Institut des Sciences et Techniques de Valenciennes
Licence Sciences, Technologie, Santé
Les bases de l'algorithmique et de la programmation
2018-2019
TP 2

Environnement de travail

Le langage Scala servira de support au module. L'environnement de travail est lancé par l'article Scala IDE dans le menu Démarrer. Cet environnement rassemble un éditeur de code qui enrichit le texte source de la syntaxe du langage, un compilateur qui vérifie la syntaxe et génère le code exécutable.

L'environnement permet d'accéder aux opérations nécessaires aux quelques opérations nécessaires à une bonne manipulation :
Créer un projet Scala : Onglet File/New/Scala Project et définissez le nom de votre projet (par exemple TP1).
Créer une nouvelle application Scala : Onglet File/New/Scala App et définissez le nom de votre application (par exemple Exo1). Cette opération sera à renouveler pour chaque exercice.
Ecrire votre application : dans la fenêtre centrale, tapez le code Scala
|ester votre application : Enregistrer votre application et avec la flèche verte, exécuter votre application en choisissant Run as/Scala Application. La fenêtre du bas Console vous permet d'interagir avec votre application.

Exercice 1. Boucle tant que

Le programme suivant illustre la façon d'écrire en Scala la boucle tant que.

```
import scala.io.StdIn ;

object exemple_while extends App {
    var i : Int = 20 ;

    while (i > 0) {
        println ("i = " + i) ;
    }
}
```

Il correspond à l'algorithme suivant.

```
programme exemple_while
    i : ENTIER ;
début
    tant que i > 0 faire
        écrire("i = ", i) ;
    fin tant
fin
```

1. Saisir et exécuter le programme.
2. Identifier l'incohérence.
3. Modifier le code pour l'éliminer.

Exercice 2. Boucle pour

Le programme suivant illustre la façon d'écrire en Scala la boucle pour.

```
import scala.io.StdIn ;

object exemple_for extends App {
    for (i <- 0 to 10) {
        print(i) ;
    }
    for (i <- 0 to 10 by 2) {
        print(i) ;
    }
    println() ;
    println() ;
    for (i <- 0 to 10 by -1) {
        print(i) ;
    }
    for (i <- 10 to 0 by -2) {
        print(i) ;
    }
}
```

Il correspond à l'algorithme suivant.

```
programme exemple_for
début
    pour i de 0 à 10 pas 1 faire
        ecrire(i) ;
    fin pour
    pour i de 0 à 10 pas 2 faire
        ecrire(i) ;
    fin pour
    pour i de 0 à 10 bas 1 faire
        ecrire(i) ;
    fin pour
    pour i de 10 à 0 bas 2 faire
        ecrire(i) ;
    fin pour
fin
```

1. Saisir et exécuter le programme.
2. Améliorer l'affichage
3. Identifier les incohérences.
4. Modifier le code pour les éliminer.
5. Réécrire les boucles pour avec des tant que.

Exercice 3. Codage et test

1. Coder en Scala les algorithmes de l'exercice 1 de la fiche 3 d'exercices de TD.
2. Vérifier que les résultats vus en TD coïncident avec les versions codées.

Exercice 4. Conception d'algorithme, codage et test

On donne l'algorithme suivant.

```
programme affichage_de_valeurs
    choix : ENTIER ;
    borne_inf, borne_sup : ENTIER ;
début
    choix <- -1 ;
    écrire("Entrer la valeur de la borne inférieure : ") ;
    lire(borne_inf) ;
    écrire("Entrer la valeur de la borne supérieure : ") ;
    lire(borne_sup) ;
    tant que choix ≠ 0 faire
        écrire("0 - Quitter le programme") ;
        écrire("1 - Afficher les valeurs comprises entre les bornes") ;
        écrire("2 - Afficher les valeurs paires comprises entre les bornes") ;
        écrire("3 - Afficher les valeurs impaires comprises entre les bornes") ;
        écrire("Entrer votre choix : ") ;
        lire(choix) ;
        si choix = 1 alors
            écrire("Valeurs entre les bornes") ;
        fin si
        si choix = 2 alors
            écrire("Valeurs paires entre les bornes") ;
        fin si
        si choix = 3 alors
            écrire("Valeurs impaires entre les bornes") ;
        fin si
    fin tant
fin
```

1. Coder l'algorithme en Scala.
2. Compléter l'algorithme pour réaliser les traitements demandés.
3. Tester le code.

Exercice 5. Test d'algorithmes

1. Coder en Scala les algorithmes des exercices 5 et 6 de la fiche 3 d'exercices de TD.
2. Concevoir les jeux de données de test.
3. Réaliser les tests.

Exercice 6. Comparaison d'algorithmes

Soit les deux algorithmes suivants de calcul du PGCD de deux entiers.

```
programme PGCD1
    n, p : ENTIER ;
début
    lire(n, p) ;
    tant que a*b ≠ 0 faire
        si a > b alors
            a <- a - b ;
        sinon
            b <- b - a ;
        fin si
    fin tant
    si a ≠ 0 alors
        écrire(a) ;
    sinon
        écrire(b) ;
    fin si
fin
```

```
programme PGCD2
    n, p, r : ENTIER ;
début
    lire(n, p) ;
    r <- n mod p ;
    tant que r ≠ 0 faire
        a <- b ;
        b <- r ;
        r <- n mod p ;
    fin tant
    écrire(b) ;
fin
```

1. Coder en Scala les algorithmes.
2. Modifier les programmes pour de mettre en évidence le plus intéressant.

Exercice 7

1. Coder en Scala l'algorithme de l'exercice 3 de la fiche 3 d'exercices de TD.
2. Coder en Scala l'algorithme de l'exercice 4 de la fiche 3 d'exercices de TD.
3. Ecrire un algorithme déterminant le minimum et le maximum d'une liste d'entiers saisis un par un.