**2.1.2**

**Chaine** distanceParcourueEnChaine

**Chaine** uniteConversion

**Chaine** distanceParcourue

**Booleen** saisieCorrecte

**Booleen** veutQuitter

**Entier** indexDistanceParcourue

**Réel** distanceParcourureEnReel

distanceParcourueEnReel < - - 0

veutQuitter < -- false

**Ecrire** « programme de conversion kilomètres <-> Miles »

**Faire**

saisieCorrecte < -- true

**Ecrire** « saisir une valeur et son unité séparés par un espace : »

**Lire** distanceParcourue

**Si** (distanceParcourue == ‘q’) **alors**

veutQuitter < -- true

**Sinon**

**Si** (|distanceParcourue| > 3) **alors**

uniteConversion < -- distanceParcourue<|distanceParcourue|-1>

uniteConversion < -- distanceParcourue<|distanceParcourue|>

indexDistanceParcourue < -- 1

**TantQue** (indexDistanceParcourue > |distanceParcourue|-3)

distanceParcourueEnChaine <-- distanceParcourue<indexDistanceParcourue>

**Faire**

**Si** (distanceParcourueEnReel < 0.01

OU distanceParcourueEnReel > 1 000 000) **Alors**

saisieCorrecte <-- false

**FinSi**

**Sinon**

saisieCorrecte <-- false

**FinSI**

**FinSi**

**TantQue** (veutQuitter == false ET saisieCorrecte == false)

**Si** (veutQuitter == false) **alors**

**Réel** resultatConversion

**Si** (uniteConversion == « mi » ) **alors**

resultatConversion <-- distanceParcourueEnReel \* 1.609

Ecrire distanceParcourueEnReel, «  miles =  », resultatConversion, « Kilomètres »

**Sinon**

resulatConversion <-- distanceParcourueEnReel / 1.609

Ecrire distanceParcourueEnReel, «  Kilomètres =  », resultatConversion, « Miles »

**FinSI**

**FinSI**