TP5: String I/O formatting

Par bechir brahem GL2 groupe 2

1- Strings:

- La classe String est finale ces méthodes ne peuvent pas être surchargé/surclassé
- Quand Jvm rencontre un nouveau String literal il l'ajoute au dictionnaire
- Les méthodes de StringBuffer et StringBuilder sont les mêmes mais StringBuilder est plus rapide parce qu'il ne s'occupe pas de la synchronisation pour les thread safety.
- Quelques méthodes qui se trouvent dans la classe String:
  charAt() , concat() , equalsIgnoreCase() ,length() , replace() , substring() , toLowerCase() , toString() , toUpperCase() et trim() .

exemple1.java

```
4 public class exemple1 {
3     public static void main(String []s) {
2         String x = "Java";
1         x = x.concat(" Rules!");
5         System.out.println("x = " + x);
1         x.toLowerCase( );
2         System.out.println("x = " + x);
3         x = x.toLowerCase( );
4         System.out.println("x = " + x);
5     }
6 }
```

```
~/Desktop/java/atelier/tp5  javac exemple1.java
~/Desktop/java/atelier/tp5  java exemple1
x = Java Rules!
x = Java Rules!
x = java rules!
```

- Quelques méthodes qui se trouvent dans la classe StringBuffer et StringBuilder:
  append() , delete() , insert() ,reverse() et toString() .

exemple2.java

```
 8 public class exemple2 {
 7     public static void main(String []s) {
 6         StringBuffer sb = new StringBuffer("abc");
 5         sb.append("def");
 4         System.out.println("sb = " + sb);
 3         StringBuilder sb2 = new StringBuilder("abc");
 2         sb2.append("def").reverse().insert(3, "---");
 1         System.out.println( sb2 );
 9     }
 1 }
```

```
~/Desktop/java/atelier/tp5  javac exemple2.java
~/Desktop/java/atelier/tp5  java exemple2
sb = abcdef
fed---cba
```

2- File class:

Les objets File peuvent représenter des fichiers ou des dossiers.
On peut executer les operations comme: add, rename, delete

```
~/Desktop/java/atelier/tp5 javac shell.java
~/Desktop/java/atelier/tp5 ls
dont.java  exemple1.class  exemple1.java  exemple2.class  exemple2.java  file  input  oo  remove.class  shell.class  shell.java
~/Desktop/java/atelier/tp5 java shell
Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
1
        oo
        exemple2.java
        remove.class
        input
        file
        exemple1.class
        shell.class
        shell.java
        exemple2.class
        exemple1.java
        dont.java
```

```
~/Desktop/java/atelier/tp5 java shell
Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
2
        /home/bb/Desktop/java/atelier/tp5/.
```

```
Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
4
saisir le fichier
input
Data in the file:

test file content
```

```
Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
7
Saisir le nom de votre dossier :
dossier
        succés

Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
1
        oo
        exemple2.java
        remove.class
        input
        file
        exemple1.class
        shell.class
        shell.java
        exemple2.class
        exemple1.java
        dossier
        dont.java
```

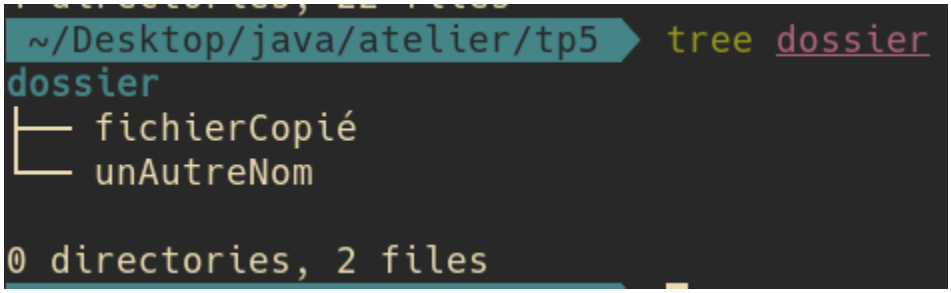
```
Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
1
        oo
        exemple2.java
        remove.class
        input
        file
        exemple1.class
        shell.class
        shell.java
        exemple2.class
        exemple1.java
        dossier
        dont.java

Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
8
saisir le nom de votre fichier:
file
        ERREUR: le fichier existe
```

```
Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
3
saisir le fichier
file
          Deleted the file: file
```

```
Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
6
saisir votre fichier source
input
sasir votre dossier/fichier de destination
dossier/unAutreNom
         succés
```

```
Choisir une commande :
1- ls
2- pwd
3- rm
4- cat
5- cp
6- mv
7- mkdir
8- touch
99- exit
5
saisir le fichier a copier
dont.java
saisir le fichier de destination
dossier/fichierCopié
```



Le code est rattaché dans l'email.
Remarque: chmod dépends du système de gestion des fichiers et ainsi dans Windows elle ne marche pas correctement

3- Serialization:

Sérialisation permet de transmettre des objets java entre des JVM différentes (par internet ou pour les stocker ou d'autres méthodes de transfert)

Une classe peut implémenter Serializable (ou externalizable)
ObjectOutputStream.writeObject() sérialise un objet
ObjectInputStream.readObject() désérialiser un objet
Le mot clé transient devant une variable ou méthodes ne transmet pas cette dernière durant la serialization

exemple:

Personne.java

```java
import java.io.Serializable;
public class Personne implements Serializable{
    private String nom = "";
    private String prenom = "";
    private int taille = 0;
    public Personne(){};
    public Personne(String nom, String prenom, int taille) {
        this.nom = nom;
        this.taille = taille;
        this.prenom = prenom;
    }
    public String getNom() {
        return nom;
    }
    public void setNom(String nom) {
        this.nom = nom;
    }
    public int getTaille() {
        return taille;
    }
    public void setTaille(int taille) {
        this.taille = taille;
    }
    public String getPrenom() {
        return prenom;
    }
    public void setPrenom(String prenom) {
        this.prenom = prenom;
    }
}
```

Etudiant.java

```java
public class Etudiant extends Personne {
    public int id;
    public Etudiant(String nom, String prenom, int taille,int id) {
        super(nom, prenom, taille) ;
        this.id=id;
    }
}
```

SerializerPersonne.java

```java
import java.io.*;
public class SerializerPersonne {
    public static void main(String argv[]) {
        Etudiant personne = new Etudiant("ahmed", "fehem", 175,3);
        try {
            FileOutputStream fichier = new FileOutputStream("personne.ser");
            ObjectOutputStream oos = new ObjectOutputStream(fichier);
            oos.writeObject(personne);
            oos.flush();
            oos.close();
        }

        catch (java.io.IOException e) {
            e.printStackTrace();
        }
    }
}

```

DeSerializerPersonne.java

```java
import java.io.*;

public class DeSerializerPersonne {
    public static void main(String argv[]) {
        try {
            FileInputStream fichier = new FileInputStream("personne.ser");
            ObjectInputStream ois = new ObjectInputStream(fichier);
            Etudiant personne = (Etudiant) ois.readObject();
            System.out.println("Personne : ");
            System.out.println("nom : " + personne.getNom());
            System.out.println("prenom : " + personne.getPrenom());
            System.out.println("taille : " + personne.getTaille());
            System.out.println("id : " + personne.id);

        } catch (java.io.IOException e) {
            e.printStackTrace();
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

```
~/Desktop/java/app repartie/tp2  javac DeSerializerPersonne.java Etudiant.java Personne.java SerializerPersonne.java
~/Desktop/java/app repartie/tp2  java SerializerPersonne
~/Desktop/java/app repartie/tp2  cat personne.ser
sr Etudiant~ IidxPersonne  ItailleLnomtLjava/lang/String;Lprenomq~xp ahmedtfehem%
~/Desktop/java/app repartie/tp2  java DeSerializerPersonne
Personne :
nom : ahmed
prenom : fehem
taille : 175
id : 3
```