SEG2501 - Conception des logiciels III

Professeur : Stéphane S. Somé

Examen de mi-session : 02 Mars 2002

Nom : _____ Numéro d'étudiant : _____

- Durée : 1h20
- Examen à livres fermés.
- Calculatrices non permises.

Part 1 : ☐ /20 Part 3 : ☐ /20

Part 2 : ☐ /20

Total : ☐ /60

Partie 1/20 points

Encerclez le numéro de la réponse la **PLUS** appropriée dans la liste des réponses données (un choix par question). Deux (2) points par réponse.

Question 1.1 Laquelle de ces réponses est vraie ? La qualité dans le génie logiciel est définie comme

a) L'habilité du système à satisfaire à la fonctionnalité voulue par les usagers et clients.

b) L'habilité du système à satisfaire à la performance voulue par les usagers et clients.

c) L'habilité à satisfaire au coût voulue par les usagers et clients.

d) a), b) et c)

Question 1.2 Laquelle de ce qui suit n'est pas une exigence non fonctionnelle ?

a) Le système doit pouvoir traiter 1000 données en 10 secondes.

b) Le système doit pouvoir calculer la moyenne des notes des étudiants.

c) Le système doit pouvoir retourner dans un état sécure après toute exception.

d) Le système doit être implanté en C++.

Question 1.3 Laquelle de ce qui suit est fausse

a) SDL signifie Specification and Description Language.

b) Un système SDL consiste en blocs (blocks) connectés par des canaux (channels).

c) SOON fait parti de SDL.

d) Un bloc SDL peut être subdivisé en blocs.

Question 1.4 Lequel de ce qui suit est un générateur en SDL ?

a) Integer

b) Character

c) Charstring

d) String

Question 1.5 A propos de l'architecture de *Von Neumann*, laquelle de ces affirmations est fausse ?

a) Les données ainsi que les programmes sont stockés dans la même mémoire.

b) L'unité centrale (CPU) est séparée de la mémoire.

c) Les instructions et données doivent être transmises de la mémoire au CPU.

d) Les ordinateurs construits après 1995 ne sont pas basés sur l'architecture de *Von Neumann*.

Question 1.6 Lequel ce ce qui suit n'est pas une composante d'un compilateur ?

a) Analyseur lexicale

b) Analyseur syntaxique

c) Token

d) Générateurs de code

Question 1.7 Laquelle de ces classes de langages de Chomsky est la plus restrictive ?

a) Langages hors-contextes

b) Langages réguliers

c) Langages récursivement énumérables

d) Langages avec contexte

Question 1.8 Soit la grammaire suivante ayant comme symbole but `<A>`.

```
<A> -> a <B> a | a
<B> -> <B> <C> | b
<C> -> <A> c   | <B>
```

Laquelle des chaînes suivantes est générée par cette grammaire ?

a) bababc

b) abaaac

c) abbbabbbc

d) abbbbba

Question 1.9 Laquelle des chaînes suivantes ne fais pas parti du langage dénoté par l'expression régulière `(ab*c*)*` ?

a) a

b) b

c) ac

d) abc

Question 1.10 Laquelle des réponses suivantes est vraie ? Soient `r` et `s` deux expressions régulières, `r|s` dénote le langage

a) $L(r) \cup L(s)$

b) $L(r) \cap L(s)$

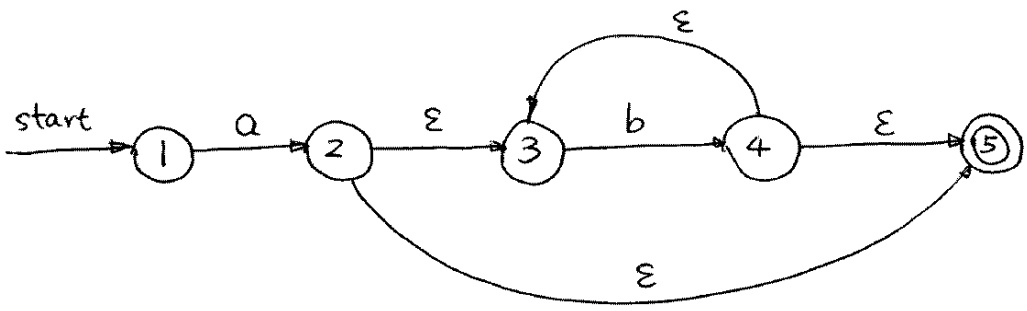
c) $L(r)L(s)$

d) $L(r)^{L(s)}$

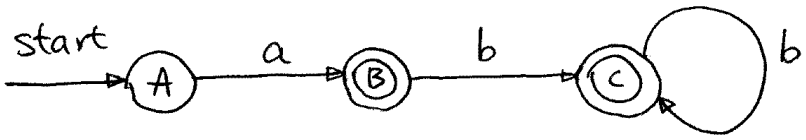
Partie 2/20 points

Question 2.1/4 points

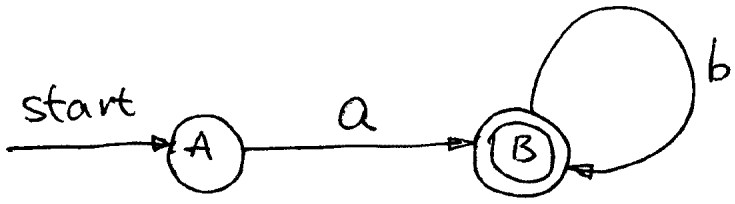
i) Pour chacun des automates à états-finis suivants, dites s'il est déterminisme (DFA) ou non-déterministe (NFA).



Réponse: _____



Réponse: _____



Réponse: _____

ii) Ces trois automates reconnaissent-ils le même langage ? (oui/non) _____

Question 2.2/4 points

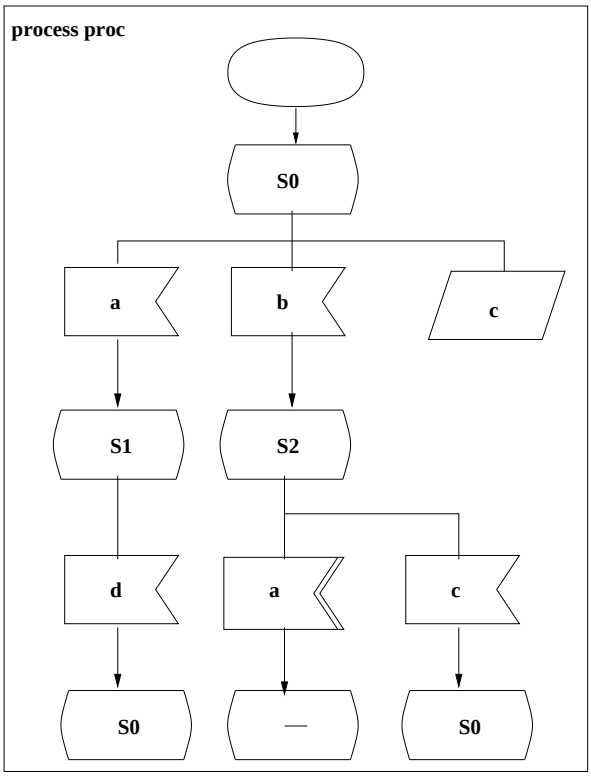
Donnez une définition régulière pour les identificateurs tel que ceux-ci sont des chaînes constituées de lettres de l'alphabet, de digits ou du caractère '_', débutant avec une lettre ou le caractère '_'.

Question 2.3/6 points
Montrez que la grammaire suivante est ambiguë.

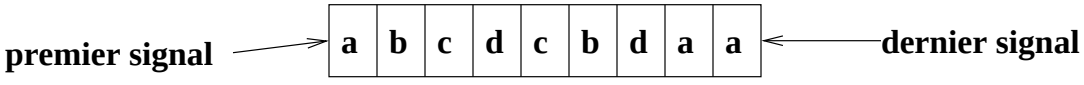
```
<A>  -> <B>
<B>  -> <B> * <B> | <id>
<id> -> a | b | c
```

Question 2.4/6 points

Supposez la définition de processus *proc* suivante.



Supposez que *proc* est dans l'état *S0*, la *file d'attente de messages* de *proc* contient dans l'ordre du premier au dernier signal, les signaux *a*, *b*, *c*, *d*, *c*, *b*, *d*, *a*, *a*.



Remplissez le tableaux suivant en indiquant pour chacun des signaux si ceux-ci sont consommés ou non et quel est l'état suivant dans lequel le processus entre après le traitement du signal.

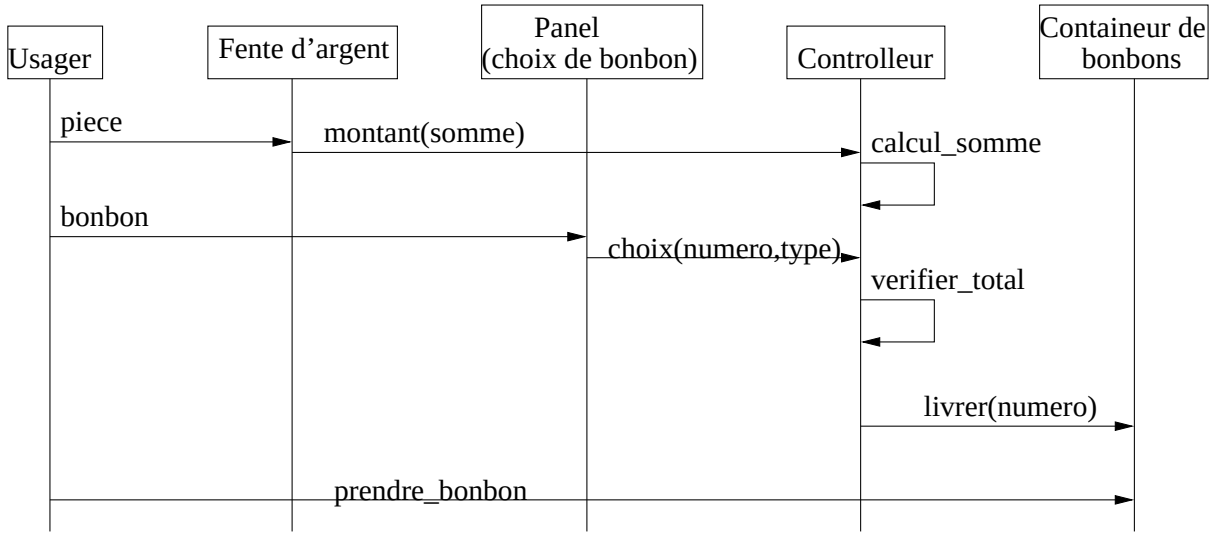
	Signal	Consommé oui ou non	Etat après le traitement du signal
1	a		
2			
3			
4			
5			
6			
7			
8			
9			

Partie 3/20 points

Considérez une machine distributrice de bonbons. Cette machine contient différent types de bonbons. A chaque type de bonbon correspond un numéro et un prix. Le prix est soit $1.00 ou $1.50. La machine comprend:

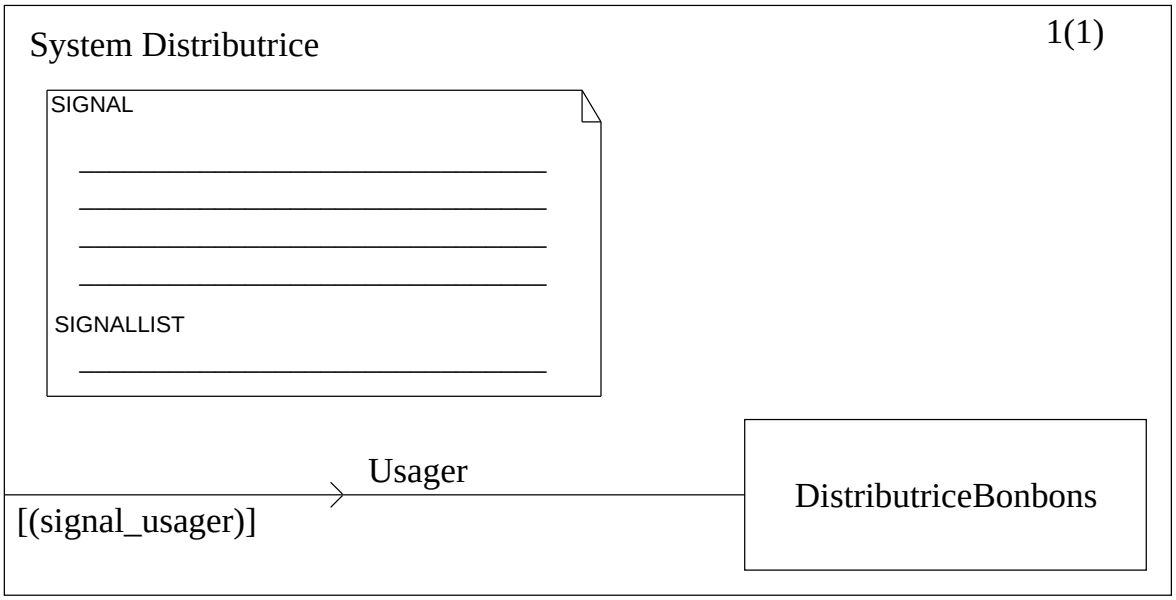
- une fente d'insertion des pièces de monnaie
- un panel avec un bouton par type de bonbon par lesquels les usagers choisissent les bonbons
- un containeur où sont stockés les bonbons
- une zone de récupération de bonbons
- un contrôleur

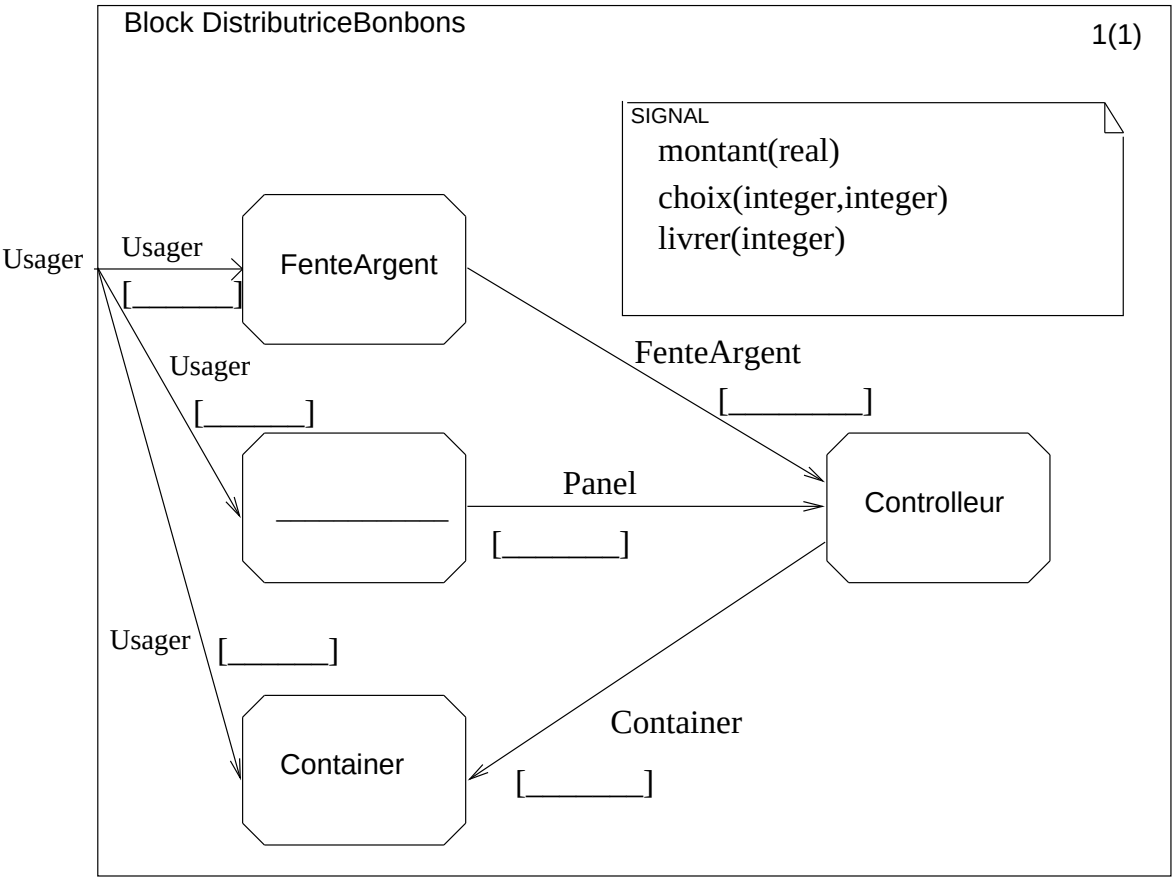
Un usager débute par insérer de la monnaie dans la fente d'argent. L'usager peut insérer une ou plusieurs pièces. Lorsque la somme entrée est suffisante, l'usager presse le bouton correspondant au bonbon voulu. La machine vérifie la somme entrée. Si la somme entrée est correcte, la machine retourne le bonbon choisi dans la zone de récupération de bonbons et l'usager peut alors prendre le bonbon. Si la somme entrée est insuffisante, la machine garde l'argent et l'usager doit recommencer. La machine ne retourne jamais de monnaie.
Le Message Sequences Charts suivants décrit un exemple d'interaction de la machine distributrice de bonbons avec ses usagers. Ce MSC correspond au cas où la somme entrée est correcte.



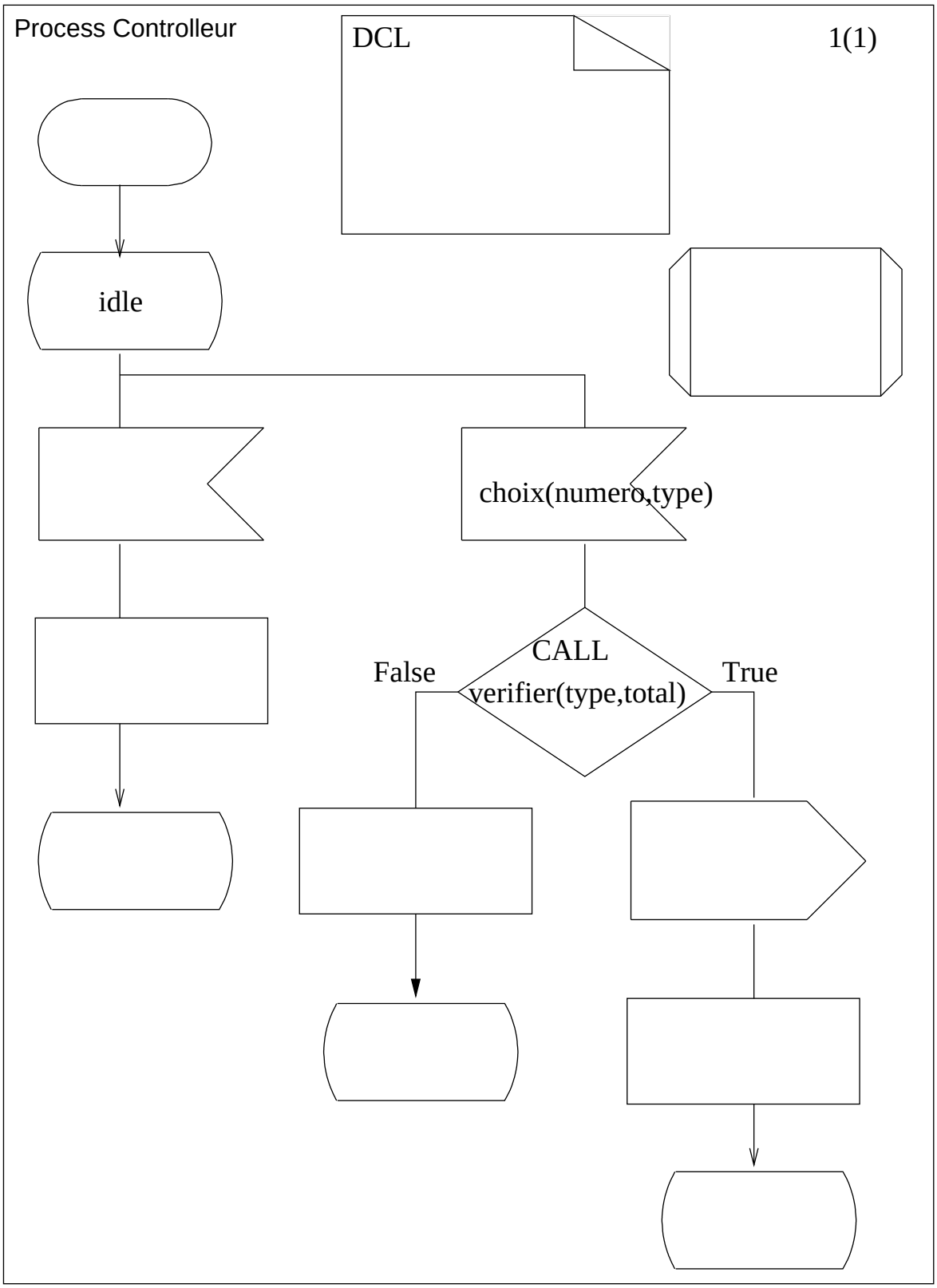
Question 3.1/5 points

Les diagrammes suivants décrivent la structure du système. Complétez l'information manquante de ces diagrammes.





Question 3.2/9 points
Complétez la définition du processus Controller suivante.



Question 3.3/6 points

La procédure verifier prend en paramètre un entier représentant le type de bonbon ainsi qu'un réel représentant le total d'argent entré.
La valeur du paramètre type = 1 pour les bonbons coûtant $1.00 et type = 2 pour les bonbons coûtant $2.00. Verifier doit retourner le booleen true si le total d'argent est supérieur ou égal au prix correspondant au type de bonbon et false sinon.
Ecrivez la procédure verifier en SDL.