PROGRAMMATION C
TD3

LICENCE MATHS-INFO
30 JANVIER 2012

Exercice 1

Nous avons déjà vu comment construire un `makefile` à partir d'un arbre des dépendances. Faites l'inverse en construisant l'arbre des dépendances à partir du `makefile` :

```
all: executable
executable: file1.o file2.o
    gcc -o $@ $^
file1.o: file1.c file1.h
    gcc -c $<
file2.o: file2.c file1.h file2.h
    gcc -c $<
clean:
    rm file1.o file2.o executable
```

Exercice 2

Déterminer ce que fait le programme ci-dessous lorsque l'on compile avec ou sans l'option `-D DEBUG`.

Contenu du fichier `exo2.h`

```
#ifndef EXO2_H
#define EXO2_H

#define x 1+2

#define A 0
#define B A
#define A 1

#endif
```

Contenu du fichier `exo2.c`

```
#include <stdio.h>
#include "exo2.h"

int main(void){
  int n = x+3;
  n += x*3;
  #ifdef DEBUG
    printf("n=%d at %d\n",n,__LINE__);
  #endif
  n += B;
  #ifdef DEBUG
    printf("n=%d at %d\n",n,__LINE__);
  #endif
  printf("%d\n",n);
  return 0;
}
```

Exercice 3

- Définissez un type `Heure` avec trois champs numériques (heures, minutes, secondes).
- Ecrivez une fonction d'affichage `void printHeure(Heure heure)`.
- Ecrivez une fonction `int compareHeures(Heure heure1, Heure heure2)` qui compare deux heures : elle renvoie -1 si `heure1` précède `heure2`, 0 si elles sont égales, 1 sinon.
- Ecrivez une fonction qui ajoute une durée en secondes à une heure, et dont l'en-tête est `Heure ajoute(Heure heure, int secondes)`.
- Définissez un type `Date` avec trois champs correspondant au jour, au mois et à l'année. Le champs du mois sera un type `Mois` préalablement créé et pouvant prendre 12 valeurs uniquement. Les autres champs sont numériques.
- Ecrivez une fonction `int Mois2int(Mois m)` de conversion d'un mois en un entier (janvier=1, février=2, etc.).
- Ecrivez une fonction `int compareDates(Date d1, Date d2)` qui compare deux dates : elle renvoie -1 si `d1` précède `d2`, 0 si elles sont égales, 1 sinon.
- Créez un type `Timestamp` permettant de stocker une date et une heure et écrivez une fonction `int compareTimestamps(Timestamp ts1, Timestamp ts2)` qui compare deux `Timestamp` : elle renvoie -1 si `ts1` précède `ts2`, 0 s'ils sont égaux, 1 sinon.

Exercice 4

Définissez un nouveau type `Carte` qui permet, pour une variable de ce type, de choisir une des quatre familles (pique, coeur, carreau, trèfle) et une valeur parmi huit (7, 8, 9, 10, valet, dame, roi, as).

Dans un tableau de taille 32, créez un jeu traditionnel de 32 cartes.

Pour commencer à modéliser le jeu de « bataille », définissez un nouveau type `Coup` qui prend trois valeurs : gagnant, perdant, bataille.

Ecrivez une fonction `jouerCoup` qui prend deux cartes en argument et renvoie une variable de type `Coup` : si la première carte a une valeur supérieure (reps. inférieure) à la seconde, le coup sera gagnant (resp. perdant) ; si les valeurs sont les mêmes, il y aura bataille.