```
* Phase 3

pFiModel = ouvrir "fiModel.csv" en lecture
pFiTest = ouvrir "testSet.csv" en lecture

if(pFiModel AND pFiTest)

o----------------------o ↓ pFiModel, models, ESTSET_NAME, MAX_LINES_TESTSET, MAX_LINES_IGNORE_TESTSET, movementsNames
| convertFileToTable |
o----------------------o ↓ models

o----------------------o ↓ pFiTest, testSet, MODELS_NAME, MAX_LINES_PATTERN, MAX_LINES_IGNORE_PATTERN, movementsNames
| convertFileToTable |
o----------------------o ↓ testSet

o--------------o ↓ testSet,models
| traitement |
o--------------o

Fermer pFiTest , pFiModel

else
sortir "Impossible d'ouvrir le fichier"
```

```
o----------------------o ↓ fichier,nbLines, nbColumnsToIgnore,movementsNames
| convertFileToTable |
o----------------------o ↓ tab
* 

ouvrir la ligne d'entête à ignorer

lineNumber = 0
while(lineNumber < nbLines)

ouvrir la ligne

nextToken = NULL
token = strtok_s(line,",",&nextToken)

// Rechercher son indice de mouvement en fonction de son nom
iName = 0
while(iName < MAX_MOVEMENTS AND token == movementsNames[iName])
iName++


tab[lineNumber].move = iName + 1

// Extraction des valeurs de chaque ligne en ignorant le nombre de colonnes nécessaire
// Pour le testSet , il faut ignorer les 3 premieres colonnes
// Pour le pattern, il faut en ignorer qu'une , d'où la variable nbColumnsToIngore

iTemp = 0
while(iTemps < MAX_TEMPS + nbColumnsToIgnore AND token ≠ null)
if(iTemps ≥ nbColumnsToIgnore)
tab[lineNumber].v_acc[iTemps - nbColumnsToIgnore] = token

token = élément suivant de la ligne en cours
iTemp++

lineNumber++
```

```
o--------------o ↓ testSet,models
| traitement |
o--------------o
* 
iLine = 0
while(iLine < MAX_LINES_TESTSET)

currentMovement = testSet[iLine].move
nbIdenticalValues = 0
count = 0

while(iLine < MAX_LINES_TESTSET AND currentMovement == testSet[iLine].move )
realClasses[iLine] = currentMovement
distance = 0
lowValue = HV
iMovementLowDistance = 0

iMovement = 0
```

```
    while (iMovement < 6)
    o─────────o ↓ testSet[iLine].v_acc,models[iMovement].v_acc
    | calcul |
    o─────────o ↓ distance
     if(distance < lowValue)
     lowValue = distance
     iMovementLowDistance = iMovement

     iMovement++


    estimateClasses[iLine] = iMovementLowDistance + 1
    Sortir "RealClasses : %d et EstimateClasses : %d", realClasses[iLine] , estimateClasses[iLine]
     if(realClasses[iLine] == estimateClasses[iLine])
     nbIdenticalValues++

    count++
    iLine++

    Sortir "Pourcentage de valeur identique pour ce mouvement : %lf %" (nbIdenticalValues / count) *100
```

```
o─────────o ↓ tab1,tab2
| calcul |
o─────────o ↓ distance
   *

 distance = 0

 iTemps = 0
    while(iTemps < MAX_TEMPS)
 distance += (tab1[iTemps] - tab2[iTemps]) * (tab1[iTemps] - tab2[iTemps])
 iTemps++


 distance = racine(distance)
```