Exo 1

1

1

a) $B = B_0 + 80, A = A_0 - 150, C = C_0 + 70$

b) $B = B_0 + 80, A = A_0 - 150, C = C_0 + 70$

c) $B = B_0 + 150, A = A_0 - 150, C = C_0 + 70$

2

3

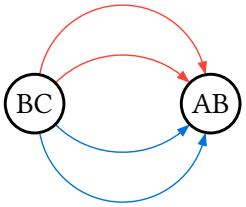
b) Oui, on peut faire les deux exécutions dans le sens qu'on veut

c) impossible

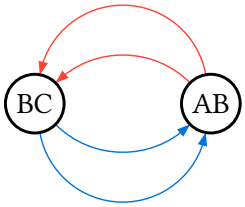
4

a)

b)



c)



Exo 2

2

1

```
procedure transfert(CompteDébité, CompteCrédité, montant)
  verrouillage(CompteDébité);
  variable := Lire(CompteDébité);
  Ecrire (CompteDébité, variable – montant);

  verrouillage(CompteCrédité);
  variable := Lire (CompteCrédité);
  Ecrire(CompteCrédité, variable + montant);
  déverrouillage(CompteDébité);
  déverrouillage(CompteCrédité);
```

2

b) L1(A)L2(B)E2(B)L2(C)E1(A)L1(B) blocage à L1(B)

c) L1(A)E1(A)L1(B)L2(B) blocage à L2(B)

2

1

2

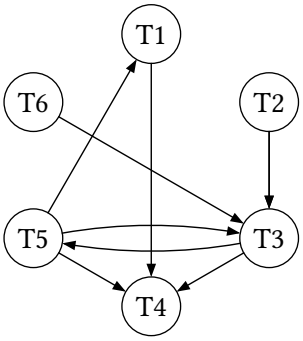
Le verrou étant présent sur A, impossible de faire la somme.

3

Il n'y aurait aucun verrou et A pourrait opérer. On aurait comme résultat $A_0 - 150 + B_0$

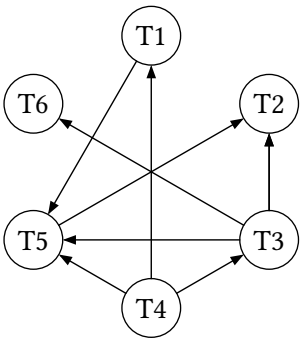
3

1



2

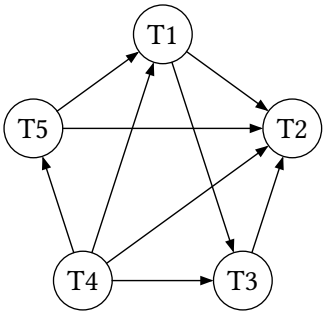
A: e2, e3, l5 B: l2, l4, l1 C: e5, l1, l3, e4 D: l6, l2, e3



Pas de cycle, pas de deadlock, on assure la finition

4

1



Pas de cycle: sérialisable

$T_4 \rightarrow T_5 \rightarrow T_1 \rightarrow T_3 \rightarrow T_2$[1]

[1] Algorithme: Sérialisable donc sans cycle donc il existe un noeud qui ne connait personne. Il est dernier. On le supprime du graphe ainsi que toutes ses connexions. On sélectionne le prochain noeud qui ne connait personne, on répète. Si 2 choix possibles, deux possibilités

2
Sérialisable donc marche avec double verrou

5

1
V2(T) L2(T) V(Y) L3(Y) V(Y) V3(X) L3(X) V3(T) L3(T) E2(T) D2(T) E2(X) E3(Y) C3 E5(X) C5 L2(Y) L2(Z) C2 E4(Z) E1(X) L1(Y) E1(Y) L4(T) L1(Z) C1 E4(T), C4

marre

Exo 3

a)
Sans Thomas Refus par L2(b) car E3(b) avant
Avec Thomas Toujours refus, Thomas s'applique aux écritures

b)
Sans Thomas Refus par E1(a) car E2(a) avant
Avec Thomas Accepté, si E1(a) est exécutée entre L1(a) et E2(b) il n'y a aucun effet

c)
Sans Thomas Refus par E1(a) car L2(a) avant
Avec Thomas Toujours refus car T(La) > 1

d)
Sans Thomas Refus par E2(a) car E3(a) avant
Avec Thomas Refusé: On applique Thomas avec E2(a) mais toujours bloqué par L2(a) après