

**Université Sultan Moulay Slimane**

**Beni-Mellal**



**École nationale des sciences appliquées**

**Khouribga**



# COMPTE RENDU :

## TP9 JAVA POO

Filière : Informatique et Ingénierie des données (iid1)

Réalisé par :

- LAHMAMA Fatima-Zahraa
- EL FATHI Zakaria

Encadré par :

- Mr. GHERABI Noredline

## EXERCICE 1 :

Un code java qui permet de saisir des lignes de texte au clavier et de les enregistrer dans un fichier dont le nom est donné en paramètre. La lecture des lignes se poursuit tant que l'utilisateur ne tape pas le mot 'Stop'

```
package tp9;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class ex1 {

    public static void main(String[] args) throws IOException {

        System.out.println("entrer le nom : ");
        Scanner sc=new Scanner(System.in);
        String var= sc.nextLine();//on prend Le nom du fichier
        //creation d'un fichier dans Le repertoire indique
        FileWriter writer =new FileWriter("C:/Users/EliteBook/Desktop/tp9 java
worksheet/"+var);
        BufferedWriter out= new BufferedWriter(writer);// ouverture du fichier
en ecriture

        String n;
        System.out.println("input sth: ");
        n=sc.nextLine();

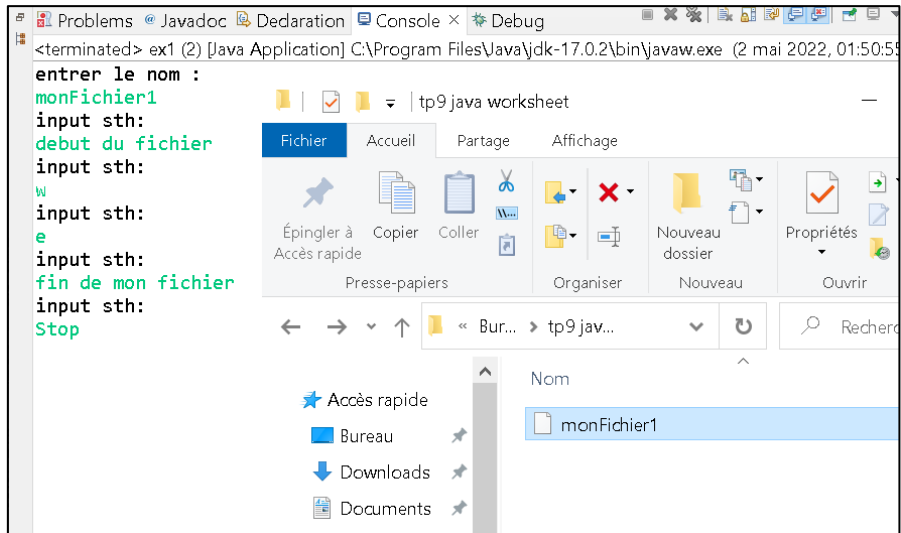
        while(n != "Stop") { //iterer tant que L'entree est differente de
'Stop'
            out.write(n);
            System.out.println("input sth: ");
            n=sc.nextLine();
            out.newLine();
            if (n.equals("Stop")) break;//condition d'arret du programme
        }

        out.close();// fermeture du fichier
    }
}
```

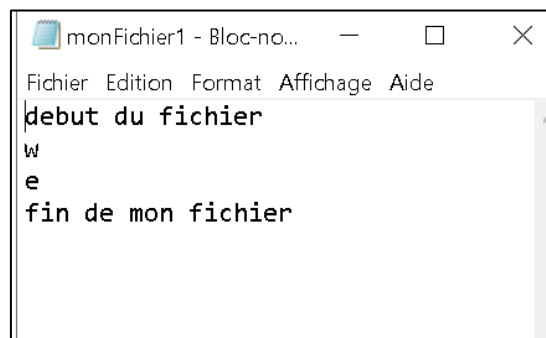
## TRAVAUX PRATIQUES 9

### – JAVA POO

Au moment d'exécution on aperçoit la création d'un fichier dans le répertoire  
« C:/Users/EliteBook/Desktop/tp9 java worksheet » avec le nom entre :



Le fichier contient :



## EXERCICE 2 :

Une classe qui copie un fichier texte source vers un fichier texte destination, selon deux méthodes possibles : caractère par caractère, ou ligne par ligne. Le programme affiche les données et le nombre de lignes du fichier source.

D'abord on crée la classe avec son constructeur qui initialise les attributs :

```
public class ex2 {
    String source, destination;
    // constructeur avec parametres
    public ex2(final String source, final String destination) {
```

## TRAVAUX PRATIQUES 9

### – JAVA POO

```
        this.source=source;
        this.destination=destination;

    }
```

Tout d'abord on définit une méthode d'affichage du contenu et qui, en même temps, calcule le nombre de ligne du fichier comme suit :

```
static public void AffichageContenu(String file) {
    // maintenant on compte le nombre de lignes
    try {
        ArrayList lignes =new ArrayList ();
        BufferedReader br=new BufferedReader(new
        FileReader("C:/Users/EliteBook/Desktop/tp9 java worksheet/output/"+file));
        String ligne;

        int i=0;

        while((ligne=br.readLine())!=null) {
            lignes.add(ligne);
            i++;
        }

        System.out.println("-----le fichier contient "+i+ " lignes son contenu
est -----\\n");
        // affichage du contenu
        Iterator<Integer> it = lignes.iterator();
        while (it.hasNext()) {
            System.out.println(it.next());}
        br.close();// fermeture de lecteur de caracteres
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

On commence par la première méthode (*caractère par caractère*) :

```
//caractere par caractere
static public void caractereParCaractere(String file) throws
FileNotFoundException {
```

TRAVAUX PRATIQUES 9  
– JAVA POO

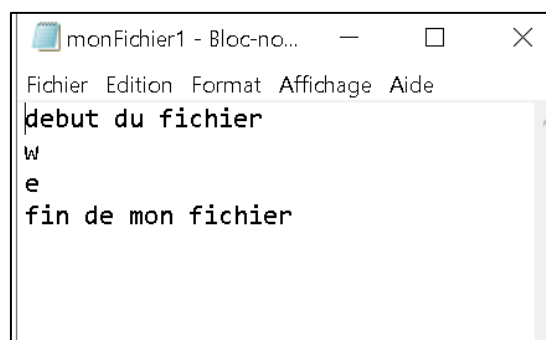
```
        int o;

        FileInputStream input =new
FileInputStream("C:/Users/EliteBook/Desktop/tp9 java worksheet/"+file);
        FileOutputStream out= new
FileOutputStream("C:/Users/EliteBook/Desktop/tp9 java
worksheet/output/"+file);
        try {
            while((o=input.read())!=-1) {
                out.write(o);
            }

            //fermeture des deux fichiers
            input.close();
            out.close();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

        //appel de la fonction d'affichage et compteur
        AffichageContenu(file);
    }
}
```

Et on prend le fichier du premier exercice :



On l'appelle dans la classe principale :

```
package tp9;

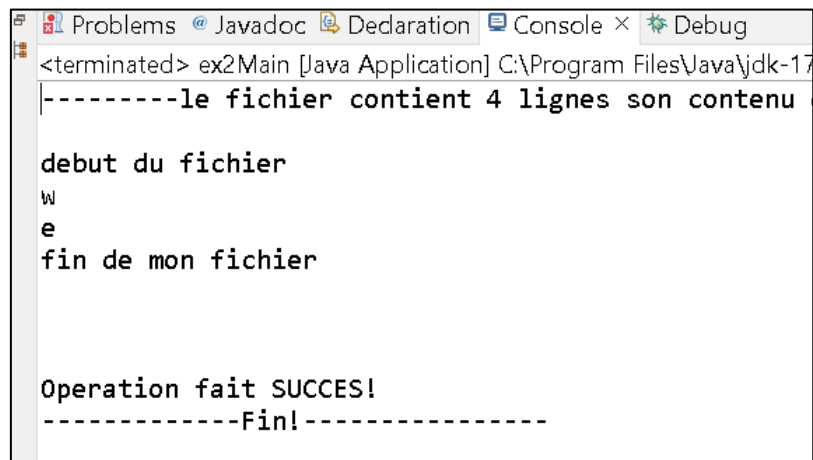
import java.io.FileNotFoundException;
public class ex2Main {
    public static void main(String[] args) {
        try {
```

## TRAVAUX PRATIQUES 9

### – JAVA POO

```
        ex2.caractereParCaractere("monFichier1");
    } catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    System.out.println("\n\nOperation fait SUCCES!\n-----Fin!-----
-----");
}
}
```

Donc dans la console d'exécution on trouve :



The screenshot shows a Java IDE with a console window. The console output is as follows:

```
<terminated> ex2Main [Java Application] C:\Program Files\Java\jdk-17
-----le fichier contient 4 lignes son contenu

debut du fichier
W
e
fin de mon fichier

Operation fait SUCCES!
-----Fin!-----
```

Maintenant on essaie avec la deuxième méthode (**ligne par ligne**) :

```
//ligne par ligne!
static public void ligneParLigne(String file) {
    // initialisation des objets
    BufferedReader br = null;
    PrintWriter pw = null;

    try {
        // ouverture de deux fichiers; un en écriture et l'autre en lecture
        br = new BufferedReader(new
FileReader("C:/Users/EliteBook/Desktop/tp9 java worksheet/"+file));
        pw = new PrintWriter(new
FileWriter("C:/Users/EliteBook/Desktop/tp9 java worksheet/output/"+file));

        String ligne;

        while ((ligne = br.readLine()) != null) {
            pw.println(ligne);
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

## TRAVAUX PRATIQUES 9

### – JAVA POO

```
        }

        br.close();
        pw.close();

    }

    catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }

    //appel de la fonction d'affichage et compteur

    AffichageContenu(file);
}
```

On l'appelle (le même fichier précédant !) dans la classe principale :

```
package tp9;

import java.io.FileNotFoundException;

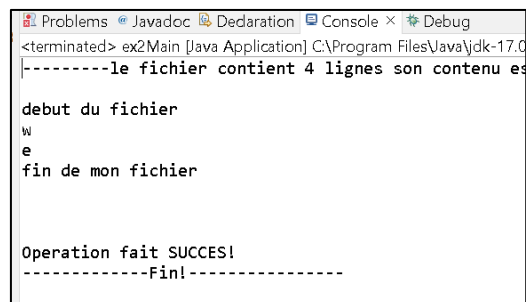
public class ex2Main {

    public static void main(String[] args) {

        ex2.ligneEParLigne("monFichier1");

        System.out.println("\n\nOperation fait SUCCES!\n-----Fin!-----
        -----");
    }
}
```

Cela donne :



```
Problems Javadoc Declaration Console × Debug
<terminated> ex2Main [Java Application] C:\Program Files\Java\jdk-17.0
-----le fichier contient 4 lignes son contenu es

debut du fichier
w
e
fin de mon fichier

Operation fait SUCCES!
-----Fin!-----
```

## EXERCICE 3 :

On définit, d'abord, une classe qui a comme attribut un réel et proposant une méthode de remplissage d'un fichier binaire :

```
public class ex3 {  
    float valeur;  
    //constructeur avec parametre  
    public ex3(float V) {  
        this.valeur=V;  
    }  
  
    @Override  
    public String toString() {  
        return "ex3 [valeur=" + valeur + "];"  
    }  
  
    //methode de remplissage  
    static public void remplir(float d,String file) {  
        DataOutputStream out;  
  
        try {  
            out = new DataOutputStream(  
new FileOutputStream("C:/Users/EliteBook/Desktop/tp9 java worksheet/output/"+file));  
  
                for (int i = 1; i < 7; i++) {  
                    out.writeFloat( (float)(i*d));  
                }  
  
                out.close();  
            } catch (IOException e) {  
                e.printStackTrace();  
            }  
        }  
    }
```

Après dans la même classe on cherche à concevoir une autre méthode qui analyse ce type de fichiers et qui calcule le nombre de lignes(on a utilisé une autre méthode qui calcule le nombre de lignes séparément )



## TRAVAUX PRATIQUES 9

### – JAVA POO

```
//methode d'analyse
static public void analyser(String file) throws IOException {
    float ch;
    int i=1;
    DataInputStream Input;
    try {
        Input=new DataInputStream(
new FileInputStream("C:/Users/EliteBook/Desktop/tp9 java worksheet/output/"+file) );
        // Le contenu
        while ( (ch=Input.readFloat())!=0) {
            // System.out.println(ch);
            System.out.println(i+",\t"+ch);
            i++;
        }

        Input.close();
    } catch (IOException e) {
        // e.printStackTrace();
    }
}

static public int nbrLignes(String file) {

    int i=0;
    DataInputStream Input;
    try {
        Input=new DataInputStream(
new FileInputStream("C:/Users/EliteBook/Desktop/tp9 java worksheet/output/"+file) );

        float ch;
        while ( (ch=Input.readFloat())!=0) {
            i++;
        }

    } catch (IOException e) {
        // e.printStackTrace();
    }
    return i;
}
}
```

Après on fait un appel dans la méthode main pour tester l'ensemble des méthodes :

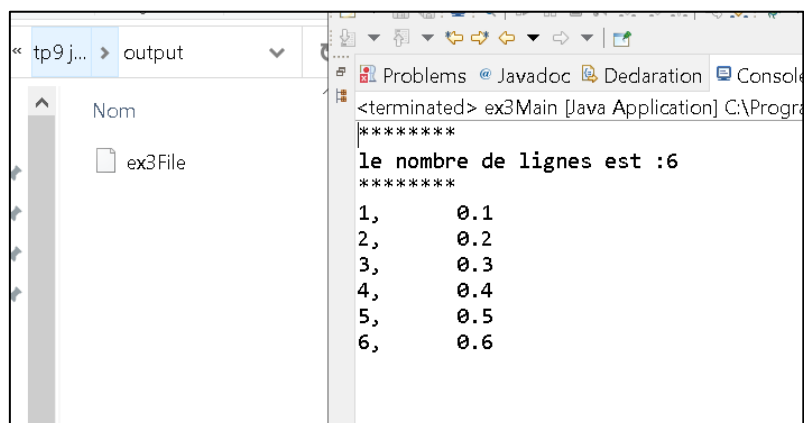
```
package tp9;

import java.io.IOException;
```

## TRAVAUX PRATIQUES 9 – JAVA POO

```
public class ex3Main {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        ex3.remplir(0.1f,"ex3File");  
        System.out.println("*****\nle nombre de lignes est  
:" + ex3.nbrLignes("ex3File") + "\n*****");  
        try {  
            ex3.analyser("ex3File");  
        } catch (IOException e) {  
            // TODO Auto-generated catch block  
            e.printStackTrace();  
        }  
    }  
}
```

A l'exécution cela donne :



## EXERCICE 4 :

Un code JAVA qui permet de copier un fichier texte dans un autre endroit

Code :

```
package tp9;  
  
import java.io.BufferedReader;  
import java.io.FileInputStream;  
import java.io.FileNotFoundException;  
import java.io.FileOutputStream;
```

TRAVAUX PRATIQUES 9  
– JAVA POO

```
import java.io.FileReader;
import java.io.IOException;
import java.util.Scanner;

public class ex4 {

    public static void main(String[] args) {

        System.out.println("entre l'endroit:");
        Scanner sc=new Scanner(System.in);
        String path=sc.nextLine();

        System.out.println("entre le nom du fichier:");
        String nom=sc.nextLine();

        System.out.println("entre le nouveau endroit:");
        String endroit=sc.nextLine();
        // appel de la fonction
        fct(nom,path,endroit);
    }

    static public void fct(String file,String path, String endroit) {
        int o;
        try {

            FileInputStream input =new FileInputStream(path+file);
            FileOutputStream out= new FileOutputStream(endroit+file);

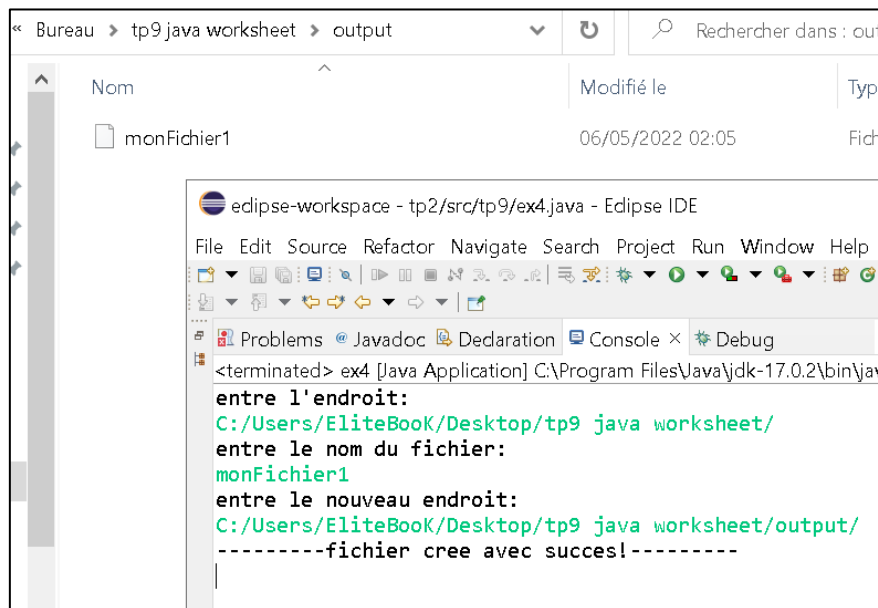
            while((o=input.read())!=-1) {
                out.write(o);
            }

            input.close();
            out.close();
            System.out.println("-----fichier cree avec succes!-----");
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

## TRAVAUX PRATIQUES 9

### – JAVA POO

Exécution :



## EXERCICE 5 :

Une classe « Client » qui implémente l'interface Serializable ayant comme attribut le nom et le prénom et l'adresse du client :

```
package tp9;
import java.io.Serializable;
public class Client implements Serializable {
    private String nom, prenom, adresse;

    //constructeur avec parametres
    public Client(String nom, String prenom, String adresse) {
        super();
        this.nom = nom;
        this.prenom = prenom;
        this.adresse = adresse;
    }

    //getters & setters
    public String getNom() {
        return nom;
    }

    public void setNom(String nom) {
        this.nom = nom;
    }
}
```

## TRAVAUX PRATIQUES 9

### – JAVA POO

```
public String getPrenom() {  
    return prenom;  
}  
  
public void setPrenom(String prenom) {  
    this.prenom = prenom;  
}  
  
public String getAdresse() {  
    return adresse;  
}  
  
public void setAdresse(String adresse) {  
    this.adresse = adresse;  
}  
}
```

Un code java qui sauvegarde 3 clients dans un fichier binaire nommé « file.dot »

Dans la classe principale :

```
static public void sauvegarde(String file) {  
    //code qui initialise 3 clients  
    Scanner sc=new Scanner(System.in);  
    Client C1,C2,C3;  
    System.out.println("Client C1:\n");  
    C1=new Client(sc.nextLine(),sc.nextLine(),sc.nextLine());  
  
    System.out.println("Client C2:\n");  
    C2=new Client(sc.nextLine(),sc.nextLine(),sc.nextLine());  
  
    System.out.println("Client C3:\n");  
    C3=new Client(sc.nextLine(),sc.nextLine(),sc.nextLine());  
    //sauvegarde dans un fichier binaire file.dat  
    String path="C:/Users/EliteBook/Desktop/tp9 java worksheet/output/"+file;  
    ObjectOutputStream outObj;  
    try {  
        outObj=new ObjectOutputStream(new FileOutputStream(path));  
        outObj.writeObject(C1);  
        outObj.writeObject(C2);  
        outObj.writeObject(C3);  
  
        outObj.close();  
    } catch (IOException e) {  
        e.printStackTrace();  
    }  
}
```

## TRAVAUX PRATIQUES 9

### – JAVA POO

Un autre code java qui recharge les noms des clients et les mettre dans un tableau de chaines de caractères, dans la même classe principale :

```
static public void tabNom(String pathFichier) {
    //recharger des noms des clients et les mettre dans un tableau

    ObjectInputStream inputObj=null;
    try {
        try {
            inputObj=new ObjectInputStream(new FileInputStream(pathFichier));
        } catch (FileNotFoundException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
        Client c=null;
        String [] tab = new String[3];
        int i=0;

        try {
            while((c=(Client) inputObj.readObject())!=null) {
                tab[i]=c.getNom();
                System.out.println(tab[i]);
                i++;
            }

            inputObj.close();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            // e.printStackTrace();
        }
    } catch (ClassNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

On fait appel dans la méthode principale pour tester les deux méthodes avec le code suivant :

```
package tp9;

import java.io.FileInputStream;
import java.io.FileNotFoundException;
```

## TRAVAUX PRATIQUES 9

### – JAVA POO

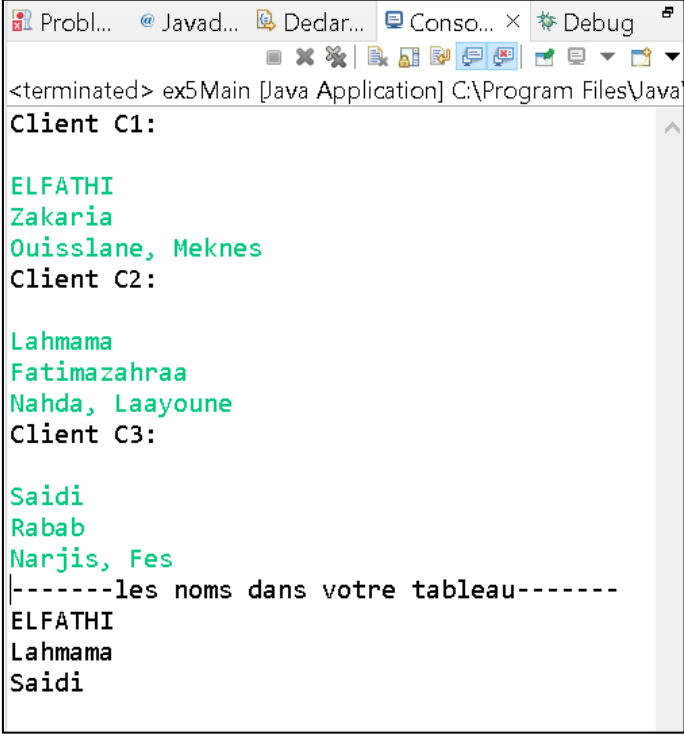
```
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.util.Scanner;

public class ex5Main {

    public static void main(String[] args) {

        sauvegarde("fichier1.dat");
        tabNom("C:/Users/EliteBook/Desktop/tp9 java worksheet/output/fichier1.dat");
    }
}
```

A l'exécution cela donne :



```
<terminated> ex5Main [Java Application] C:\Program Files\Java\
Client C1:

ELFATHI
Zakaria
Ouisslane, Meknes
Client C2:

Lahmama
Fatimazahraa
Nahda, Laayoune
Client C3:

Saïdi
Rabab
Narjis, Fes
|-----les noms dans votre tableau-----
ELFATHI
Lahmama
Saïdi
```

Un code JAVA qui permet de lire les données des deux fichiers simultanément contenant des clients et affichent leurs nom et prénom, en utilisant les Threads, donc on crée une classe :

```
package tp9;

public class ex5Threads extends Thread {
```

## TRAVAUX PRATIQUES 9

### – JAVA POO

```
String f, pathFichier;
public ex5Threads(String F,String pathFile) {
    super();
    pathFichier=pathFile;
    f=F;
}

// La fonction run()
public void run() {
    ex5Main.sauvegarde(f);
    pathFichier+=f;
    ex5Main.tabNom(pathFichier);
}
}
```

Et dans la méthode principale on fait :

```
ex5Threads T1 = new ex5Threads("fich1.dot","C:/Users/EliteBook/Desktop/tp9 java
worksheet/output/");
    ex5Threads T2 = new
ex5Threads("fich2.dot","C:/Users/EliteBook/Desktop/tp9 java worksheet/output/");
    T1.start();
    try {
        T1.join();
    } catch (InterruptedException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
    T2.start();
```

Après exécution :

<pre>&lt;terminated&gt; ex5Main [Java Application] C:\Program Files\Java\jd Client C1: LAHMAMA FATIMAZAHRAA hay Assalam, Laayoune Client C2: ELFATHI ZAKARIA Ouisslane, meknes Client C3: saidi rabab narjis, fes -----les noms dans votre tableau----- LAHMAMA ELFATHI</pre>	<pre>ELFATHI saidi Client C1: Rachidi Maha Qods, Taza Client C2: Maslouhi Chihab Massira 1, Taza Client C3: Rachdi Reda montfleuri 1, fes  -----les noms dans votre tableau----- Rachidi Maslouhi Rachdi</pre>
---	--