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

Travaux pratiques 3 : Premier pas en langage C : variables impératives, printf et structure de contrôle "if"

L'objectif de ce TP est de vous permettre de suivre la chaîne de compilation présentée en cours pour écrire et exécuter vos programmes de façon autonome.

Correction. C'est leur premiere manip. de shell pour utiliser gcc, lancer le prog etc. On donne les points à suivre mais il vaut mieux les leur expliquer.

1 Introduction

Vous allez mettre tous vos programmes écrits dans ce TP dans le répertoire TP3 :

- À partir du début de votre arborescence, créez le répertoire TP3 : `mkdir TP3`
- Allez dans ce répertoire pour y mettre des fichiers : `cd TP3`
- Créez un nouveau fichier source pour le langage C : `gedit exo1.c &` ou `emacs exo1.c &`
- Après avoir fini d'écrire votre programme, enregistrez le fichier source sur le disque dur (la mémoire secondaire).
- Créez un programme exécutable en lançant la compilation et l'édition de liens : `gcc -Wall exo1.c -o exo1`
- Quand l'étape précédente a réussi, exécutez le programme pour vérifier qu'il fonctionne : `./exo1` ou `exo1`

Pour gagner du temps, vous pouvez écrire le préambule dans un fichier à part pour vous permettre de faire des copier-coller du préambule dans vos nouveaux programmes :

- (si votre répertoire courant est TP3 uniquement) Retourner au début de votre arborescence : `cd ..`
- À partir du début de votre arborescence, créez le répertoire *exemples_src* à l'aide de la commande : `mkdir exemples_src`
- Allez dans ce répertoire pour y mettre des fichiers en tapant la commande : `cd exemples_src`
- Créez un nouveau fichier source pour le langage C : `gedit programme_vide.c &` ou `emacs programme_vide.c &` ou `kate programme_vide.c &` ou `kwrite programme_vide.c &`
- Après avoir fini d'écrire votre programme, enregistrez le fichier source sur le disque dur (la mémoire secondaire).

Vous pouvez utiliser à tout moment la commande "ls" (list directory) pour voir la liste des fichiers d'un répertoire.

2 Affichage

1. Écrire un programme qui affiche à l'écran "coucou".

2. Modifier ce programme pour qu'il affiche à l'écran "coucou" sur 5 lignes de deux façons :
   - avec 5 printf ;
   - avec un seul printf.
3. Écrire un programme qui affiche à l'écran l'évaluation de l'expression 7 * 3 + 2.
4. Modifier ce programme pour qu'il affiche à l'écran l'évaluation de l'expression x * 3 + 2, avec la variable entière `x` initialisée à une valeur quelconque.

Correction. Durée 1/2 heure ?

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

/* declaration constantes et types utilisateurs */

/* declaration de fonctions utilisateurs */

/* fonction principale */
int main()
{
    /* affiche coucou sur une ligne 5 fois */
    printf("coucou\ncoucou\ncoucou\ncoucou\ncoucou\n");

    /* valeur fonction */
    return EXIT_SUCCESS;
}

/* implantation de fonctions utilisateurs */
```

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

/* declaration constantes et types utilisateurs */

/* declaration de fonctions utilisateurs */

/* fonction principale */
int main()
{
    /* declaration et initialisation variables */
    int x = 3;

    printf("x * 7 + 2 = %d\n",x * 7 + 2);

  /* valeur fonction */
```

```
  return EXIT_SUCCESS;
}

/* implantation de fonctions utilisateurs */
```

3 Faut-il sortir le chien ?

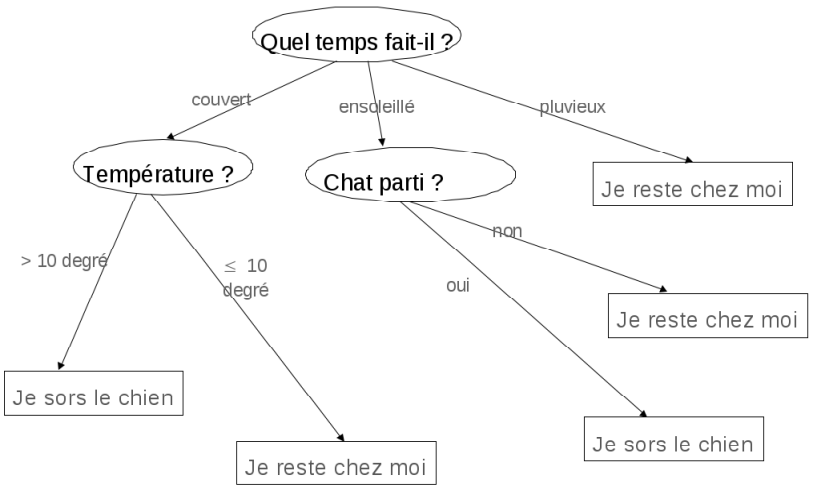

FIG. 1 – Décider si je sors le chien ou je reste chez moi avec l'arbre de décision

Un arbre de décision[1] est un graphe particulier où les nœuds sont des questions et les arêtes sont les réponses à ces questions. Il se lit de haut en bas. On avance dans l'arbre en répondant aux questions. Les nœuds les plus bas jouent le rôle particulier de classes de réponse au problème initial. Ici, il y a deux classes de réponse : *« Je sors le chien »* et *« Je reste chez moi »*. Par exemple, si le temps est couvert, qu'il fait 15 degrés et que le voisin n'est pas parti avec son chat, je sors le chien.

Soient 3 variables entières représentant les propriétés du jour courant pour prendre la décision :

- `temps` est la variable représentant le temps qu'il fait ; elle contient une valeur pour PLUVIEUX, une pour ENSOLEILLÉ et une pour COUVERT.
- `temperature` est la variable représentant la température en degrés Celsius.
- `chat_parti` est la variable représentant le fait que le voisin est parti avec son chat ou non ; elle contient une valeur pour NON et une pour OUI.

Écrire un programme implantant l'arbre de décision pour proposer une réponse étant donné un jour. Après chaque test effectué sur une variable, vous afficherez sa valeur afin de suivre la progression dans l'arbre. La procédure à suivre est rappelée :

1. Se donner des exemples de problèmes et les résoudre à la main.
2. Écrire un algorithme en français permettant de résoudre ces problèmes.

[1] (cf. `http://fr.wikipedia.org/wiki/Arbre_de_décision`) Les arbres de décision sont très utilisés en informatique pour prendre des décisions automatiquement. Ils sont soit programmés par un humain soit appris automatiquement par un algorithme d'apprentissage automatique.

3. Traduire l'algorithme en langage C, en utilisant l'algorithme pour les commentaires.
4. Tester le programme sur les exemples pour s'assurer de sa correction.

Correction. Durée 1 heure ?

Les printf indiquant les tests effectués sont à ajouter.

```
algo :
si couvert alors
  si temp > 10 alors
    sors le chien
  sinon
    reste chez moi
si ensoleille alors
  si voisin absent alors
    sors le chien
  sinon
    reste chez moi
si pluvieux alors
  reste chez moi
```

Les choix sont mutuellement exclusifs.

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

/* declaration constantes et types utilisateurs */

/* declaration de fonctions utilisateurs */

/* fonction principale */
int main()
{
    /* declaration et initialisation variables */
    int temps = 2; /* temps : 0 COUVERT, 1 ENSOLEILLE, 2 PLUVIEUX */
    int temperature = 15; /* temperature en degres celsius */
    int chat_parti = 1; /* 0 NON, 1 OUI */

/*
   si couvert alors
     si temperature > 10 alors
       sors le chien
     sinon
       reste chez moi
   si ensoleille alors
     si voisin absent alors
       sors le chien
     sinon
```

```
       reste chez moi
   et si pluvieux alors
     reste chez moi
*/

    /* cas mutuellement exclusif */

    if(temps == 0) /* COUVERT */
    {
        if(temperature > 10)
        {
            /* on sort le chien */
            printf("Je sors le chien.\n");
        }
        else /* temperature <= 10 */
        {
            /* je reste chez moi */
            printf("Je reste chez moi.\n");
        }
    }

    if(temps == 1) /* ENSOLEILLE */
    {
        if(chat_parti == 1) /* OUI */
        {
            /* on sort le chien */
            printf("Je sors le chien.\n");
        }
        else /* NON */
        {
            /* je reste chez moi */
            printf("Je reste chez moi.\n");
        }
    }

    if(temps == 2) /* PLUVIEUX */
    {
        /* je reste chez moi */
        printf("Je reste chez moi.\n");
    }

    return EXIT_SUCCESS;
}

/* implantation de fonctions utilisateurs */
```