'    |
22             |        |        |        | Vous avez saisi 1 puis 12.2 puis c.
```

Ils l'ont vu en cours : vous pouvez leur faire remarquer que si la lecture du caractère s'était faite avec `"%c"`, la var c contiendrait `'\n'`, caractère qui suit la chaîne "12.2" (due à la mémoire tampon + scanf).

1.1 Moyenne d'une série d'entiers saisie par l'utilisateur

Écrire un programme qui demande à l'utilisateur combien d'entiers composent sa série, lit la série d'entiers et affiche la moyenne des valeurs de la série. L'ensemble de la série ne doit pas être stockée en mémoire.

Correction.

```
/* declaration de fonctionnalites supplementaires */
#include <stdlib.h> /* EXIT_SUCCESS */
```

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

/* declaration constantes et types utilisateurs */

/* declaration de fonctions utilisateurs */

/* fonction principale */
int main()
{
    int n; /* taille de la serie a saisir par l'utilisateur*/
    int elt; /* un element de la serie a saisir par l'utilsateur */
    double somme = 0.0; /* somme de la serie a calculer pour afficher la moyenne
                           c'est un double car sinon le C fait une division entiere
    int i; /* var. de boucle */

    /* demande la taille de la serie a l'utilisateur */
    printf("Combien d'elements dans la série ? ");
    scanf("%d",&n);

    /* saisie serie (n entiers) et calcul incremental de la somme */
    for(i = 0;i < n;i = i + 1) /* chaque entier de la serie */
    {
        /* saisir sa valeur */
        scanf("%d",&elt);

        /* l'ajoute a la somme partielle */
        somme = somme + elt;
    }
    /* i >= n */

    /* somme contient la somme des elements de la serie.
       la moyenne est somme / n */
    printf("La moyenne des valeurs de cette serie est : %g\n",somme / n); /* ce n'es

    return EXIT_SUCCESS;
}

/* implantation de fonctions utilisateurs */
```

1.2 Initialisation d'un tableau par l'utilisateur

Écrire un programme qui déclare un tableau d'entiers de taille arbitraire TAILLE (une constante symbolique) et qui demande à l'utilisateur de saisir au clavier les valeurs des cases du tableau. Afficher le tableau.

Correction.

```
/* declaration de fonctionnalites supplementaires */
#include <stdlib.h> /* EXIT_SUCCESS */
#include <stdio.h> /* printf, scanf */

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

```
/* declaration de fonctions utilisateurs */

/* fonction principale */
int main()
{
    int tab[TAILLE]; /* tableau a initialiser par l'utilisateur */
    int i; /* var. de boucle */

    /* demande a l'utilisateur de saisir 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 */
    {
        /* saisir sa valeur */
        scanf("%d",&tab[i]);
    }
    /* i >= TAILLE */

    /* affichage du tableau */
    printf("Vous avez saisi le tableau suivant : ");

    for(i = 0;i < TAILLE;i = i + 1) /* chaque case du tableau */
    {
        /* afficher sa valeur */
        printf("%d ",tab[i]);
    }
    /* i >= TAILLE */
    printf("\n");

    return EXIT_SUCCESS;
}

/* implantation de fonctions utilisateurs */
```

Faire la trace, en donnant à l'avance les valeurs saisies. On considère, pour simplifier dans un premier temps (S1), que "entrée" est utilisée après chaque entier.

2 En TP : le nombre secret

Nous voulons programmer le jeu du nombre à découvrir. Le joueur doit deviner un nombre secret choisit par l'ordinateur entre 0 et `NB_MAX` (une constante du programme). S'il propose un nombre trop grand, l'ordinateur lui répond "Plus petit", s'il propose trop petit, l'ordinateur lui répond "Plus grand". Dans ces deux cas, il est invité à proposer un autre nombre. Le jeu s'arrête quand il devine juste. Un exemple d'exécution de ce jeu pourrait être :

```
Votre choix ?
8
Plus petit.
Votre choix ?
4
```

```
Plus petit.
Votre choix ?
2
Vous avez trouvé le nombre secret.
```

1. Proposez un algorithme en français pour le jeu.
2. Traduisez-le en langage C et exécutez-le.
3. Pourquoi préférez-vous une boucle `while` ici ?

Pour rendre le jeu intéressant, l'ordinateur doit choisir le nombre secret *au hasard*. La librairie C standard propose des fonctions renvoyant des nombres pseudo-aléatoires [1] déclarées dans `<stdlib.h>`. L'ordinateur va utiliser la fonction : `int rand();` qui renvoie un nombre pseudo-aléatoire entier entre 0 et la constante `RAND_MAX` (égale à 2147483647) inclus. Pour renvoyer un nombre pseudo-aléatoire entre 0 et `NB_MAX`, `NB_MAX` inclus (`NB_MAX` << `RAND_MAX`), il suffit de calculer le reste de la division entière de `rand()` par `(NB_MAX + 1)`, c'est-à-dire le nombre renvoyé par rand() modulo `(NB_MAX + 1)` (opérateur % en C). rand vient de random qui veut dire aléatoire en anglais.

Un exemple de programme illustrant l'utilisation de rand pour engendrer un nombre pseudo-aléatoire est le suivant :

```
#include <stdlib.h> /* EXIT_SUCCESS, rand, srand */
#include <time.h> /* time */

#define NB_MAX 15 /* nombre secret entre 0 et NB_MAX inclus */

int main()
{
    int nombre_secret; /* nombre secret à deviner */

    /* initialisation du générateur de nombres pseudo-aléatoires */
    srand(time(NULL)); /* à ne faire qu'une fois */

    /* tirage du nombre secret */
    nombre_secret = rand() % (NB_MAX + 1); /* entre 0 et NB_MAX inclus */

    /* manche joueur ... */

    return EXIT_SUCCESS;
}
```

Correction.

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

#define TRUE 1
#define FALSE 0

#define NB_MAX 15 /* nombre secret entre 0 et NB_MAX inclus */

/* declaration de fonctions utilisateurs */
```

1. http ://fr.wikipedia.org/wiki/Générateur_de_nombres_pseudo-aléatoires

```
int main()
{
    int nombre_secret; /* nombre secret à deviner */
    int choix; /* choix de l'utilisateur pour le nombre secret */
    int pas_trouve = TRUE; /* TRUE si pas trouvé nombre secret */

    /* initialisation du générateur de nombres pseudo-aléatoires */
    srand(time(NULL)); /* à ne faire qu'une fois */

    /* tirage du nombre secret */
    nombre_secret = rand() % (NB_MAX + 1); /* entre 0 et NB_MAX inclus */

    /* manche joueur */
    while(pas_trouve) /* pas trouvé nombre secret */
    {
        /* demande nombre à l'utilisateur */
        printf("Votre choix (nombre entre 0 et %d) ?\n",NB_MAX);
        scanf("%d",&choix);

        if(choix == nombre_secret) /* trouvé */
        {
            pas_trouve = FALSE;
        }
        else /* pas trouvé */
        {
            /* donne indice */
            if(choix > nombre_secret)
            {
                printf("Plus petit.\n");
            }
            else
            {
                printf("Plus grand.\n");
            }
        }
    }
    /* trouvé nombre secret */

    printf("Vous avez trouvé le nombre secret.\n");

    return EXIT_SUCCESS;
}

/* Definition de fonctions utilisateurs */
```