Université P. & M. Curie
Licence Mention Informatique – L3
Programmation Concurrente (LI330)

Examen partiel
3 Novembre 2010
Durée : 2 heures

Les téléphones portables doivent être éteints et rangés dans vos sacs.
Le mémento Ada/C est le seul document autorisé pendant l'épreuve.

Le barème (sur 20) est donné à titre indicatif.
Il vous est conseillé de soigner votre copie et de rédiger des réponses claires (en particulier les programmes).
Toutes vos réponses doivent être dûment justifiées.
Lisez attentivement le sujet. Toute réponse hors sujet sera considérée comme fausse.

Exercice 1 – Problème : le "fast food"

On se propose de développer un simulateur du comportement de clients dans un "fast-food" simplifié. Les acteurs identifiés dans le système sont les suivants (voir figure 1) :

- `Max_Cuisiniers` cuisiniers,
- `Max_Caissiers` Caissiers,
- `Max_Clients` Clients,

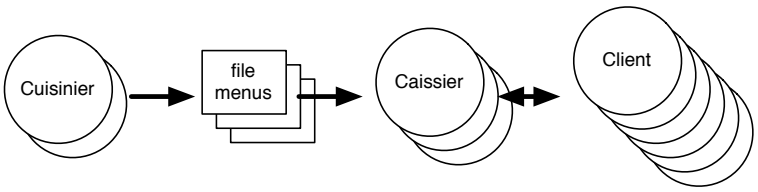

FIGURE 1 – Architecture du simulateur de fast-food

Les caissiers communiquent directement avec les clients pour gérer les commandes et prennent les menus préparés dans une file alimentée par les cuisiniers. Il y a `Max_Menus` types de menus et chaque menu préparé est classé sur une file dédiée. Par soucis de simplicité, on considère qu'un client ne commande qu'un seul menu à la fois.

Cela nous permet de déclarer les entités suivantes :

```
--=========================================================================
-- Constantes et types de base
Max_Menus      : constant Positive := 4;
Max_Cuisiniers : constant Positive := 2;
Max_Caissiers  : constant Positive := 5;
Max_Clients    : constant Positive := 15;
Max_File       : constant Positive := 5;

type Id_Menus is range 1 .. Max_Menus;
type Id_Cuisiniers is range 1 .. Max_Cuisiniers;
type Id_Caissier is range 1 .. Max_Caissiers;
type Id_Clients is range 1 .. Max_Clients;

-- Fonctions de service
function Choisir_Menu return Id_Menus; -- fonction de choix aléatoire d'un menu
function Choisir_Caissier return Id_Caissier; -- fonction de choix aléatoire d'un caissier

--=========================================================================
-- Un cuisinier
task type Cuisinier is
   -- Initialisation
   entry Init (Me : Id_Cuisiniers);
end Cuisinier;

--=========================================================================
-- Un caissier
task type Caissier is
   -- Initialisation
```

```
29        entry Init (Me : Id_Caissier);
30        -- D'autres entrées si nécessaire
31     end Caissier;
32
33     --=========================================================================
34     -- Un client
35     task type Client is
36        -- Initialisation
37        entry Init (Me : Id_Clients);
38        -- D'autres entrées si nécessaire
39     end Client;
40
41     --=========================================================================
42     -- Une file de menus
43     protected type File_Un_Menu (Taille_Max : Positive) is
44        -- Déposer un nouveau menu préparé
45        ??? Deposer;
46        -- Retirer un menu
47        ??? Retirer;
48     private
49        Nb_Menus : Natural := 0;
50     end File_Un_Menu;
51
52     --=========================================================================
53     -- Les variables globales
54     Les_Cuisiniers : array (Id_Cuisiniers) of Cuisinier;
55     Les_Caissiers  : array (Id_Caissier) of Caissier;
56     Les_Clients    : array (Id_Clients) of Client;
57     File_Menus     : array (Id_Menus) of File_Un_Menu (Max_File);
```

Question 1 – 1 point

Combien de tâches identifiez-vous dans le système ? Expliquez et détaillez votre calcul.

Le programme principal a pour objectif d'affecter une identité aux différentes tâches du système. L'identité associée à chaque cuisinier, caissier ou client correspond à son indice dans le tableau correspondant.

Question 2 – 1,5 points

Écrivez le source du programme principal.

Intéressons-nous dans un premier temps aux `File_Un_Menu`. On peut déposer un menu confectionné s'il reste des places dans la file et on ne peut en retirer que s'il y a au moins un menu de prêt.

Question 3 – 1 point

Explicitez en justifiant votre réponse à quelle catégorie de mécanisme `File_Un_Menu.Deposer` et `File_Un_Menu.Retirer` appartiennent.

Question 4 – 0,5 point

Que peut-il se passer si un Caissier tente de retirer un menu au moment où un cuisiner est en train d'en ajouter un dans la file ?

Question 5 – 1,5 points

Écrivez le source du corps du type protégé `File_Un_Menu`.

Considérons maintenant les cuisiniers. Chacun d'eux doit respecter le comportement suivant :

1. recevoir ses paramètres d'initialisation,
2. démarrer une boucle infinie dans laquelle :
   - choisir un menu a réaliser (on considèrera un choix aléatoire),
   - confectionner le menu (affichage d'une trace) et le déposer dans la file des menus prêts.

Question 6 – 1,5 points

Écrivez le source d'une tâche `Cuisinier`.

Dans un deuxième temps, nous traitons le cas d'un `Client` qui doit se comporter comme indiqué ci-dessous :

1. recevoir ses paramètres d'initialisation,
2. choisir aléatoirement un menu et un caissier,
3. Commander le menu choisi au caissier sélectionné (la commande est caractérisée par l'identité du client concerné et le type de menu choisi),
4. Attendre que le caissier notifie la réception de la commande et le payer.

Intéressons-nous enfin au comportement d'un `Caissier` qui doit respecter le protocole suivant :

1. recevoir ses paramètres d'initialisation,
2. démarrer une boucle dans laquelle il attend une commande qu'il doit traiter comme suit :
   – retirer le menu demandé de la file correspondante,
   – réveiller le client qui l'a contacté pour le lui donner et recevoir son paiement
   on sort de la boucle lorsqu'il n'y a plus de clients capables de contacter un caissier.

Question 7 – 1 point

Dans quelle(s) tâche(s) doivent se trouver les points d'entrées correspondant aux interactions entre un `Client` et un `Caissier` ? Justifiez votre choix. Vous donnerez les nouvelles spécifications pour la (les) tâche(s) modifiée(s).

Question 8 – 2 points

Écrivez le source d'une tâche `Client`.

Question 9 – 3,5 points

Écrivez le source du type de tâches `Caissier`.

Nous positionnons les constantes du programme aux valeurs suivantes :

```
Max_Menus      : constant Positive := 1;
Max_Cuisiniers : constant Positive := 1;
Max_Caissiers  : constant Positive := 1;
Max_Clients    : constant Positive := 1;
Max_File       : constant Positive := 2;
```

Question 10 – 3,5 points

Donnez, sous forme d'un chronogramme, l'exécution complète de la première commande traitée par ce système (du démarrage du système de tâches à la terminaison du premier client). Vous considèrerez qu'une commutation a lieu si et seulement si la tâche élue exécute une action bloquante.

Question 11 – 1 point

On rajoute un nouveau type de menu (la constante `Max_Menus` vaut `2`). Expliquez sur l'exemple construit dans la question précédente comment le système peut se trouver en interblocage.

Exercice 2 – Questions de cours

Question 1 – 1 point

Donnez une définition simple de la famine.

Question 2 – 1 point

Expliquez les différences entre un tableau contraint et un tableau non contraint.