3ETI, Entrainement Examen [CSC2]
Developpement Logiciel en C
CPE Lyon

2012-2013 (entrainement)
durée 1h20

Tous documents et calculatrices autorisés.

Le sujet comporte 5 pages

Le temps approximatif ainsi que le barème sont indiqués pour les grandes parties. Notez que le barème est donné à titre purement indicatif et pourra être adapté par la suite.

En cas de doute sur la compréhension de l'énoncé, explicitez ce que vous comprenez et poursuivez l'exercice dans cette logique.

Note préliminaire:

- Toutes les questions concernent le language de programmation C dans le cadre du developpement de logiciels sur architecture PC standard.
- Nous vous invitons à commenter le code et les réponses. En particulier, en cas d'ambiguité de compréhension d'une question, ajoutez toutes remarques et illustrations supplémentaires permettant d'expliquer votre démarche.
- Sauf mention contraire explicite, on supposera que l'on dispose d'un système Linux standard fonctionnant correctement sur un PC récent 32 ou 64 bits (identiques aux conditions de TP des PC de CPE: *int* étant encodé sur 4 octets, pointeurs étant encodés respectivement sur 4(/8) octets sur 32(/64) bits).
- On supposera que le code C est compilé avec une version récente de gcc sous la norme C99 (ou ultérieure) identique aux conditions de TP des PC de CPE.
- On supposera dans chaque cas que les `#include` des en-tête standards nécessaires à la bonne compilation et exécution des programmes décrits sont correctement installés et appelés (ex. stdio, stdlib, string, math, etc).

1 Généralités de programmation

[30min max] 3 points

Répondez le plus précisément aux questions suivantes en quelques lignes:

Question 1

Qu'est ce que gcc*, quel est son rôle?*

Question 2

Qu'est ce qu'un Makefile, quel est son rôle?

Question 3 *Quelle commande permet de visualiser le code assembleur généré par* gcc *lors de son appel sur un fichier contenant du code C?*

Question 4 *En C, dans quels cas passe t-on un paramètre d'une fonction par pointeur (au moins 2 cas différents) ?*

Question 5 *Quels sont les différents types de tests d'un logiciel? Quels sont leurs rôles respectifs?*

Question 6 *Pourquoi doit-on toujours initialiser un pointeur à* `NULL` *lorsqu'on ne peut pas lui affecter directement sa valeur finale?*

2 Manipulation de données

[50min max] 5 points

Supposons que l'on place dans un même répertoire les 3 fichiers suivants modélisant une abstraction d'ordinateur:

- `ordinateur.h` contenant les en-têtes de fonctions et définition de structures associées à un ordinateur.
- `ordinateur.c` contient les corps de fonctions (=implémentations) associés à un ordinateur.
- `main.c` contient la fonction `main()` et les appels généraux.

Le code contenu dans le fichier `ordinateur.h` est le suivant:

[ordinateur.h]

```
#ifndef ORDINATEUR_H
#define ORDINATEUR_H

//structure contenant une longueur et une largeur
struct dimension
{
  float longueur;
  float largeur;
};

//un type de carte mere
struct type_carte_mere
{
  int max_ram; //quantitee max de ram admissible
  int nbr_usb; //nombre de ports USB
  struct dimension; //dimension de la carte
};

//un type de RAM
struct type_ram
{
  int quantitee; //nombre de Giga de RAM
  int frequence; //frequence des barettes
};

//un processeur central
struct type_processeur
{
  int frequence; //frequence du processeur
  int nbr_coeur; //nombre de coeurs
};

//les marques possibles de carte graphiques
enum marque_carte_graphique {NVIDIA,ATI,INTEL,APPLE};

//une carte graphique
struct type_carte_graphique
{
  enum marque_carte_graphique marque;
  int nbr_coeur; //nombre de coeurs de calculs
  int frequence; //frequence des coeurs
  int quantitee_vram; //quantitee de memoire graphique
};

//l'entitee ordinateur contenant les differentes pieces
struct ordinateur
{
  struct type_carte_mere carte_mere;
  struct type_ram ram;
  struct type_processeur processeur;
  struct type_carte_graphique carte_graphique;
};
```

Les en-têtes des fonctions suivantes y sont également définies:

[Ordinateur.h]

```
//initialise un ordinateur a une configuration standard de PC de CPE
void ordinateur_initialise_pc_cpe(struct ordinateur* pc);

//initialise un ordinateur a une configuration standard de
//   MAC type Mac Book Air
void ordinateur_initialise_mac_book_air(struct ordinateur* pc);

//initialise un ordinateur a une configuration standard de
//   PC pour un joueur
void ordinateur_initialise_pc_joueur(struct ordinateur* pc);


#endif
```

Le code contenu dans `ordinateur.c` n'est pas fourni mais implémente correctement ces 3 fonctions.

Le fichier `main.c` est le suivant:

[main.c]

```
#include "ordinateur.h"

#ifdef LOTO
#define ACTUEL mac_book_air
#else
#define ACTUEL pc_cpe
#endif

int main()
{
  struct ordinateur pc_cpe;
  ordinateur_initialise_pc_cpe(&pc_cpe);

  struct ordinateur pc_joueur;
  ordinateur_initialise_pc_joueur(&pc_joueur);

  printf("RAM a CPE: %d\n", X1);

  enum marque_carte_graphique_joueur=X2;
  printf("Carte graphique PC de joueur: %s\n",
                 nom_marque(marque_pc_joueur));


  const ordinateur* le_plus_rapide=ordinateur_le_plus_rapide(&pc_joueur,
                                                             &pc_cpe);

  struct ordinateur pc_a_moi;
  ordinateur_initialise_ACTUEL(&pc_a_moi);

  return 0;

}
```

2.1 Champs d'une struct

Question 7 *Par quoi doit-on remplacer l'instruction* `X1` *pour demander l'affichage de la quantitée de RAM de la variable* `pc_cpe`*?*

Question 8 *Par quoi doit-on remplacer l'instruction* `X2` *pour stocker la marque de la carte graphique de la variable* `pc_joueur`*?*

2.2 Manipulation d'enum et méthodologie

La fonction `nom_marque()` permet de convertir une enumeration de type `marque_carte_graphique` (paramètre d'entrée de la fonction) vers une chaine de caractères indiquant en toute lettre la marque correspondante (retour de la fonction).

Question 9 *Donnez la signature (=en-tête) de la fonction* `nom_marque()`

Question 10 *Donnez le contrat que doit suivre la fonction* `nom_marque()`

Question 11 *Donnez l'implémentation de la fonction* `nom_marque()` *qui suit le contrat donné.*

2.3 Adressage de variables et méthodologie

La fonction `ordinateur_le_plus_rapide` prend en paramètre deux adresses pointant vers des `struct ordinateur` et renvoie un pointeur vers l'ordinateur le plus rapide.
Dans le cas présent, on calculera la *rapidité* d'un ordinateur en considérant la formule:

$$\text{rapidité} = N_c \times f_c + 0.2\ N_{gpu} \times f_{gpu}\ ,$$

avec N_c: le nombre de coeurs du processeur central, f_c sa fréquence, N_{gpu} le nombre de coeurs de calcul des cartes graphiques et f_{gpu} la fréquence de la carte graphique.

Question 12 *Dans quel fichier serait-il raisonnable de déclarer l'en tête de cette fonction? Dans quel fichier serait-il raisonnable d'implémenter le corps de cette fonction? (Plusieurs réponses possibles)*

Question 13 *Donnez l'en-tête de la fonction* `ordinateur_le_plus_rapide`*. Quels variables doivent être qualifiés de* `const` *?*

Question 14 *Ecrivez l'***algorithme** *correspondant à la fonction* `ordinateur_le_plus_rapide`*.*

Question 15 *Ecrivez l'***implémentation** *de cette fonction suivant l'algorithme décrit.*

2.4 Processus de compilation

Question 16 *Dans le cas présent, avec quel type d'ordinateur est initialisé la variable* `pc_a_moi`*.*

Question 17 *Quelle(s) ligne(s) de commande(s) doit-on taper pour compiler le programme et générer un executable du nom de* `pgm` *?*