Licence 1 - section B
TD 2 d'éléments d'informatique

Catherine RECANATI – Département d'Informatique – Institut Galilée

Semaine du 7 au 11 novembre 2016

1 Déclaration et affectation de variables

Exercice 1.1 Trace de programme.

Soit le programme suivant :

```
1         /* declaration de fonctions ou constantes externes */
2  #include <stdlib.h>    /* pour EXIT_SUCCESS */
3  #include <stdio.h>    /* pour la fonction printf */
4
5   /* fonction principale */
6   int main()
7   {
8      /* declaration de variables locales */
9      int x;
10
11     x = 3;      /* initialisation de x */
12     x = x + 1;
13     printf("x = %d\n", x);
14
15     /* valeur de retour de la fonction main */
16     return EXIT_SUCCESS;
17  }
```

1. Que fait ce programme ?
2. Donner la trace du programme C. Pour cela vous utiliserez un tableau comportant 1 colonne pour le numéro de ligne + autant de colonnes que de variables utilisées dans le programme + 1 colonne pour les affichages éventuels du programme sur la console.

Exercice 1.2 Echange du contenu de 2 variables.

Modifiez le programme précédent pour qu'il déclare 2 variables entières `x` et `y`, plus une variable auxiliaire appelée `sauve`. La fonction main principale initialisera la variable x à 3 et la variable y à 0, affichera les valeurs de x et de y, puis échangera leurs valeurs en utilisant la variable auxiliaire `sauve`. La fonction réaffichera ensuite la valeur de x et de y à la console pour s'assurer que l'échange a bien été effectué.

2 Lire et imprimer des variables

On a vu en cours la fonction printf qui utilise des indications de formats avec le symbole %, comme par exemple %d indiquant un format d'impression décimal. Il existe d'autres formats reconnus par printf comme le format %x des nombres hexadécimaux. Le tableau suivant indique divers formats que vous pourrez tester dans un programme :

Formats	Sorties	Exemples
%d ou %i	Entier decimal signe	392
%u	Entier decimal non signe	7235
%o	Octal non signe	610
%x	Entier hexadecimal non signe	7fa
%X	Entier hexadecimal non signe	7FA

```
%f          Decimal virgule flottante, minuscules                 392.65
%F          Decimal virgule flottante, majuscules                 392.65
%e          Notation scientifique (mantisse/exposant min)        3.9265e+2
%E          Notation scientifique (mantisse/exposant maj)        3.9265E+2
%g          Utilise la representation la + courte: %e ou %f         392.65
%G          Utilise la representation la + courte: %E ou %F         392.65
%c          Caractere (char)                                          a
%s          Chaine de caracteres                                 bonjour
%p          Adresse  (pointeur)                                  b8000000
```

A noter que pour les flottants, on peut indiquer le nombre de caractères figurant avant et après la virgule, comme avec `%3.1f` pour obtenir l'affichage 392.6 au lieu de 392.65.

La fonction qui permet de lire des caractères du clavier pour initialiser une variable s'appelle `scanf`. Elle prend (comme `printf`) en premier argument une chaine de caractères contenant des indications de format, et en second argument l'adresse en mémoire d'une variable. Pour connaitre l'adresse d'une variable, on lui appliquera l'opérateur & (prononcer "et commercial" ou "esperluette").

Ainsi, les lignes de programme suivantes permettent de lire une variable initialisée d'abord à zéro à partir des caractères entrés au clavier par l'utilisateur du programme :

```
1    int variable = 0;
2    scanf ("%d", &variable);   /* & est l'operateur d'adresse */
3    printf ("%d\n", variable);
```

La plupart des formats d'impression de printf du tableau précédent restent valables comme format de lecture pour `scanf`. Ainsi on pourra utiliser `%d %i` et `%u` pour lire des décimaux,`%x` et `%X` pour des hexadécimaux, `%o` pour des nombres en notation octale, `%e` , `%f` , et `%g` pour lire des float, et `%lf` pour des double (le l devant le f signifie long). Enfin `%c` et `%s` permettront de lire respectivement un caractère et une chaine de caractères.

A noter que si l'on préfixe l'indicateur de format d'un nombre, cela précisera le nombre de caractères qui seront lus par `scanf`.

```
1   int nb_de_5_chiffres = 0;
2   scanf ("%5d", &nb_de_5_chiffres);
3   printf ("---> %d\n", nb_de_5_chiffres);
```

Exécutions :

```
18
---> 18
1000
---> 1000
1584669842
---> 15846
```

Mais attention, si l'utilisateur tape plus de caracères que nécessaire, les caracères supplémentaires ne seront pas lus mais resteront dans le tampon qui lit les entrées. C'est eux qui se présenteront ensuite pour être lus si on fait un nouvel appel à la fonction scanf.

Exercice 2.1 Lire et imprimer une variable.

1. Ecrire un programme dont la fonction principale lit une variable x de type unsigned long int avec scanf et l'imprime ensuite dans les formats %u, %o et %X.
2. Modifier le programme précdent pour que la fonction main appelle une fonction imprime(x) qui imprime son argument dans les formats %u, %o et %X.

3 Quelques petits programmes (à faire tourner sur machine en TP)

Exercice 3.1 Majeur ou mineur ?

Écrire un programme qui demande à l'utilisateur d'entrer son âge et affiche ensuite si la personne est majeure ou mineure.

Exercice 3.2 Année bissextile ?

Écrire un programme qui demande à l'utilisateur d'entrer une année et qui affiche ensuite si l'année est bissextile ou non. On rappelle qu'une année est bissextile dans les deux cas suivants :

1. si l'anne est divisible par 4 et non divisible par 100, ou
2. si l'anne est divisible par 400.

Donc en particulier l'année 1900 n'est pas bissextile. Par contre l'an 2000 l'est.

Exercice 3.3 Minimum de 3 valeurs.

Soient 3 variables `a`, `b`, `c`, initialisées au clavier par l'utilisateur. Écrire une fonction minimum (a, b, c) qui calcule et retourne le minimum de ces 3 valeurs. Appeler la fonction dans le programme principal et imprimer le résultat.