Raffinage programme principal

R0 : “Comment interpréter un code intermédiaire”

R1 : Comment R0 ?
- Définir le tableau d’instructions
- Remplir tableau (instructions)
- Initialiser les variables en mémoire (allocation)

TANT QUE “non fini”
- Récupérer l’instruction

  Si GOTO x ou IF FAIRE
  - Passer à la ligne x

  SINON FAIRE
  - Réaliser l’instruction
  - Passer à la ligne suivante

  FIN SI

FIN TQ

R2 : Comment “Définir le tableau d’instructions” ? **in Fichier** / **out Capacite**

=> appel de *CalculerCapacite*, fonction interne qui calcule la capacité du tableau en fonction du nombre de lignes du fichier

R2 : Comment “Remplir tableau (instructions)” ? **In Fichier** / **out Tab**

=> appel de *Remplir_tab_instruc*

R2 : Comment “Initialiser les variables en mémoire” ? **In Code** / **Out Mem**

Pour chaque ligne jusqu’à “Début” :
- Split jusqu’à “:” pour récupérer le type
- Split entre chaque “,” pour récupérer les noms des variables

  Pour chaque variable :
  - Initialisation d’une case mémoire avec le bon type

    => appel de Initialiser, procédure interne qui crée la mémoire (= Null) : Out Mem
  - Valorisation de sa clé et de son type et de son suivant si nécessaire

  Fin pour

Fin pour

R2 : Comment “non fini” ? **In Code** / **out estFini**

Il reste une ligne à lire

R2 : Comment “Récupérer l’instruction” ? **in Code, CP** / **out part1, part2, part3, part4**

=> Première partie de “effectuer_instru”

R2 : Comment “Réaliser l’instruction” ? **in part1, part2, part3, part4** / **in out CP, in out mem**

=> Deuxieme partie de “effectuer_instru”

R2 : Comment “Passer à la ligne suivante” ? (Quand non (GOTO ou IF)) **in out CP**

- Incrémentation de CP

Raffinage fonction “Remplir_tab_instruc”

R0 : Comment “Remplir tableau (instructions)” ? **out Tab**, **in Fichier**

- Lire jusqu’à “Début”

POUR chaque ligne FAIRE

- Mettre dans le tableau sous la bonne forme (les commentaires deviennent des null) : **out Tab**, **in Fichier**

FIN POUR

R1 : Comment “Mettre dans le tableau sous la bonne forme (les commentaires deviennent des null)” ? **out Tab**, **in Fichier**

POUR chaque ligne du code intermédiaire FAIRE

- Récupérer le premier mot : **in out ligne, mot**; **in delimiteur**

  SI premier mot de la ligne est “null” ou “--” THEN
  - Rempli la première case de cette ligne en “null”

  ELSE
  - Récupérer le reste de la ligne et le placer dans le tableau

  FIN SI

FIN POUR

R2 : Comment “Récupérer le premier mot” ? (Fonction Slice_mot) : **in out ligne, mot**; **in delimiteur**

- Trouver l'indice du délimiteur
- Extraire la sous-chaîne jusqu'au délimiteur

SI Delimiteur trouvé THEN

- Stock le mot
- Supprimer le mot de la ligne

SINON

- Copie la ligne entière dans le mot
- Réinitialiser la ligne

FIN SI

Raffinage fonction “CalculerCapacite”

R0 : Calculer la capacité du tableau d'instructions ? **in Fichier** / **out Capacite**

R1 : Comment "Calculer la capacité du tableau d'instructions" ? **in Fichier** / **out Capacite**

1) Parcourir le fichier jusqu'à "début" **in Fichier**
2) Calculer le nombre de lignes de la ligne début + 1 / jusqu'à la fin du programme : out Capacite **in Fichier** / **out Capacite**

R2 : Comment "Parcourir le fichier jusqu'à "début"" **in Fichier**

```
Tant que not (Ligne = "Debut" ou Ligne = "Début")
   Lire la ligne suivante
FTQ
```

R2 : Comment "Calculer le nombre de lignes de la ligne début + 1 / jusqu'à la fin du programme (mot "Fin")" ? **in Fichier** / **out Capacite**

```
Capacite <- 0
Tant que not (Ligne = "Fin") faire
   Capacite <- Capacite + 1
   Lire la ligne suivante
FTQ
```

Raffinage “effectuer_instru”

=> mise en place des deux raffinages suivant du programme principal

R2 : Comment “Récupérer l’instruction” ? **in Tab _instruction, CP** / **out part1, part2, part3, part4**

- Récupérer la partie 1 de l’instruction (dans le tableau d’instruction à l’indice CP) **in Tab, CP** / **out part1**
- Récupérer la partie 2 de l’instruction **in Tab, CP** / **out part2**
- Récupérer la partie 3 de l’instruction **in Tab, CP** / **out part3**
- Récupérer la partie 4 de l’instruction **in Tab, CP** / **out part4**

R2 : Comment “Réaliser l’instruction” ? **in part1, part2, part3, part4** / **in out CP, in out mem**

- En fonction du premier mot de l’instruction récupérée => appel de sous-programme

Si part1=”Null” : ne fait rien
Sinon Si part1=”GOTO” : va à la ligne demandée
Sinon Si part1=”IF” : évalue le “if” et va à la ligne en lien
Sinon

Si part3=”affect” : effectue l’affectation demandée
Sinon : effectue l’opération demandée
FSI
FSI