Cahier des charges : projet stage

Création d'une application d'exercices de systèmes et codage de l'information

Client : GENIET Annie

Développeurs : HUBERT Luc, M'DAHOMA Dhouriati, OUAZZANICHAHDI Yousra, PUISAIS Adeline, REHIOUI Mounia, SARLANDE Baptiste

Année : 2021 / 2022

Contexte

La troisième année de licence Génie Bio-informatique à l'université de Poitiers comporte plusieurs aspects d'apprentissage de l'informatique :

- une initiation à l'informatique où sont vus les principes du codage de l'information.
- Un enseignement comportant les principes généraux des systèmes d'exploitation.

Tous ces enseignements sont vus durant les cours magistraux et mis en pratique à travers des exercices proposés lors des séances de Travaux Dirigés ou dans les Devoir maison. Ils sont nécessairement en nombre limité. Pour certains, ils sont également mis en pratique durant des Travaux Pratique.
La cliente, Mme Annie Geniet, en charge de ces enseignements, souhaite proposer aux étudiants une application leur donnant accès à un plus large panel d'exercices, afin de leur permettre de s'entraîner autant que nécessaire, tout cela ayant pour but d'acquérir les notions vues lors des cours.

Objectifs généraux :

L'application à développer comportera 4 grandes parties, chacune associée à l'un des 4 thèmes suivants :

1. Codage de l'information
2. Ordonnancement
3. Gestion de la mémoire
4. Gestion de fichiers

Chaque thème est ensuite divisé en plusieurs exercices, qui seront décrits dans les parties dédiées dans ce document. Tous les exercices suivront la même trame :

a. Les données pourront être, au choix de l'utilisateur, soit saisies soit générées aléatoirement par l'application. Dans la suite du document, le terme "entrée" représente l'une ou l'autre des solutions.

b. L'utilisateur devra saisir ses réponses dans les espaces dédiés à cet effet.

c. Si la/les réponse(s) fournie(s) est/sont correcte(s), un message apparaîtra. Dans le cas contraire, l'utilisateur pourra proposer une/des nouvelle(s) réponse(s). Il aura le droit à 3 essais au bout desquels la bonne réponse lui sera donnée.

d. Un contrôle syntaxique est mis en place, qui vérifie si les données saisies sont correctes. Un message sera affiché si la saisie n'est pas du bon type ou bien si les données ne respectent pas les contraintes. Dans le cas où la saisie est syntaxiquement correcte alors elle pourra (et seulement dans ce cas), être validée.

e. Les valeurs pourront être fournies entrecoupées d'autant d'espace que voulu. Elles doivent donc être composées uniquement des symboles autorisés pour la base choisie et d'espaces. Sinon, elles sont considérées comme syntaxiquement incorrectes.

f. Lorsque cela est pertinent, des rappels méthodologiques seront affichés au début de l'exercice.

A la fin de l'exercice, la liste des exercices du thème est reproposée jusqu'à ce que l'utilisateur décide de revenir à l'écran où il peut choisir les thèmes.

Les parties suivantes détaillent le contenu de chacune des parties.

I. La partie Codage de l'information :

Dans l'interface principale de l'application, Si on choisit le thème Codage de l'information un nouvel écran s'affiche proposant une liste d'exercices disponibles. L'utilisateur choisit alors l'exercice qu'il veut faire.

Huit exercices sont proposés :

1. Les entiers non signés
2. Les opérations en binaire
3. Les multiplicités en binaire
4. Opérations sans calcul
5. Les entiers signés
6. Les décimaux
7. Les réels
8. Les tableaux

1) Entiers non signés

Objectif : Faire des conversions entre différentes bases Les

bases considérées sont les bases : 10-2-8-16

Données entrées : base de départ - base d'arrivée
Si les bases sont saisies, l'entier à convertir est entré
Si les bases sont générées aléatoirement, la valeur de l'entier à convertir est également générée.

Réponse attendue : le nombre codé dans la base d'arrivée.

Contraintes :

- Les entiers en base 10 sont compris entre 0 et 10 000
- Si on travaille avec les bases 2, 8 ou 16 (sans aller de ou vers la base 10), les contraintes de taille lors des entrées sont :
    - Base 2 : maximum 32 bits
    - Base 8 : maximum 10 symboles
    - Base 16 : maximum 8 symboles

Rappels méthodologiques :

- pour aller de base 10 à la base 2 : méthode de la division ou méthode de la soustraction
- pour aller de base 2 à base 8 ou 16 : méthode de la compression
- pour aller de base 8 ou 16 à base 2 : méthode de l'expansion
- pour aller de base 8 à base 16 ou réciproquement : passer par la base 2 en intermédiaire.

2) <u>Opérations en binaire</u>

Objectif : Effectuer des opérations en base 2.
Les opérations considérées sont les multiplications, les additions et les soustractions.

Données entrées : choix de l'opération - valeur des opérandes.
Si le choix de l'opération est saisi, la valeur des opérandes est entrée.
Si le choix de l'opération est aléatoire, la valeur des opérandes est également générée.

Réponse attendue : le nombre en base de 2 résultant de l'opération.

Contraintes :

- Les entrées ne pourront pas excéder 16 bits.

Rappel méthodologique :

- N'oubliez pas que : 1 + 1 = 10.
- N'oubliez pas les retenues.

3) <u>Multiplicité</u>

Objectif : savoir si un nombre binaire est multiple d'une puissance de 2.

Données générées : Entiers codés en binaire - un entier qui est un puissance de 2.
10 valeurs sont générées aléatoirement, ainsi que la puissance de 2.

Réponse attendue : Les valeurs qui sont des multiples de l'entier doivent être cochées.

Contraintes :

- Les entrées ne pourront pas excéder 32 bits.
- Au moins 3 multiples de la puissance de 2 donnée se trouvent parmis les 10 valeurs générées.

Rappel méthodologique :

- Si un entier est multiple d'une puissance (n) de 2, il doit se terminer par (n) 0.

4) Opérations sans calcul

Objectif : Multiplier ou diviser un nombre en base 2 par une puissance de 2.

Données entrées : Type d'opération - Nombre en base 2 - Puissance de 2. Lorsque le type d'opération est généré, les valeurs du nombre en base 2 et de l'entier en base 10 sont générées.

Réponse attendue : le nombre en base 2 résultant de l'opération.

Contraintes :

- Les base 2 ne peuvent pas excéder 16 bits en entrée
- Les entiers en base 10 sont compris entre 2 et 256 uniquement en puissance de 2
- Le nombre en base 2 doit se terminer par au moins autant de 0 que l'exposant de la puissance de 2 dans le cas de la division

Rappel méthodologique :

- Pour multiplier il faut ajouter des 0 à la fin du nombre en base 2 donné
- Pour diviser il faut supprimer des 0 à la fin du nombre en base 2 donné.

5) Les entiers signés

a) Les signes

Objectif : déterminer le signe d'un entier codé en SVA/C2.

Données générées : un entier codé soit en SVA, soit en C2.
La valeur est générée aléatoirement ainsi que le format (SVA ou C2).

Réponse attendue : Le signe de l'entier donné. Il doit être coché (positif/négatif).

Contraintes :

- L'entier ne peut pas excéder 16 bits.
- Dans cet exercice, une seule tentative sera accordée à l'utilisateur.

b) <u>Conversion :</u>

Objectif : Faire des conversions entre différents formats. Les formats considérés sont la base 10,SVA et C2.

Données entrées : format de départ - format d'arrivée
Si le format de départ est saisi, le format d'arrivée est saisi.
Si le format de départ est généré aléatoirement, le format d'arrivée est aussi généré.
Si on va vers SVA ou C2, on précise le nombre de bits sur lequel on veut faire le codage.

Réponse attendue : le nombre codé dans le format d'arrivée.

Contraintes :

- L'entier ne peut pas excéder 16 bits.
- Si on donne un entier en base 10, et le nombre de bits, alors la valeur absolue doit être comprise entre 1-2^(n-1) et 2^(n-1)-1.

Rappels Méthodologiques :

- Si l'entier codé en SVA/C2 commence par 1 : son signe est négatif.
- Si l'entier codé en SVA/C2 commence par 0 : son signe est positif.
- Pour convertir un nombre négatif de C2 vers SVA : on soustrait 1, puis on inverse les bits en faisant attention au signe.
- Pour coder un entier négatif en C2 : on part de la valeur absolue, on inverse les bits, puis on ajoute 1.

6) <u>Décimaux</u>

Objectif : Convertir un nombre décimal de la base 10 vers la base 2 ou inversement.

Données entrées : base de départ - base d'arrivée - nombre de bit après la virgule pour la base 2.
Si l'ordre de conversion est choisi, la valeur dans la base de départ est entrée.

Si l'ordre est choisi aléatoirement, la valeur est également générée aléatoirement.

Réponse attendue : Le nombre codé dans le format d'arrivée.

Contraintes :

- Si la valeur de départ est en base 2 elle ne peut pas dépasser 32 bits.
- Si la valeur est en base 10 elle est comprise entre 0 et 10 000.
- Si la valeur est en base 2, il y a au plus 5 bits après la virgule.
- Si on convertit en base 2, le nombre de bits après la virgule est compris entre 1 et 6.

Rappel méthodologique :

- Pour obtenir la partie décimale, on utilise la méthode de la multiplication.

7) <u>Les réels</u>

Objectif : Convertir un réel au format IEEE 32 bits donné en hexadécimal(ou l'inverse)
Bases considérées : 10 - 16

Données entrées : le sens de la conversion - le réel à convertir.
Le réel à convertir est, soit en format décimal (si c'est la base 10), soit en hexadécimal (si c'est en IEEE).

Réponse attendue : Le nombre codé dans le format d'arrivée.

Contraintes :

- Les valeurs en base 10 sont compris entre -10 000 et 10 000

Rappels méthodologiques :

- 1 bit de signe
- 8 bits d'exposant biaisé (biaisé de 127)
- 23 bits de mantisse.
- Ne pas oublier le bit implicite.

8) <u>Tableau</u>

Objectif : Calculer le nombre de mots d'un tableau et savoir les placer.

Données entrées : taille du tableau (en nombre de case) - taille d'une case (en nombre de mot) - taille d'un mot (en bits ou en octets) - adresse du premier mot du tableau.

Les valeurs sont entrés (saisies ou générées, les deux sont envisageables). Dans un second temps, le numéro de case est généré pour la deuxième question.

Réponse attendue :

- le nombre de mots contenus dans le tableau et les numéros du premier et du dernier mot du tableau.
- les mots contenus dans la case dont le numéro est généré.

Contraintes :

- La taille du tableau est comprise entre 30 et 300 cases.
- Taille mot * taille case ne doit pas dépasser 950 mots

Validé jusqu'ici
par le client Annie Geni

Retappel méthodologique :

- Si a est le premier mot du tableau, et si une case contient n mots, le premier mot de la case n°k est le mot a+(k-1)*n.

II. La partie ordonnancement

Objectif : Simuler l'exécution d'une famille de processus selon différents algorithmes

d'ordonnancement

Interface:

Si ont choisi le bouton\thème "Ordonnancement" un nouvel écran s'affiche ,un exercice s'ouvre avec un menu déroulant pour choisir quelle version de l'exercice l'utilisateur souhaite.

Huit version/ exercice:

1. Tourniquet

2.FIFO

3.PCTER

4. Priorités fixes

5. Algorithmes multi files à trois files (4 versions)

   a) Avec ou sans migration " pas encore défini

   b) Les deux premières files ordonnancées par des tourniquets de quantum q1 et q2 et:
      i. la troisième file ordonnancée par un tourniquet de quantum q3 (Version 1)
      ii. la troisième file ordonnancée par FIFO (Version 2)

Dans le menu déroulant les exercices seront affichés comme ceci:

-Tourniquet

-FIFO

-PCTER

-Priorités fixes

-Algorithmes multi files FIFO sans migration

-Algorithmes multi files FIFO avec migration

-Algorithmes multi files TOURNIQUET sans migration

-Algorithmes multi files TOURNIQUET avec migration

Pour chaque exercice les objectifs , données d'entrées et les réponses attendues sont identiques :

Données : la première donnée entrée est l'algorithme d'ordonnancement. Ensuite le

système de processus est entré. Les informations entrées sont :

- le nombre de processus

- la date d'arrivée

- la durée

- le cas échéant, la priorité

- le quantum (uniquement tourniquet) (contraintes : entier bornés)

-*Objectifs :*

- Construire un diagramme de Gantt (l'utilisateur rentrera le processus correspond au temps 0 puis temps 1 etc dans des petites cases , pour cela le développeur doit calculer la durée totale et affichée le nombre de case en conséquence)

- Calculer le temps de réponse (dénominateur sur numérateur en prenant compte du quantum sur la méthode sélectionnée

- *Données entrées :* le système de processus avec les dates d'arrivée et la durée de chaque processus , et les priorités lorsque cela est nécessaire.

-*Réponse attendue :* L'ordre de passage des processus \ diagramme de Gantt

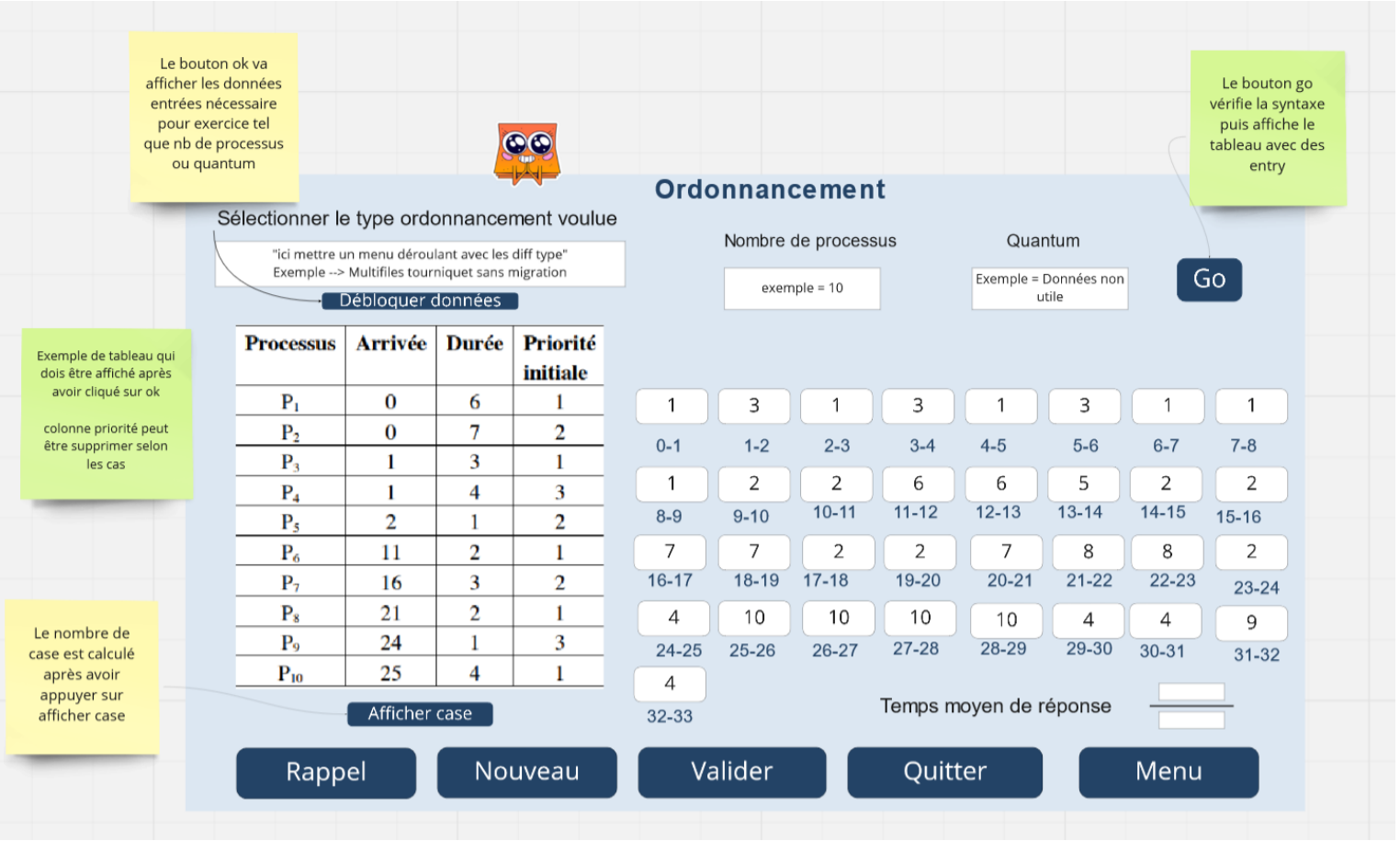
-Contraintes:

- Nombres de processus compris entre 4-10
- Durée d'un processus entre 1-10
- Quantum max 4

-Contraintes aléatoire

- Les dates d'arrivées des processus doivent être en ordre croissant
- Les dates d'arrivées des processus doivent être inférieures ou égales à la somme des durée de processus déjà présents.

l'exemple d'interface ci dessous est l'unique interface pour les différents exercices



Explication fonctionnement de cette interface :

Etat de base Bouton valider et nouveau bloquer

L'utilisateur va choisir le type d'ordonnancement pour cela Un menu déroulant sera utilisé pour déterminer le type d'ordonnancement puis pour valider le choix on appuie sur débloquer données

Les cases Nombre de processus et quantum seront débloqués si nécessaire puis l'utilisateur appuie sur Go ce qui va si la syntaxe est correcte, affiché un tableau à remplir avec le nombre de lignes correspondant au nombre de processus.

Une fois le tableau saisie on appuie sur afficher case qui va donc afficher le nombre de case correspondant à la somme des durées des processus.

Une fois les cases remplies, l'utilisateur appuie sur Valider ce qui contrôle la syntaxe et si les réponses sont justes.

En cas d'erreur :

En cas d'erreur que ce soit dans les données de départ ou dans les réponses, il est important de préciser où se situe précisément l'erreur via les messagesbox et non simplement « erreur »

Pour le processus mettre en rouge la case où il y a une mauvaise réponse permet de se simplifier les choses comme dans l'exercice des tableaux

3 chances de réponse au bout de le troisième bouton valider bloquer et nouveau débloquer

Pour le bouton Nouveau :

Penser à rebloquer la saisie du temps de réponse moyen et débloquer les données de départ et le menu déroulant.

Aléatoire :

Les données sont générés aléatoirement + le tableau , l'utilisateur ne fait que saisir le temps moyen de réponse et les cases avec les processus /temps restant et la période de temps ou il s'exécute comme pour le manuel (voir image)

1) **Tourniquet**

Contraintes:

-Voir les contraintes générales

Rappels méthodologiques :

- Le processus sélectionné est celui en top de liste et chaque nouveau processus est ajouté en fin de liste
- Un processus en cours s'exécute pendant un temps (q) et se met en fin de file s'il n'a pas fini
- L'arrivée d'un processus avec une plus grande priorité entraîne l'arrêt du processus en cours et s'exécute.

2) **FIFO**

Rappels méthodologiques:

- FIFO = First-In, First-Out , le premier processus arrivé est le premier à s'exécuter
- Temps de réponse pour un processus = Date de fin - Date d'arrivée
- Temps moyen de réponse = somme des temps de réponse de chaque processus divisé par nombre de processus

3) **PCTER**

Rappels méthodologiques:

- Le processus sélectionné est celui qui a le plus court temps d'exécution restant

4) **Priorités fixes**

Objectif : Identifier la priorité d'un processus

Rappels méthodologiques :

- Comparaison des processus pour trouver celui avec la plus grande priorité

- Respecter l'ordre de priorité : si une nouvelle tâche à une plus grande priorité, le processus est interrompu
- Lorsque deux processus ont la même priorités, la règle du FIFO s'applique

5) **Algorithmes multi files à trois files**

Rappels méthodologiques pour les 2 versions:

Quand une file est ordonnancée par un quantum n, elle applique son algorithme d'ordonnancement pendant n unités de temps,sauf si elle se retrouve vide avant l'expiration du quantum dans ce cas la file suivante prend le relai

Quand une file est interrompue, elle reprend là où elle en était, donc si un processus n'avait pas fini son quantum, il le termine.

a) Version 1 = Les 2 premières files sont en quantum q1 et q2 , la 3 -ème files ordonnancée par un tourniquet de quantum q3

Rappels méthodologiques:

- *Les questions à se poser*

- est ce que le processus a fini de s'exécuter?

- est ce qu'il y a des processus en attente?

+ **rappel tourniquet écrit précédemment**

b) Version 2= Les 2 premières files sont en quantum q1 et q2 , la 3 -ème files ordonnancée par FIFO

Rappels méthodologiques:

- FIFO = First-In, First-Out , le premier processus arrivé est le premier à s'exécuter