Referencias para la realización del ejercicio 2 apartado b.

Como sabéis el primer apartado de la practica debéis hacer que juegue el usuario por el contario en el apartado 3 debe ser el pc el que juegue de forma automática para ello se debe implementar el algoritmo optimizado.

si recordáis de la teoría la forma en que se recorren las colecciones son distintas. Dentro de las Seq existen dos tipos, LinearSeq a la que pertenecen la lista y IndexedSeq que pertenece Vector y Parvector

Cuando usamos listas se tiene la garantía que ningún elemento de la lista cambia de posición por el contario con parvector no.

Por otro lado, es el compilador el que ejecuta e interpreta el código de parvector y asigna hilos, es algo transparente al usuario, esto lo veréis y entenderéis mejor ejecutando el siguiente código

```
def iota(m:Int): List[Int] = m match {
case 0 => Nil
case m => iota (m-1):::m::Nil
}                                                    //> iota: (m: Int)List[Int]
val l= iota(9)                                       //> l  : List[Int] = List(1, 2, 3, 4, 5, 6, 7, 8, 9)
l.foreach{ e => Thread.sleep(100); print(e) }        //> 123456789
val ll= iota(9).par                                  //> ll  :
                                                       scala.collection.parallel.immutable.ParSeq[Int] =
                                                       ParVector(1, 2, 3, 4
                                                     //| , 5, 6, 7, 8, 9)
ll.foreach{ e => Thread.sleep(100); print(e) }       //> 175384269
```

Con este ejemplo de referencia revisar si se modifican elementos en los tableros, a veces no siempre se producen, y en determinadas operaciones es más fácil o difícil verlo
https://alvinalexander.com/scala/how-to-use-parallel-collections-in-scala-performance/