encadrés par C. Attiogbé

Cahier d'exercices 1 - Méthode B Nombre de séances : 4 * 1h20 (prévisionnel)

☛ Rédigez impérativement un compte-rendu d'expériences par séance (leçons, temps/exos...)

☞ Consignes particulières données par l'enseignant (remise de copies...)

Revoir votre CM et les machines étudiées (clauses et syntaxe) : jauge, température, ressources, ...

Exercice : une machine à calculer

Spécification en B d'une machine à calculer rudimentaire qui manipule des valeurs entières stockables sur 8 bits (pas de signe). La machine à calculer dispose de deux registres : un registre principal nommé RP et un registre secondaire nommé RS. La machine propose les opérations suivantes (bien sûr on peut en imaginer d'autres).

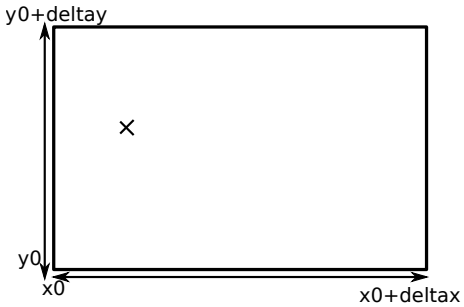
- inc1 : incrémentation de son registre principal de 1.
- dec1 : décrémentation de son registre principal de 1.
- cmp : comparaison de deux valeurs, l'une dans RP l'autre dans RS ; l'opération cmp renvoie un résultat qui est 1 ou 0 (valeurs égales, ou valeurs différentes).
- storeRP(val) : stockage d'une valeur val donnée dans le registre RP. La même opération pourra être définie pour RS.
- getRP : récupération de la valeur stockée dans RP (prévoir une variable de sortie) ; idem pour RS.

Après avoir analysé l'énoncé, écrivez la spécification en B de la machine à calculer ainsi décrite.

Exercice : positionnement d'un pointeur dans le plan

Soit à contrôler l'évolution des valeurs de deux variables xx, yy. Les valeurs sont entières. Elles peuvent représenter les coordonnées (x,y) d'une pointe traçante.

Chacune des deux variables évolue sur une plage δ à partir de sa valeur initiale ; par exemple les valeurs possibles de xx sont entre x_0 et $x_0 + \delta_x$; les valeurs possibles de yy sont entre y_0 et $y_0+\delta_y$. On définira $x_0, \delta_x, y_0+\delta_y$ comme des constantes.



On veut écrire en B une machine abstraite GestPosition qui va proposer ses opérations à un programme de contrôle de la pointe traçante.

La machine abstraite GestPosition fournira les opérations suivantes :

- Changement de la valeur de xx ; on donne une nouvelle valeur nx en paramètre
- Changement de la valeur de yy ; on donne une nouvelle valeur ny en paramètre
- Incrémentation de la valeur de xx de px (un paramètre) ;
- Incrémentation de la valeur de yy de py ;
- Décrémentation de la valeur de xx de px ;
- Décrémentation de la valeur de yy de py ;

— Récupération de la valeur de xx ;
— Récupération de la valeur de yy ;
— Test si une valeur donnée ux est entre les bornes de xx ;
— Test si une valeur donnée uy est entre les bornes de yy ;

On rappelle que la substitution bool(predicat) permet d'obtenir la valeur booléenne d'un prédicat ; par exemple rr := bool(xx < 9) donnera la valuer `true` ou `false` à rr.

Q#1 Spécifiez une machine abstraite B (sans les opérations) pour décrire l'espace d'états correspondant à l'énoncé.

Q#2 Complétez la machine abstraite par la spécification des opérations qui sont décrites.

Exercice : en guise de rappels sur relations/fonctions

```
MACHINE
    Etudiant     /* le nom de la machine (ou modèle formel) */
SETS
   ETUDIANT      /* un ensemble abstrait */
;  GROUPE = {g1, g4, g3, g2, g7, g9, g8, g0, g12} /* un ens. enumere */
;  COURS
DEFINITIONS
   NUMERO == 0..20 /* un ens. par définition et intervalle */
CONSTANTS
   maxG
PROPERTIES
   maxG         : NAT       /* un scalaire */
&  maxG > 1 & maxG < 100 /* on se donne une borne raisonnable, sinon MAXINT */
VARIABLES       /* liste de toutes les variables utilisées */
   etudiants, groupes, cours, coursDe, numeroDe,
   groupeDe, etuSalaries
INVARIANT      /* typage de toutes les variables + propriétés du modèle */
   etudiants   <: ETUDIANT  /* un sous-ensemble, mais c'est mieux si FINI, ci-dessous */
&  etuSalaries : FIN(ETUDIANT)  /* un sous-ensemble fini, à préférer partout donc */
&  groupes     <: GROUPE    /* un sous-ensemble */
&  cours       <: COURS     /* un sous-ensemble */
&  coursDe      : etudiants <-> cours   /* une relation */
&  numeroDe     : etudiants >-> NUMERO  /* une fonction totale injective */
&  groupeDe     : etudiants --> groupes  /* une fonction totale */
&  etuSalaries <: groupeDe~[{g3}]     /* une propriete :
               tous les etudiants salaries sont dans le gr3 */
&  card(groupes) <= maxG   /* on ne doit pas depasser maxG groupes */
INITIALISATION
     etudiants, groupes, cours, coursDe, numeroDe, groupeDe, etuSalaries :=
              {},{g1, g3},{},{},{},{},{}
END
```

1. Assurez-vous d'avoir tout compris dans ce modèle B ; posez toutes les questions dont les réponses ne sont pas dans votre poly de cours.
2. Dessinez des diagrammes de Euler-Venn correspondants aux différentes variables (ensembles, relations, fonctions) ; notez bien que les variables sont différentes les unes des autres ;
3. Vous allez écrire une opération ajoutGroupe(ng) qui ajoute un nouveau groupe `ng` à l'ensemble des groupes. Quelles contraintes doivent être respectées ? Ecrivez et analysez votre machine.

Exercice : raffinement et génération de code (en C)

En fonction de la progression des apprentissages, on poursuit ici avec le raffinement et la génération de code, sur les exercices précédents : gestion de position dans le plan, dictionnaire. En vous basant sur les exemples du cours (et page du cours), faites les raffinements des exercices précédents.

Exercice - spécification d'un mini-dictionnaire en B

On veut écrire un programme correct qui construit et gère un mini-mini-dictionnaire. Notre programme travaille avec un ensemble de couples (mot, signification) nommé dico qui représente le dictionnaire. La signification est ici prise sous une forme très abstraite : un mot n'a qu'une et une seule signification, mais on admet que deux mots peuvent avoir une même signification.

Au début, le dictionnaire est vide. On propose les opérations suivantes à travers notre programme.

— AjoutMot(mm, signif) : cette opération ajoute le mot mm et sa signification (signif) dans dico ;
— RetraitMot(mm) : cette opération enlève le mot mm du dictionnaire ;
— existeMot(mm) : vérifie si le mot mm est dans le dictionnaire (dico) ou non ; l'opération renvoie une valeur booléenne.
— sensDuMot(mm) : recherche et renvoie la signification d'un mot donné en paramètre.

Q#3 Ecrivez une machine abstraite B nommée DicoMot qui spécifie l'état du dictionnaire et modélise les différentes opérations décrites précédemment.
Vous utiliserez pour ce faire un ensemble abstrait MOT qui représente l'ensemble de tous les mots possibles imaginables et un ensemble abstrait SIGNIFIK qui représente (les abstractions de) toutes les significations possibles imaginables.

Exercice - spécification d'un système de gestion d'annuaire

Soient des personnes et des numéros à considérer dans un programme de gestion d'un annuaire sur un téléphone mobile. Nous considérons **une relation entre les personnes membres de l'annuaire et les numéros**.
Faisons l'hypothèse que nous avons d'une part les personnes membres de notre annuaire (notre club) et d'autre part une relation entre les personnes membres (de notre club) et des numéros de téléphones. De cette façon, les personnes membres sont indépendantes de annuaire. On peut ainsi avoir des personnes de notre club qui ont des numéros de téléphone et d'autres non ; mais toutes celles qui ont un téléphone sont forcément une des personnes de notre club.

Nous voulons développer les opérations de gestion de l'annuaire :

— ajout d'une nouvelle personne dans le club (les personnes membres),
— ajout d'un numéro de téléphone à une personne qui est déjà membre,
— consultation du ou des numéros de téléphone d'une personne dont on donne le nom.
— consultation du nom de la personne (ou des noms des personnes) ayant un numéro donné.
— suppression des coordonnées (nom, numéro) d'une personne de l'annuaire ;
— modification d'un numéro de téléphone d'une personne dont on donne le nom ;
— etc.

Ces opérations seront utilisées par un programme qui sera embarqué sur des téléphones portables.

Q#4 Développez en B une machine abstraite qui modélise l'annuaire et spécifie ses différentes opérations. On fera **deux versions**, une version où une personne n'a qu'un seul numéro de téléphone et une version ou une personne peut avoir plusieurs numéros de téléphone.

Exemple de machine B avec fonctions et opérations

```
/*? % Toujours mettre le "Qui/Quoi/Quand"
* Author: NOM Prenom
* Project : Specification d'une machine de gestion de reservoirs
?*/
MACHINE   TankMgr
    /* a TankMgr, manages several tanks; each tank has its own Id, a typeOfFioul, a volume;
       the TankMgr, supply/withdraw a given tank with volume     */
SETS /* liste des ensembles abstaits utilisés */
    TANK
    ; TANK_ID
    ; FIOUL_TYPE = {leadfree, gasoil}
VARIABLES /* liste des variables utilisées */
    theTanks
    , tkId
    , tkVol, tkCapacity
    , tkFtype
INVARIANT  /* Prédicat de typage des variables et de description des propriétés */
    theTanks <: TANK    /* un sous-ensemble de TANK */
&    tkId : theTanks >-> TANK_ID /* fonction totale injective */
&    tkVol : theTanks --> NAT
&    tkCapacity : theTanks --> NAT /* chaque cuve a sa capacité */
&    tkFtype : theTanks --> FIOUL_TYPE
INITIALISATION
       theTanks := {} || tkId := {}  || tkVol := {} || tkCapacity := {} || tkFtype := {}
OPERATIONS
        supplyTank(aTk,vv ) = /* approvisonner de vv volume */
        PRE
            aTk : TANK  &   vv  : NAT
        &   aTk : theTanks
        &   tkVol(aTk) + vv < tkCapacity(aTk) /* on ne va pas deborder */
        THEN
            tkVol(aTk) := tkVol(aTk) + vv
        END
/* il y a d'autres operations qui ne sont pas fournies ici */
END
```

Exercice : opérations sur un compte bancaire

Nous considérons des comptes bancaires tels que chaque compte est identifié par un numéro unique, a un solde, et une catégorie. Il n'y a que deux catégories, soit la catégorie C1, soit la catégorie C2.

Un compte bancaire de la catégorie C1 est un compte pour lequel la banque n'autorise aucun découvert (le solde doit toujours être positif ou nul) et il y a une limite maximale à la somme totale qu'on peut déposer sur ce compte.

Un compte de la catégorie C2 est un compte pour lequel un découvert de seuilBasC2 est autorisé ; il n'y a pas de limite maximale dans ce cas.

Ces deux exigences doivent apparaître dans l'invariant.

Analysez les données, modélisez les, puis écrivez une machine abstraite avec les opérations nécessaires pour effectuer les dépots et les retraits sur des comptes.

Pour une opération de retrait ou d'approvisionnement, le propriétaire du compte donne le numéro du compte et le montant associé à l'opération.