Ce premier devoir doit être réalisé en binôme ou bien seul. Il consiste en un simple rapport au format PDF.

Le nom du fichier PDF sera composé à partir du nom de famille (en majuscules) et du prénom (en minuscules) des membres du groupe sous la forme `FAMILLE1_Prénom1=FAMILLE2_Prénom2.pdf`.

Ce document PDF devra être déposé sur la page AMETICE de l'UE[1] *avant* ***le mardi 27 septembre 2022 à minuit****. Les projets en retard ne seront pas acceptés.*

Votre rapport au format PDF comportera une seule page. Il sera rédigé en français et décomposé en 2 parties :

Correction du code de la figure 1

La figure 1 propose une solution *erronée* pour l'exercice I.3 du TD 1. Sans surprise, ce programme lance 10 threads se chargeant d'un dixième de la tâche globale. Ce code s'exécute complètement et termine en affichant une valeur approximative pour $\pi/4$:

```
$ javac PiSurQuatre.java
$ java PiSurQuatre
Nombre de tirages: 5 million(s).
Estimation de Pi/4: 0,365413200
$
```

Vous indiquerez dans votre rapport pourquoi ce code est erroné et proposerez un code correct sans changer le **`main()`**.

Tests de deux modifications du code de la figure 2

La figure 2 s'inspire d'un exemple du cours en le modifiant pour mettre en œuvre une attente passive au lieu d'une attente active. Dans ce programme, le **`main()`** lance un thread **`a`**, effectue ensuite une pause d'une durée aléatoire, entre 0 et 10 secondes, puis réveille le thread **`a`** en modifiant son attribut **`fin`** et en lui envoyant un signal.

Vous obtiendrez en le lançant l'affichage ci-dessous.

```
$ javac Impatient.java
$ java Impatient
Pas trop tôt !
$
```

Dans ce second exercice, il s'agit de constater les erreurs qui apparaissent si l'on omet de placer un bloc **`synchronized`**, soit dans le code du **`main()`** autour de l'instruction **`notify()`**, soit dans le code de la méthode **`run()`** autour de la boucle d'attente passive.

Vous indiquerez précisément dans votre rapport les erreurs apparaissent lorsque l'on omet la prise du verrou intrinsèque via le mot-clef **`synchronized`**, soit dans le **`main()`** soit dans le **`run()`**.

Vous indiquerez également le sens du message d'erreur affiché.

---

1. ou bien, pour les étudiants non inscrits, envoyé attaché à un email destiné à **`remi.morin@univ-amu.fr`** avec pour sujet « **`[M1 POC DM1] <vos noms et prénoms>`** »

```
public class PiSurQuatre extends Thread {
    static long nbTirages = 5_000_000 ;          // Précision du calcul, fixée à 5 000 000
    static long tiragesDansLeDisque = 0 ;        // Nb de tirages dans le disque
    static int nbThreads = 10 ;                  // Nb de threads utilisés

    public static void main (String args[]) {
        System.out.println("Nombre de tirages: " + nbTirages/1_000_000 + " million(s).") ;
        PiSurQuatre[] T = new PiSurQuatre[nbThreads];
        for(int i=0; i<nbThreads; i++){
            T[i] = new PiSurQuatre();
            T[i].start();
        }
        for(int i=0; i<nbThreads; i++){
            try{ T[i].join(); } catch(InterruptedException e){e.printStackTrace();}
        }
        double résultat = (double) tiragesDansLeDisque / nbTirages ;
        System.out.format("Estimation de Pi/4: %.9f %n", résultat) ;
    }

    public void run(){
        Random aléa = new Random();
        double x, y;
        for(long i = 0; i < nbTirages/nbThreads; i++){
            x = aléa.nextDouble() ;
            y = aléa.nextDouble() ;
            if (x * x + y * y <= 1) tiragesDansLeDisque++ ;
        }
    }
}
```

FIGURE 1 – Programme *PiSurQuatre.java* erroné

```
public class Impatient {

  public static void main(String[] args) throws Exception {
        A a = new A();              // Création d'un objet "a" de la classe A
        a.start();                  // Lancement du thread "a"
        Random aléa = new Random();
        Thread.sleep(1000*aléa.nextInt(10));
        a.fin = true;               // Fin du thread "a"
        synchronized(a) {
            a.notify();
        }
    }

    static class A extends Thread {
        public volatile boolean fin = false;
        public void run() {
            synchronized(this) {
                try {
                    while(! fin) wait();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }
            }
            System.out.println("Pas trop tôt !");
        }
    }
}
```

FIGURE 2 – Programme *Impatient.java*