```
# ฟังก์ชันเอาแค่ tail ไม่เอา Head
def tails(seq: Seq[String]) : Seq[String] = {
        if (seq.isEmpty) {
            println("It's empty List")
        }
        else {
          seq.slice(1, seq.length)
        }
    }
# ฟังก์ชัน drop ตาม input ไปเรื่อยจากหน้า (Head) โดยการตัดเอาHeadออก
def dropping[T](seq: Seq[String], num: Int]: Seq(String] = {
  if (num == 0) seq else dropping(tails(seq), num-1)
# ฟังก์ชัน dropwhile Drop ไปเรื่อยๆ จนถึง "of"
def dropwhile[T](seq: Seq[String]]: Seq[String] = {
  if(seq.lenght == 0) { println("it empty") seq }
  else if (seq.indexOf("of") == 0) { seq }
  else { dropWhile(dropping(seq, 1)) }
```

ค้อนกรรไกรกระดาษ (เป่ายิ้งฉุบ)

```
import scala.util.Random
@main def hello(): Unit =
{   val x = scala.io.StdIn.readInt()
    val y:Int = Random.between(1,4)
    println(x)
    println(y)
  (x,y) match {
    case (1,3)|(2,1)|(3,2) => println("x win")
    case(3,1)|(1,2)|(2,3) => println("y win")
    case (p1,p2) if p1 == p2 => println("tie")
    }
  }
```

Quick Sort:

```
def quickSort(lst: List[Int]): List[Int] = lst match
    case Nil => Nil
    case pivot :: tail =>
        val (smaller, greater) = tail.partition(_ < pivot)
        quickSort(smaller) ::: (pivot :: quickSort(greater))
```

Selection Sort:

```
def SelectionSort(lst: List[Int]): List[Int] = lst match {
    case Nil => Nil
    case _ =>
      val minIndex = lst.indexOf(lst.min)
      val min = lst(minIndex)
      val rest = lst.patch(minIndex, Nil, 1)
      min :: SelectionSort(rest)
```

Try, Option, Either

try catch -
```
def dividebyzero(x: Int, y: Int): Unit = {
        try
          println(x/y)
        catch
          case e: Exception => println("")
def try_strtoint(s: String): Int =
    try
      s.toInt
    catch
      case e: NumberFormatException => 0
```

Option -
```
def dividebyZero(x: Int, y: Int): Option[Int] = {
  if (y == 0) println("") None
  else println(x/y) Some(x/y) }
def option_strtoint(s: String): Option[Int] =
  s.toIntOption match {
      case Some(i) => Some(i)
      case None => None
      }
```

Either -
```
def dividebyZero(x: Int, y: Int): Either[String, Int] =
      if (y == 0) Left("")        error
      else Right(x/y)             ผลลัพธ์
def either_strtoint(str: String): Either[String, Int] =
  try
    Right(str.toInt)
  catch e: NumberFormatException => Left("Error")
```