


Institut Supérieur d'Informatique et de Mathématiques 	Année Universitaire : 2021-2022
	Examen (Session Principale) Matière : Approche par composants et concurrence Filière : MR2-GL Enseignants : Kmimech Mourad & Sakka Rouis Taoufik Nb pages = 2

Exercice 1 : (4 Points)

Soit la procedure Ada suivante :

```

with Text_IO ; use Text_IO ;
procedure Tache_type is
  task type Un_Type ;
  task body Un_Type is
    begin
      loop
        Put_Line ( " tache activee " ) ;
        delay 1 .0 ;
        -- l'instruction delay expr bloque la tache pendant au moins expr unités de temps
      end loop ;
    end Un_Type ;

    T1 , T2 : Un_Type ;
    T : array ( 1 .. 10 ) of Un_Type ;
begin
  null ;
end Tache_type ;

```

- Combien de tâches ce programme contient il ?
- Quand les tâches commencent elles leur exécution ?
- Quand la procédure principale se termine-t-elle ?

Exercice 2 : (8 Points)

Soit le type générique suivant :

generic

type Element is private; -- type pour lequel "=" et ":=" sont definis

- 1) Proposer une implémentation d'une fonction générique qui permet de calculer, pour un élément x et un entier n , la somme de la série suivante : $S = \sum_{i=1}^n x^i$
- 2) Proposer une instance de cette fonction pour les valeurs de type entier.
- 3) Tester cette instance.

Exercice 3 : (8 Points)

La tâche lecteurs_redacteurs donnée ci-dessous est censée **synchroniser** deux catégories des processus : lecteurs et rédacteurs.

La politique d'acceptation des rendez-vous proposée par cette tâche **favorise** les rédacteurs.

- 1) Énumérer les facteurs qui permettent de constater cette **favorisation**.
- 2) Comment peut-on corriger cette favorisation (Récrire seulement la partie body).

```
task lecteurs_redacteurs is
    entry deb_lect; --demande d'autorisation de lecture
    entry deb_redac; --demande d'autorisation d'écriture
    entry fin_lect; --signal de fin de lecture
    entry fin_redac; --signal de fin d'écriture
end lecteurs_redacteurs;

task body lecteurs_redacteurs is
    nb_lecteurs: integer:=0;
begin
    accept deb_redac;
    accept fin_redac ;
    loop
        select
            when deb_redac'count=0 => accept deb_lect;
                                   nb_lecteurs:=nb_lecteurs+1;
        or
            accept fin_lect;
            nb_lecteurs:=nb_lecteurs-1;
        or
            accept deb_redac do
                while nb_lecteurs> 0 loop
                    accept fin_lect;
                    nb_lecteurs:=nb_lecteurs-1;
                end loop
            end deb_redac;
            accept fin_redac;
        or
            terminate;
        end select;
    end loop;
end lecteurs_redacteurs;
```

NB. La notation **deb_redac'count** donne la cardinalité (taille) de la FIFO associée à l'entrée deb_redac.