**Pseudo-code (BRAIN)**

1. **Initialisation**
   1. initialisation des variables
   2. initialisation dispositifs:
      1. moteurs (mot\_chenille\_gauche, mot\_chenille\_doite, mot\_ramasseur)
      2. capteurs IR (#1,#2,#3)
      3. caméras linéaires (#1,#2,#3,#4,#5,#6)
      4. caméra RP (VGA)
      5. capteurs photo-électriques (#1,#2,#3,#4)
   3. initialisation communication
      1. avec Raspberry Pi (=RP)
      2. avec autre Arduino
   4. initialisation des timers
2. **Partir de la maison (Start)**
   1. temps++ (démarrage clock = commence à compter) [fréq, enable,...]
   2. lire caméra RP [communication Raspberry Pi] (+image processing)
   3. lancer les moteurs du ramasseur
   4. avancer
3. **Chercher une bouteille**

→ aller en direction bouteille → lire angle, distance [communication Raspberry Pi]

→ *OPT: Aller d'abord dans la zone caillou → lire caméra linéaire + lire caméra RP (LED Rouge)*

* 1. **Si je vois un obstacle:**
     1. vérifier obstacles → brique → contourner

→ mur → tourner sur soi 60°

* 1. **Si je vois une bouteille:**
     1. s'approcher bouteille ( = lire angle , avance(angle, distance))
     2. vérifier bouteille dans entrée container (= lire capt\_photoélectrique #1)

→ non→ s'approcher bouteille {3.b.i}

→ oui → Cherche une bouteille {3}

[en //:] → vérifier bouteille dans container → oui → compte\_b +1 (\*)

→ non → (stop?) mot\_ramasseur tourne à l'envers

(\*) vérifier si container plein: si (compte\_b ==6) → rentrer à la maison {4}

1. **Rentrer à la maison** 
   1. lire caméra linéaire + caméra RP [communication Raspberry Pi]
   2. trouver LED Jaune → orientation du robot
   3. déplacement vers maison
   4. à la maison (LED Jaune == grande intensité) → déposer les bouteilles
   5. vérifier si container est vide (= lire capt\_photoélectrique #4)

→ non → danse bernoise → {4.e}

→ oui → Cherche une bouteille {3}

1. **Interruptions [en // de {3}:]** 
   1. vérifier le temps (si temps == 9min) → rentrer à la maison {4}
   2. si bumpers == oui → vérifier si (obstacle\_bump == obstacle) → tourner

(obsacle\_bump == bouteille) → {3.b}

* 1. si IR == obstacle proche → tourner