TP7

Calcul de la constante de PI en mono thread

L'outil htop est conseillé car il permet de visualiser la consommation CPU sur chaque coeur de la machine. Pour installer cet outil , il faut lancer la commande suivante en étant root :

apt-get install htop

```
package TD7;
public class ex1 {
	public static void main(String[] args) {
		double a = 0;
		for (long i = 0; i < 500000000; i++) {
			a += Math.pow(-1, i)/(2*i+1);
		}
		System.out.println("PI : "+a);
	}
}
```

Calcul de la constante PI en multi thread

```
package TD7;
public class ex2 extends Thread {
	long start,end;
	double nbrPi = 0;

	public ex2(long start, long end) {
		super();
		this.start = start;
		this.end = end;
	}
	public void run() {
		for(long i = start; i <= end; i++) {
			nbrPi += Math.pow(-1, i)/(2*i+1);
		}
	}
```

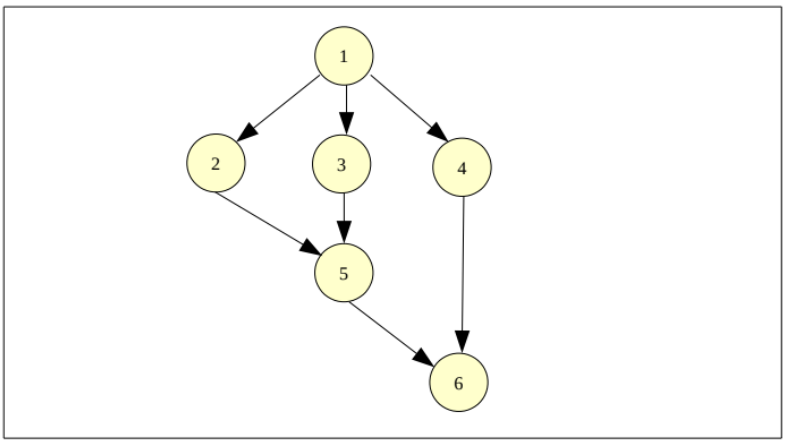
```
public static void main(String[] args) throws InterruptedException {

		long time = System.currentTimeMillis();
		int nbThread = 2;
		long N = 500000000;
		long end = N/nbThread;
		long start = 0;
		double PI = 0;
		// conteneur de tout les threads
		ex2[] threads = new ex2[nbThread];
		//creation des différents thread
		for (int i = 0; i< nbThread; i++) {

			ex2 newThread = new ex2(start,end);
			start = end+1;
			end = start + N/nbThread;
			threads[i] = newThread;
		}
		// activation des threads
		for( int i = 0; i<nbThread; i++) {

			threads[i].start();
		}
		// synchronisation
		for (int i = 0;i <nbThread; i++) {
			threads[i].join();
		}
		//recuperation des données
		for (int i = 0; i<nbThread;i++) {
			PI += threads[i].nbrPi;
		}
		System.out.println("Pi= "+PI);
		System.out.println("time en ms= " + (System.currentTimeMillis()-time));
		}
}
```

Séquencement de threads (version simple)



```
package TD7;

public class ex3 extends Thread{

	int numThread;
	public ex3(int numThread) {
		super();
		this.numThread = numThread;
	}
	public void run() {
		System.out.println("debut du thread "+numThread);
		try {
		sleep(1000*numThread);
		} catch (InterruptedException e) {
			// TODO: handle exception
		}
		System.out.println("fin du thread "+numThread);
	}
	public static void main(String[] args) throws InterruptedException {

		// conteneur de tout les threads
		ex3[] threads = new ex3[6];
		//creation des différents thread
		for (int i = 0; i < 6; i++) {
		ex3 newThread = new ex3(i+1);
		threads[i] = newThread;
		}
		threads[0].start();
		threads[0].join();
		// rien n'est fait tant que le threads[1] n'est pas die
		// cela est du a la fonction join()
		threads[1].start();
		threads[2].start();
		threads[3].start();
		threads[1].join();
		threads[2].join();
		threads[4].start();
		threads[3].join();
		threads[4].join();
		threads[5].start();
	}

}
```