INF1256 Informatique pour les sciences de la gestion

– *Structures répétitives et Expressions régulières* –

Johnny TSHEKE, Ing. Jr.

Université du Québec à Montréal
Département d'Informatique
tsheke_shele.johnny@uqam.ca

Séance 05

Notions de base des expressions régulières

- chaines de caractères ou pattern utilisés pour vérifier les formats des textes
- facilitent la validation des données
- chaines des caractères ex : `"INF2005"` cherche cette chaine
- x|y x ou y. ex : `"INF2005|INF3005"`
- [abc] une lettre parmi a, b et c
- [a-z] une lettre entre a et z
- [0-9] un chiffre entre 0 et 9
- \w n'importe quel caractère alphanumérique : `"[A-Za-z0-9\_]"`
- \W négation de \w (minuscule)
- . le point correspond à n'importe quel caractère
- \ précède un caractère spécial
- {n} n occurrences. ex : `"a{3}"` pour 3 a consécutifs
- {n,m} entre n et m occurrences. ex : `"a{3,5}"` pour entre 3 et 5 a consécutifs
- * 0 ou plusieurs occurrences → {0,}
- ? 0 ou 1 occurrence →{0,1}
- + 1 ou plusieurs occurrences →{1,}
- (chaine) une sequence à l'intérieur d'un pattern. ex : `"a?(chaine)b+"`
- $ fin de la chaine. ex : `.{1,}5$` une chaine qui se termine avec 5
- ^ début de la chaine. ex : `^J\w*` commence avec J suivi d'un nombre variable de caractères alphanumériques
- [^ abc] négation de l'ensemble. Une lettre autre que a, b ou c
- ...

⇒ `https://regex101.com` ,
⇒ `https://developer.mozilla.org/en/docs/Web/JavaScript/Guide/Regular_Expressions`

⇒ `https://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html`

Expressions régulières et validation de données en entrée

- Utiliser la méthode `hasNext(pattern)` de la classe Scanner
- pattern est une chaine (String) d'une expression régulière

Instruction `while` en Java

- En pseudocode → TANT QUE
- Forme générale

```
while ( expression ){
        //mettre les instructions du corps de la boucle ici
  }
```

- expression doit être vraie (true) pour exécuter le corps de la boucle
- On exécute le corps de la boucle aussi longtemps que expression restera vraie

⇒ Veuillez à ce que expression devienne fausse (false) à un moment ou un autre. Sinon, boucle infinie et un programme qui ne s'arrête pas !

Illustration de while

```
while( expression ) {
        // instructions du corps de la boucle décalage
        // modification possible de expresssion
         }
```

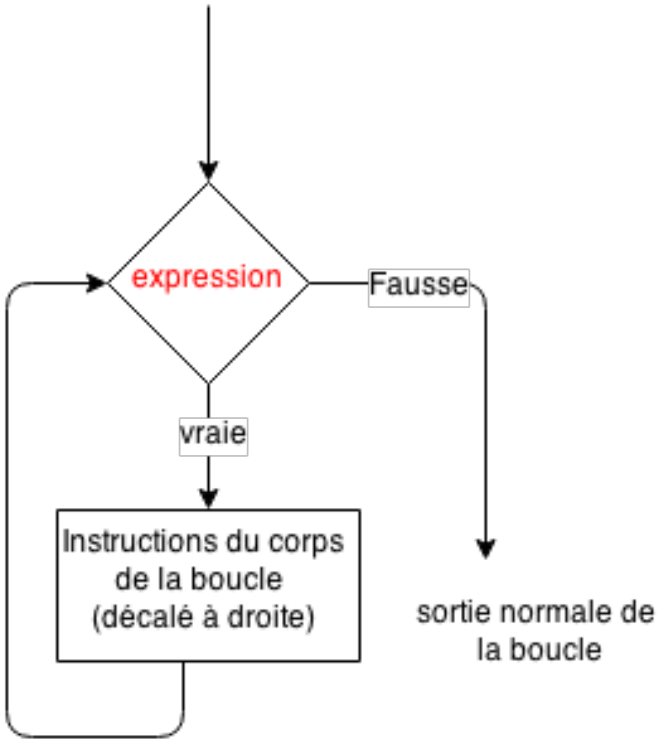
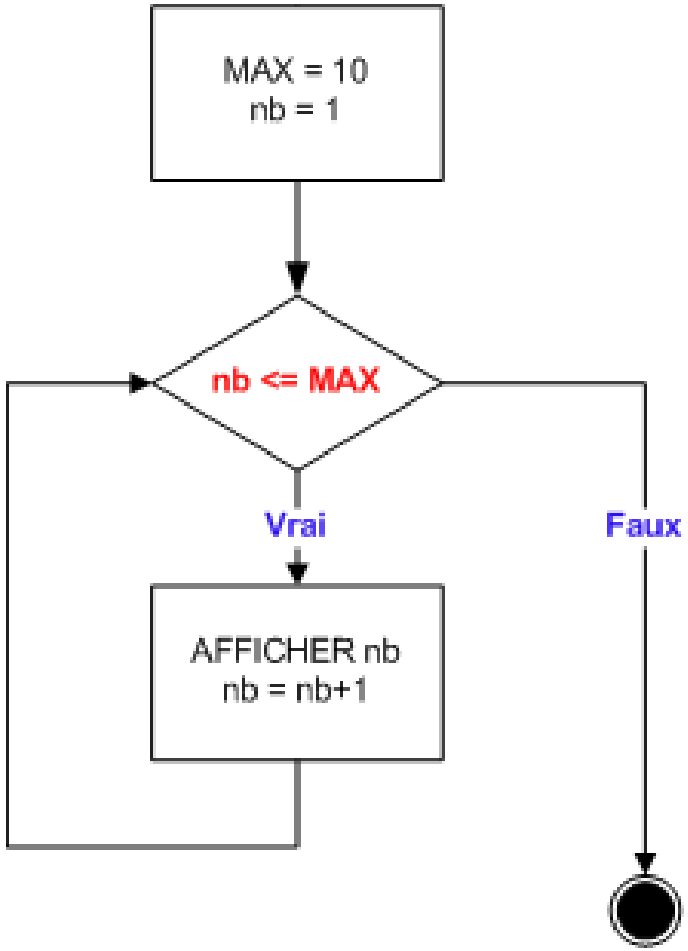


Illustration de while en pseudocode

Afficher un à un les nombres 1 à 10

```
MAX←10
nb←1
TANT QUE nb <= MAX
   AFFICHER nb
   nb← nb+1
FIN TANT QUE
```



- la boucle continue tant que nb est inférieur ou égal à MAX (qui vaut 10)
- nb augmente à chaque itération
- quand nb égal à 11 → `nb <= MAX` devient faux et la boucle s'arrête !

Exemple d'une boucle while

Affichage des nombres 1 à 10 : AfficheNombres.java

```
package inf1256s05;

public class AfficheNombres {
    static final int MAX = 10;
        public static void main(String[] args) {
                int i = 1;
                while (i<= MAX){ // affiche 1 a 10
                        System.out.println(" i = "+i);
                        i = i + 1;
                }

        }

}
```

Exemple d'une boucle while (2)

Exemple de validation d'une réponse (Oui/Non) : ValidationReponse.java

```
package inf1256s05;
import java.util.*;
public class ValidationReponse {

        public static void main(String[] args) {
                // Valider une réponse O,o,N ou n
                String pattern = "[OoNn]";//Expression régulière
        Scanner clavier = new Scanner(System.in);
        System.out.println("Entrez O ou o pour Oui / N ou n pour Non");
        while(!clavier.hasNext(pattern)){
                System.out.println("Réponse invalide");
                System.out.println("Entrez O ou o pour Oui / N ou n pour Non");
                clavier.next();//on déplace la tête de lecture
                 //on lit sans mettre dans aucune variable var valeur invalide
        }
        String reponse = clavier.next();//valeur valide
        System.out.println("Votre réponse est: " + reponse);
        clavier.close();
        }

}
```

Exemple d'une boucle while (3)

Exemple de validation d'un nombre réel : ValidationNombreReel.java

```
package inf1256s05;
import java.util.*;
public class ValidationNombreReel {

        public static void main(String[] args) {
                // Validation nombre 2 chiffres avant point et 1 à 3 chiffres après
                //Ex: 03.000
        Scanner clavier = new Scanner(System.in).useLocale(Locale.US);
        /*ceci force les paramètres locaux à USA. donc au lieu
         * de virgule, on force à utiliser point sur les nombres réels
         */
        String pattern = "[0-9]{2}[.]{1}[0-9]{1,3}";
        boolean nombreValide = false;
        Double nombreSaisie=0.0;
        while(!nombreValide){
                System.out.println("Entrez un nombre réel ..");
                System.out.println("2 chiffres pour la partie entière");
                System.out.println("1 à 3 chiffres pour la partie décimale");
                if(clavier.hasNext(pattern)){
                        nombreSaisie=clavier.nextDouble();
                        nombreValide = true; //pour arreter la boucle
                }else{
                        clavier.next();
                }
        }
        System.out.format("Le nombre saisie est: %06.3f",nombreSaisie);
        clavier.close();
        }
}
```

Exemple d'une boucle while (4)

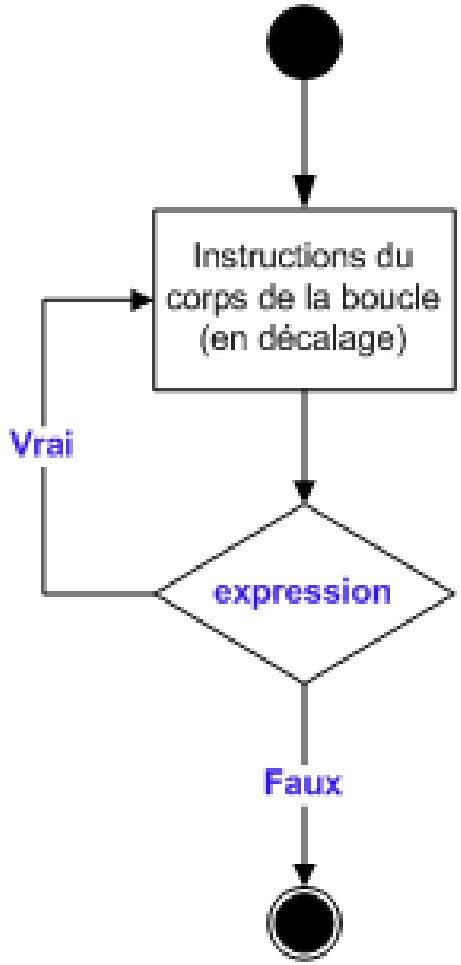
Exemple de validation d'un code permanent : CodePermanent.java

```
package inf1256s05;
import java.util.*;
public class CodePermanent {

        public static void main(String[] args) {

      Scanner clavier = new Scanner(System.in);
      String pattern = "[A-Za-z]{4}[0-9]{8}";
      boolean stop=false;
      while(!stop){
          System.out.println("Entrez un code permanent valide");
          System.out.println("4 lettres suivies de 8 chiffres");
          if(clavier.hasNext(pattern)){
                  stop = true;
          }else{
                  System.out.println("Ce code n'est pas valide");
                  clavier.next();
          }
      }
      String codePermanent = clavier.next();
      System.out.println("Le code permanent est "+codePermanent);
      clavier.close();
       }

}
```

Instruction do while en Java

- pseudocode → REPETER ... TANT QUE
- Le corps du block doit s'exécuter au moins une fois
- Forme générale :

```
do {
        //Instructions du corps de la boucle
 } while (expression )
```



Exemple do ..while

Exemple de validation d'un nombre entier : ValidationNombreEntier.java

```
package inf1256s05;
import java.util.*;
public class ValidationNombreEntier {

        public static void main(String[] args) {
                // Validation nombre entier avec do ..while
        Scanner clavier = new Scanner(System.in);
        boolean nombreValide = false;
        int nombreSaisie=0;
        // on pourrait aussi le valider avec le pattern "[0-9]+"
        do{
                System.out.format("Entrez un nombre entier svp %n");
                if(clavier.hasNextInt()){
                        nombreSaisie =clavier.nextInt();
                        nombreValide = true; //pour arreter la boucle
                }else{
            System.out.format("Pas un nombre entier valide %n");
                clavier.next();
                }
        }while(!nombreValide);
        System.out.format("Le nombre saisie est %d %n", nombreSaisie);
        clavier.close();
        }

}
```

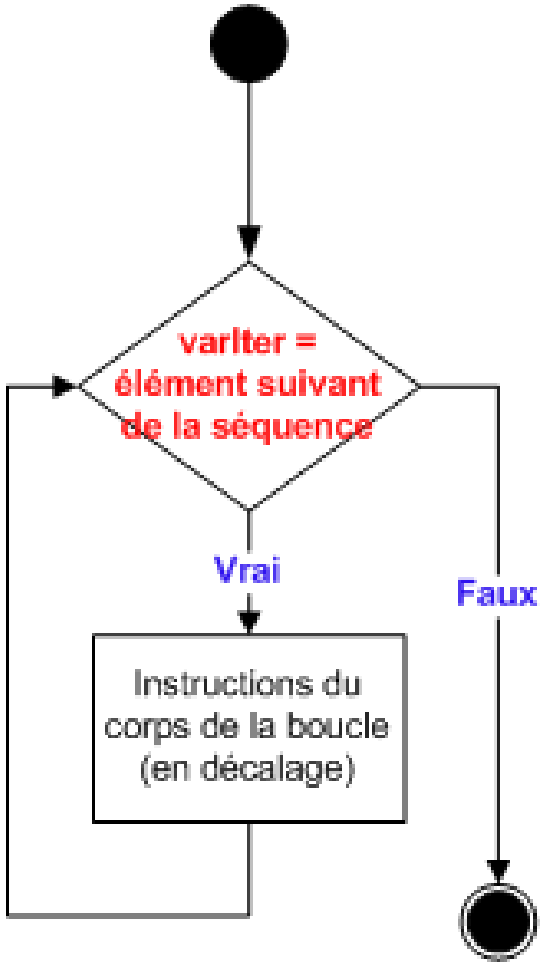
Instruction for en Java

- itération→ une exécution du corps de la boucle
- nombre d'itérations déterminer par une séquence
- Au moins une variable est modifié à chaque itération
  → Variable d'itération
- à chaque itération, la variable d'itération est associée à l'élément suivant de la séquence, en commençant par le premier

```
for (  initialisation , condition  ,  modifvar ) {
//Instructions du corps de la boucle
   }
```

Pour une boucle infinie

```
for (   ,   ,   ) {
//Instructions du corps de la boucle
   }
```



Exemple d'une boucle for

Exemple d'une boucle for : AfficheTroisFoisNon.java

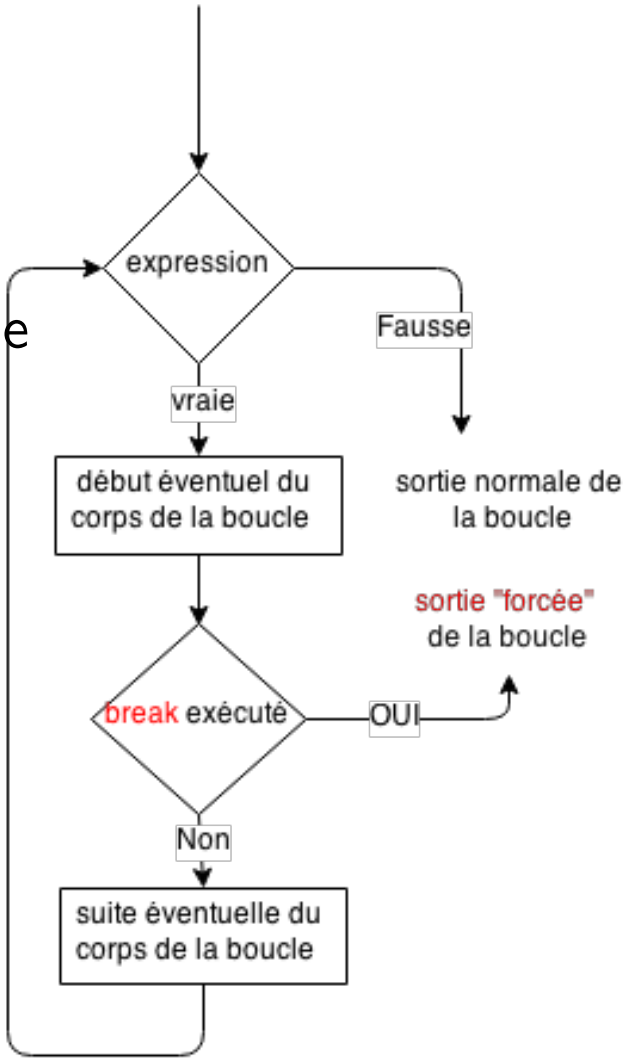
```
package inf1256s05;

public class AfficheTroisFoisNon {
    static final int MAX = 3;
        public static void main(String[] args) {
                final String NON = "NON!";
                for(int i = 0; i< MAX; i++){
                        System.out.println(NON);
                }
        }
}
```

Instruction `break` dans une boucle : `while`, `for`

- Permet de sortir directement de la boucle
- Arrête la boucle
- Si boucle imbriquer alors → sortie de la boucle la plus interne où se trouve le break et on continue dans la boucle ascendante
- S'applique dans for, while et do .. while
- La forme avec étiquette sort de la boucle correspondant à l'étiquette spécifiée

`https://docs.oracle.com/javase/tutorial/java/nutsandbolts/branch.html`

Pour une boucle while



Exemple d'une instruction break dans un `while`

Exemple d'une boucle infinie et break : BoucleInfinieEtBreak.java

```
package inf1256s05;
import java.util.*;
public class BoucleInfinieEtBreak {

        public static void main(String[] args) {

         Scanner clavier = new Scanner(System.in);
         while(true){
                 System.out.println("Entrez Un nombre entier pour arreter cette boucle ↙
    ↳ infinie");
                 if(clavier.hasNextInt()){
                         break;
                 }
                 clavier.next();
         }
         int nbStop =clavier.nextInt();
         System.out.format("Voici le nombre qui a arreté la boucle: %d %n",nbStop);
         clavier.close();
        }

}
```

Exemple d'une instruction break dans un `while` (2)

break dans une boucle while non imbriquée : ExempleWhileAvecBreak.java

```
/*
 * @ Author: Johnny Tsheke @ UQAM
 * 2017-11-27
 */
package inf1256s05;

public class ExempleWhileAvecBreak {
    static final int MAX = 3;
        public static void main(String[] args) {
                final String NON = "NON!";
                int i=0;
                while( i< MAX){
                        i++;
                        if(i==1){
                                break;
                        }
                        System.out.println(NON);
                }
        }
}
```

Exemple d'une instruction break avec étiquette dans un for

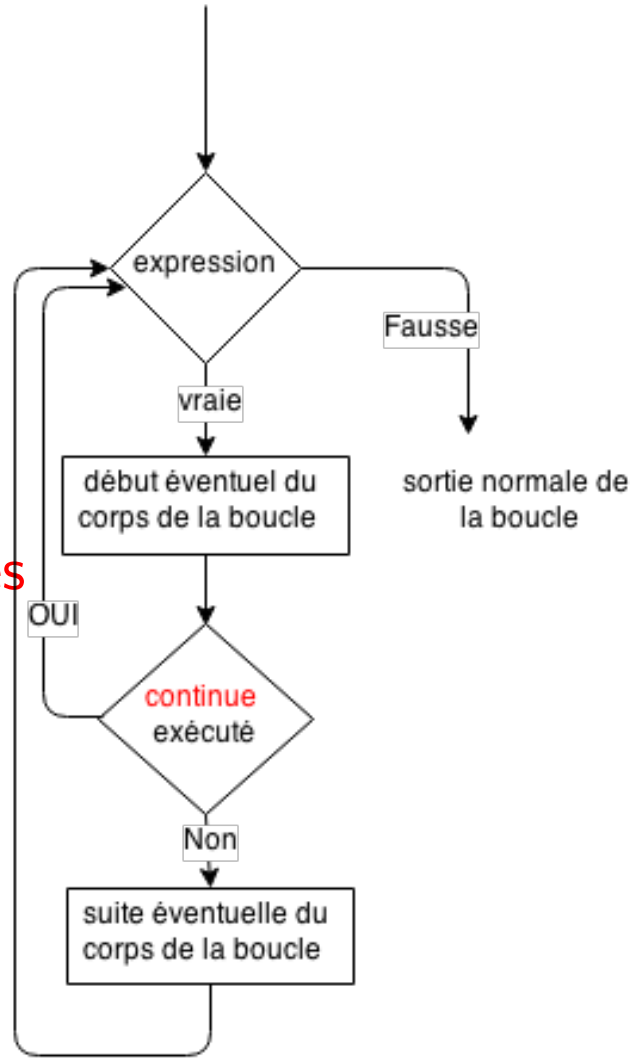
Exemple break dans for : AfficheTroisFoisNonAvecBreak.java

```
package inf1256s05;

public class AfficheTroisFoisNonAvecBreak {
    static final int MAXI = 3;
    static final int MAXJ = 5;
        public static void main(String[] args) {
                final String NON = "NON!";
                etiquette: for(int j=0;j< MAXJ;j++){
                  System.out.format("%n Ligne : %d ",j);
                 for(int i = 0; i< MAXI; i++){
                        System.out.format(" %s",NON);
                        if((j == MAXI) && (i == 1)){
                                break etiquette;// arrete la boucle de l'étiquette
                        }
                }
                }
        }
}
```

Instruction `continue` dans une boucle

- Arrête l'itération courante pour passer à la suivante
- N'arrête pas la boucle en elle même
- On reste dans la boucle mais on va vérifier les conditions pour exécuter l'itération suivante
- S'applique dans for comme dans while

Pour une boucle while



Exemple d'une instruction continue dans un `while`

continue dans une boucle while non imbriquée : ExempleWhileAvecContinue.java

```
/*
 * @ Author: Johnny Tsheke @ UQAM
 * 2017-11-27
 */
package inf1256s05;

public class ExempleWhileAvecContinue {
    static final int MAX = 3;
        public static void main(String[] args) {
                final String NON = "NON!";
                int i=0;
                while( i< MAX){
                        i++;
                        if(i==1){
                                continue;
                        }
                        System.out.println(NON);
                }
        }
}
```

Exemple d'une instruction continue dans un `for`

Exemple de continue dans for : AfficheTroisFoisNonAvecContinue.java

```
package inf1256s05;
import java.util.*;
public class AfficheTroisFoisNonAvecContinue {
    static final int MAX = 3;
        public static void main(String[] args) {
                final String NON = "NON!";
                for(int i = 0; i< MAX; i++){
                        if(i == 1){
                                continue;
                        }
                       System.out.format("%d . %s %n",i,NON);
                }
        }
}
```

Instruction `return` dans une boucle

- Sort de la méthode et retourne au point d'appel
- `Retourne une valeur si méthode retournant valeur` (ex : `return(1);`)
- `Retourne rien si une méthode void` (ex : `return;`)

⇒ Voir plus de détails au cours sur les méthodes

Exemple d'une instruction return dans un `for`

Exemple de return dans for : AfficheTroisFoisNonAvecReturn.java

```
package inf1256s05;
import java.util.*;
public class AfficheTroisFoisNonAvecReturn {
    static final int MAX = 3;
        public static void main(String[] args) {
                final String NON = "NON!";
                for(int i = 0; i< MAX; i++){
                        if(i == 1){
                                return;//on sort de la methode
                        }
                       System.out.format("%d . %s %n",i,NON);
                }
        }
}
```

Quand choisir for ou while

for : Quand le nombre d'itérations peut-être associé à une séquence donnée (déterminée d'avance) et finie. Généralement, ce qui se passe dans le corps de la boucle, ne modifie pas le nombre d'itérations.

while : Boucle plus générale. Plus appropriée lorsqu'on ne sait pas associer le nombre d'itérations à une séquence déterminée d'avance. Généralement, le nombre d'itérations varient en fonction de ce qui se passe dans le corps de la boucle au moment de l'exécution.

⇒ Quelque soit le choix, veuillez à ce que la boucle se termine.
⇒ Sinon, il faut s'assurer qu'on veut bien une boucle infinie !