

## CPU ( Intra 99 )

- ❑ Un ordinateur parallèle possède 4 processeurs, chacun pouvant exécuter une seule tâche à la fois. Le système d'exploitation cependant est un peu vétuste (pour ne pas dire plus). Les tâches sont soumises en traitement par lot selon une classe et sont exécutées par un processeur d'une seule traite (pas de méthode du tourniquet [*round-robin*]) Chaque tâche possède son propre temps d'exécution. Il y a trois classes de priorité, les tâches de classe 1 étant de plus haute priorité, et les tâches de classe 3 de plus basse priorité.
- ❑ Les tâches sont emmagasinées dans une ou plusieurs files d'attente (dépend du désign) en fonction de la priorité. Le temps de service (exécution) pour une tâche de classe  $i$  est distribué selon une distribution exponentielle de moyenne  $s(i)$ . Chaque classe possède son processus d'arrivée selon une distribution de Poisson avec un taux moyen par minute  $r(i)$ .



## CPU ( Intra 99 )

Classe $i$	Moyenne service $s(i)$ (minutes)	Taux moyen $r(i)$
1	0.15	5
2	1.5	0.62
3	4.9	0.21

- ❑ Le système d'exploitation est tel qu'une tâche en arrivée est assignée immédiatement au premier processeur. S'il y a un processeur libre, il est choisi immédiatement, sinon on choisit aléatoirement selon une fonction **Choix.F** qui vous est fournie gratuitement par la maison (vous n'avez pas besoin de la coder) qui retourne une valeur dans l'intervalle  $[1, 4]$  selon un algorithme très complexe. Ce choix est maintenu jusqu'il arrive.



## CPU ( Intra 99 )

- Le système d'exploitation est tel qu'une tâche en arrivée est assignée immédiatement au premier processeur. S'il y a un processeur libre, il est choisi immédiatement, sinon on choisit aléatoirement selon une fonction **Choix.F** qui vous est fournie gratuitement par la maison (vous n'avez pas besoin de la coder) qui retourne une valeur dans l'intervalle  $[1, 4]$  selon un algorithme très complexe. Ce choix est maintenu quoiqu'il arrive.
- Les tâches de plus haute priorité réquisitionnent le processeur avant les tâches de priorité inférieure.
- Tous les paramètres (variables) pour lesquels des valeurs ne sont pas spécifiées dans cette description sont considérés comme des valeurs obtenues d'un fichier (SIMU12).



## CPU ( Intra 99 )

- Initialement, le système est vide. La simulation doit durer exactement 720 minutes. Écrire un modèle SIMSCRIPT qui
  - pour chaque classe, calcule les temps d'attente minimum, maximum, et moyen,
  - calcule le pourcentage d'utilisation des UCT`s (CPU`s) et le temps moyen d'exécution des tâches par classe de tâche.
- Utilisez des suites différentes pour les arrivées et les temps d'exécution.
- N.B. Dans la solution de ce problème, on demande de mettre l'accent sur le désing du modèle informatique, i.e. identifiez en premier les processus, les ressources, les entités, les statistiques, etc... Les déclarations de variables ne sont pas les parties les plus importantes ainsi que les entrées/sorties pour les paramètres et les résultats.



## Preamble

Preamble

Last Column is 80

Normally Mode is undefined...

Permanent Entities .....

Every Classe has a Moyenne.Service,  
a Taux.Moyen,  
a Inter.Arrivee,  
a Temps.Service,  
a Temps.Attente

Define Moyenne.Service,  
Taux.Moyen,  
Inter.Arrivee,  
Temps.Service,  
Temps.Attente as Real Variables

Resources .....

Every Cpu has a Numero

Define Numero as an Integer Variable



## Preamble

Processes .....

Every Generateur.Arrivee has a No.Classe

Define No.Classe as an integer variable

Every Tache has a Tk.Classe,  
a Tk.Priorite

Define Tk.Classe, Tk.Priorite as integer variable

Event include Fin.Simulation

" Déclaration des fonctions

Define Choix.F as an Integer Function with 0 arguments

" Declaration des variables statistiques

Tally moy.Attente as the mean,  
min.Attente as the minimum,  
max.Attente as the maximum of Temps.Attente

Tally moy.Temps.Execution as the mean of Temps.Service

Accumulate Pour.Utilisation as the mean of N.X.Cpu

Define Forever to mean While ( 1 = 1 )

end Preamble



## CPU ( Intra 99 )

```
Main  
  
  Call Initialiser  
  for each Classe,  
  do  
    Activate a Generateur.Arrivee ( Classe ) now  
  loop  
  
  Schedule a Fin.Simulation in 720 Minutes  
  
  Start Simulation  
  
end " Main
```



## Process Generateur.Arrivee

```
Process Generateur.Arrivee ( No.Classe )  
  Define No.Classe as an integer variable  
  
  Forever  
  Do  
    Wait Exponential.F ( InterArrivee(No.Classe), 5 ) Minutes  
    Activate a Tache ( No.Classe, N.Classe - No.Classe) Now  
  Loop  
  
end "Generateur.Arrivee
```



## Event Fin.Simulation

```

Event Fin.Simulation
  print 1 line thus
    Moyenne   Minimum   Maximum   Ex. moyenne
  for each Classe,
  do
    print 1 line with moy.Attente * Hours.v * Minutes.V,
      min.Attente * Hours.v * Minutes.V,
      max.Attente * Hours.v * Minutes.V,
      moy.Temps.Execution * Hours.v * Minutes.V

    thus
      **** **   **** **   **** **   **** **
  loop

  for each cpu
  do
    print 1 line with cpu, Pour.Utilisation(cpu) thus
    * **** **
  loop
  read as /
  Stop
end "Fin.Simulation
  
```



## Routine Initialiser

```

Routine Initialiser
  print 1 line thus
  Donnez le nombre de classes
  Read N.Classe
  Create AllClasse
  For Each Classe,
  Do
    Print 1 line with Classe thus
    Classe *** Donnez la Moyenne et Taux moyen
    Read Moyenne.Service(Classe), Taux.Moyen(Classe)
    Inter.Arrivee (Classe) = 1.0 / Taux.Moyen(Classe)
  Loop

  print 1 line thus
  Donnez le nombre de CPU
  Read N.Cpu

  Create all Cpu
  for each Cpu,
  do
    let U.Cpu(Cpu) = 1.0
    let Numero ( Cpu ) = Cpu
  loop
end " Initialiser
  
```



## CPU ( Intra 99 )

```
Process Tache ( No.Classe, Priorite )
Define No.Classe, Priorite as integer variable
Define Temps.Arrivee as a real variable

Define Le.Cpu as an integer variable

for each cpu with u.Cpu (Cpu) > 0
  find the first case
  if found,
    Le.Cpu = Cpu
  else
    Le.Cpu = Choix.F
  endif

Temps.Arrivee = Time.V
request 1 Cpu(Le.Cpu) with priority Priorite (Le.Cpu)
  Temps.Attente(No.Classe) = Time.V - Temps.Arrivee
  Temps.Arrivee = Time.V
  Work Exponential.F ( Moyenne.Service (No.Classe), No.Classe) Minutes
  Temps.Service ( No.Classe ) = Time.V - Temps.Arrivee
relinquish 1 Cpu(Le.Cpu)
end " Tache
```



## Function Choix.F

```
Function Choix.F
  Return with intF(Uniform.F ( 1.0, N.Cpu, 10))
End " Choix.F
```

