```
entrée : fiModele.xlsx
┌─── *
│ obtenir tailleEchantillon // int
│ obtenir alphaControl // 5%
│ obtenir alphaWarning // 1%
│
│ n = 0
│ sommeXi = 0
│ sommeXiCarré = 0
│ o──────────────────────────────────────o ↓  *f
│ │ construction de la table ( Simpson ) │
│ o──────────────────────────────────────o ↓  tabNormale
│ xi = lire dataFile
│ ╔══ do while (!EOF)
│ ╠ sommeXi += xi
│ ╠ sommeXiCarré += xi * xi
│ ╠ xi = lire dataFile
│ ╠ n++
│ ╚═══
│ moyenne = somme xi / n
│ variance = sommeXiCarré / n - (moyenne * moyenne)
│ o──────────────────────────────────────────o ↓  alphaControl, tabNormale
│ │ recherche de la valeur en fonction de alpha │
│ o──────────────────────────────────────────o ↓  coéfficientAlphaControl
│
│ o──────────────────────────────────────────o ↓  alphaWarning, tabNormale
│ │ recherche de la valeur en fonction de alpha │
│ o──────────────────────────────────────────o ↓  coéfficientAlphaWarning
│
│ o──────────────────────o ↓  moyenne, variance, n, coéfficientAlphaControl
│ │ calcul de l'intervalle │
│ o──────────────────────o ↓  lowerControlLimit, upperControlLimit
│ sortie de l'intervalleControle
│
│ o──────────────────────o ↓  moyenne, variance, n, coéfficientAlphaWarning
│ │ calcul de l'intervalle │
│ o──────────────────────o ↓  lowerWarningLimit, upperWarningLimit
│ sortie de l'intervalleSurveillance
│
│ o──────────────────────────────o ↓  tailleEchantillon, lowerControlLimit,
upperControlLimit, lowerWarningLimit, upperWarningLimit, dataAEvaluer
│ │ traitement sur base du modèle │
│ o──────────────────────────────o
└──
```

```
* construction de la table ( Simpson )
// comme la loi normale est symétrique j'ai choisi de construire mon tableau a partir de x = 0;
nbPts = 3 // précision suffisante avec un intervalle de 0.01
dernierRésultat = 0.5
tabNormal[0][0] = dernierRésultat
bInf = 0
iLigne = 0
do while (iLigne < 40) // attention travailler en entier
 iColonne = 0
 do while (iColonne < 10) // attention travailler en entier
  bSup = iLigne / 10 + iColonne / 100
  o──────────o ↓  nbPts, bInf, bSup, *f // f est un pointeur vers l'inégrale de gauss
  │ Simpson │
  o──────────o ↓  resultat

  dernierRésultat += resultat
  tabNormal[iLigne][iColonne] = resultat * 10000 // pour ne pas avoir de problème de
conversion binaire mais ???
  bInf = bSup;
  iColonne++

 iLigne++
```

```
* recherche de la valeur en fonction de alpha
 interpolation = 0
 zoneNormaliser = (1 - alpha / 200) * 10000

 iLigne = 0
 do while (iLigne < 39 AND tabNormale[iLigne + 1][0] ≤ zoneNormaliser)
 iLigne++


 iColonne = 0
 do while (iColonne < 9 AND tabNormale[iLigne][iColonne + 1] ≤ zoneNormaliser)
 iColonne++


 if (tabNormale[iLigne][iColonne] ≠ zoneNormaliser)
  iLigneBord = iLigne
  iColonneBord = iColonne + 1
  if (iColonne == 9 AND tabNormale[iLigne][iColonne] <  zoneNormaliser)
   iLigneBord = iLigne + 1
   iColonneBord = 0

  interpolation = (zoneNormaliser - tabNormale[iLigne][iColonne])
  interpolation /= (tabNormale[iLigneBord][iColonneBord] -
tabNormale[iLigne][iColonne])
 coéfficientAlpha = iLigne / 10 + iColonne / 100 + interpolation / 100
```

```
* calcul de l'intervalle
 margeErreur = coefficentAlpha * sqrt(variance / n);
 intervalleInf = moyenne - margeErreur;
 intervalleSup = moyenne + margeErreur;
```

```
* traitement sur base du modèle

sommeEchantillon = 0
xi = lire dataAEvaluer
int numEchantillon = 1;
tailleReelleEchantillon = 0;
do while (!EOF)
    do while (tailleReelleEchantillon < tailleEchantillon)
        sommeEchantillon += xi
        xi = lire dataAEvaluer
        tailleReelleEchantillon++
    xBarre = sommeEchantillon / tailleReelleEchantillon

    if (xBarre < lowerWarningLimit || xBarre > upperWarningLimit)
        libEchantillion = "Echantillon " + numEchantillon + " composé de " +
tailleReelleEchantillon + " valeurs : "
        texte = "nous sommes en dehors de l'intervalle de surveillance"
        if (xBarre < lowerControlLimit || xBarre > upperControlLimit)
            texte = "valeurs érronées"

        sortir libEchantillion + texte

    numEchantillon++
```