# Programmation multi-cœurs — TD 4
# Conditions de progression

**Exercice 1.** *Conditions de vivacité*. On cherche à implémenter un registre linéarisable avec une méthode `getAndMax` qui écrit l'argument s'il est supérieur à la valeur stockée dans le registre, et retourne la valeur du registre à l'appel de la méthode. Donnez les conditions de progression vérifiées par chacune des implémentations suivantes, parmi : wait-freedom, deadlock-freedom, starvation-freedom et lock-freedom.

***a.*** Pour chaque condition de progression vérifiée, donnez un argument pour le justifier ;

***b.*** Pour chaque condition de progression non vérifiée, donnez un contre-exemple.

```
class Maximizer1 {
  AtomicInteger x = new AtomicInteger(0);
  int getAndMax (int v) {
    while(true) {
      int old = x.get();
      if(x.compareAndSet(old, max(old, v)))
        return old;
    }
  }
}
```

```
class Maximizer2 {
  AtomicInteger x = new AtomicInteger(0);
  int getAndMax (int v) {
    while(true) {
      int old = x.get();
      if(v <= old) return old;
      if(x.compareAndSet(old, v))
        return old;
    }
  }
}
```

```
class Maximizer3 {
  AtomicInteger x = new AtomicInteger(0);
  int getAndMax (int v) {
    while(true) {
      int old = x.get() / 2;
      if(v <= old) return old;
      if(x.compareAndSet(2*old, 2*old+1)){
        x.set(2*v);
        return old;
      }
    }
  }
}
```

```
class Maximizer4 {
  AtomicInteger x = new AtomicInteger(0);
  int getAndMax (int v) {
    while(true) {
      int old = x.get() / 2;
      if(x.compareAndSet(2*old, 2*old+1)){
        x.set(2*max(old, v));
        return old;
      }
    }
  }
}
```

**Exercice 2.** *Pile de Treiber*. Donnez une implémentation linéarisable lock-free de la spécification séquentielle :

```
specification Stack<T> {
  private List<T> list = new LinkedList<T>();

  public void push (T value) {
    list.addFirst(value);
  }

  public T pop () {
    if (list.length() == 0) throw new EmptyException();
    T value = list.getFirst();
    list.removeFirst();
    return value;
  }
}
```