WiX Engineering

Scala Speed Dating

Amit Anafy

 amitanafy@gmail.com |  @Amita |  linkedin/Amita |  github.com/Amita

How does it work?

Work with a partner

Answer questions creatively

Share your code on Slack

Stop when you hear the tune

Talk about what you wrote

Switch partner

What are our goals?

Code

Open your mind to new ideas

Add some new tools to our Scala tool box

Work with people we haven’t worked with before

What are NOT our goals?

Clean code

Efficient code

Hard questions

How do we start?

1. Split into pairs - choose one computer
2. Join the scala-speed-dating workspace at https://bit.ly/2S2GbIO
3. Join the channel #lambda-world
4. See the first question and start speed dating ☺

Exercise

To join go to: https://bit.ly/2S2GbIO

Write a function that gets a list and a number and return a list with each element multiplied by the number

Input:
List(1,2,3), 2

Output:
List(2,4,6)

Exercise

Input:
List(1,2,3), 2

Output:
List(2,4,6)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def multi(list: List[Int], mul:Int) =
 list.map(_ * mul)
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a sequence and a number return the list with only the element that are smaller than this number

Input:
Seq(1,2,3), 2

Output:
Seq(1)

Exercise

Input:
Seq(1,2,3), 2

Output:
Seq(1)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def smaller(seq:Seq[Int],delim:Int) =
 seq.filter(_<delim)
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Write a function that get a number and return the sum of all the natural number under it that divide by 3 or 5

Input: 10

Output: 23
Numbers divided by 3 or 5: 3,5,6,9

Exercise

Input: 10

Output: 23
Numbers divided by 3 or 5: 3,5,6,9

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def threeFive(num: Int) =
 (1 until num)
  .filter(x=> (x%3==0) || (x%5==0))
  .sum
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a list, and a number (repeateBy) return the list with each element in the list repeate by amount of times

Input:
Seq(1,2,3), 2

Output:
Seq(1,1,2,2,3,3)

Exercise

Input:
Seq(1,2,3), 2

Output:
Seq(1,1,2,2,3,3)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def repeat(seq:Seq[Int], repeater:Int)=
seq.flatMap(Seq.fill(repeater)(_))
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a list return the list with only the odd position numbers (0 based)

Input:
Seq(2,5,3,4,6,7,
9)

Output:
Seq(5,4,7)

Exercise

Input:
Seq(2,5,3,4,6,
7,9)

Output:
Seq(5,4,7)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def oddPos(seq: Seq[Int]) =
 seq
  .zipWithIndex
  .collect {
   case(element, index)
   if index % 2 == 1 => element
  }
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Write a function called ‘unique’ that takes a List and returns the same List, but with duplicated items removed.

Input:
Seq(3,2,7,6,4,7,
3,0,1)

Output:
Seq(3,2,7,6,4,0,1)

Distinct?

Be Creative

Exercise

Input:
Seq(3,2,7,6,4,
7,3,0,1)

Output:
Seq(3,2,7,6,4,0,1)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def unique(list: Seq[Int]): Seq[Int] =
 list
  .foldLeft(Set.empty[Int]) {
   (set, value) =>
   if (set(value)) set else set + value
  }.toSeq
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Write a function that gets a list and return the list with the zeros to the end while keeping the order

Input:
Seq(0,2,4,5,0,77)

Output:
Seq(2,4,5,77,
0,0)

Exercise

Input:
Seq(0,2,4,5,0,
77)

Output:
Seq(2,4,5,77,
0,0)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def moveZeros(list: Seq[Int]) =
 list.filter(_!=0) ++
  Seq.fill(list.count(_==0))(0)
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Write a function that given a number n and return the nth fibonacci number

Input:
4

Output:
3

Exercise

Input:
4

Output:
3

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def fib(num: Int) =
 (1 until num)
  .foldLeft((0, 1)) {
   case ((prev,cur), _) => (cur, prev + cur)
 }._2
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a list of string tuples, return the sum of length of all strings in the list

Input:
Seq(("pickle","rick"),
("morty","jr"))

Output:
17

Exercise

Input:
Seq(("pickle","rick"),
("morty","jr"))

Output:
17

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def sumLength(tuples: Seq[(String, Stri
 tuples
  .map(p => p._1.length + p._2.length)
  .sum
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Write a function that gets a list of int and return the list average

Input:
Seq(3,2,4,6,5,7,8,0,1)

Output:
4.0

Exercise

Input:
Seq(3,2,4,6,5,7,8,0,
1)

Output:
4.0

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def average(seq: Seq[Int]) =
 seq.sum.toDouble/seq.length
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Turn a Sequence of strings to a map from each element to itself

Input:
(“am“,“b“,“c“)

Output:
(”am”->”am”,
”b”->”b”,”c”->”c”)

Exercise

Input:
Seq(“am“,“b“,“c“)

Output:
Seq(”am”->”am”,
”b”->”b”,”c”->”c”)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def makeMap(seq: Seq[String]) =
 seq.map(v => v -> v).toMap
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Write a function that gets a number and checks if it is a prime number

Input:
1483

Output:
true

Exercise

Input:
1483

Output:
true

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def prime(num: Int): Boolean =
 (2 to math.sqrt(num).ceil.toInt)
  .forall(num % _ != 0)
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a list of (String, Int), return a map from the string to the accumulated int.

Input:
Seq(("A",2), ("B",1), ("A", 3))

Output:
Seq(("A",5), ("B",1))

Exercise

Input:
Seq(("A",2), ("B",1),
("A", 3))

Output:
Seq(("A",5),
("B",1))

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def accum(listOfPairs:Seq[(String,Int)]) =
 listOfPairs.groupBy(_._1)
 .mapValues(_.map(_._2).sum)
 .toList
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Write a function that get a string and return if it is a palindrome

Input:
hannah
robigibor

Output:
true
true

Exercise

Input:
hannah
robigibor

Output:
true
true

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def palindrome(st: String) =
 st.zip(st.reverse)
  .take(st.length/2)
  .forall { case (f, s) => f == s }
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Find all substrings

Input:
exp
see

Output:
Set(p,ex,x,xp,e,exp)
Set(e,se,s,ee,see)

Exercise

Input:
exp
see

Output:
Set(p,ex,x,xp,e,exp)
Set(e,se,s,ee,see)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def substrings(s: String) =
(for { start <- 0 to s.length
       end <- start+1 to s.length
     } yield s.substring(start, end))
 .toSet
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a string, reverse each word within a sentence while still preserving whitespace and initial word order.

Input:
reverse me

Output:
esrever em

Exercise

Input:
reverse me

Output:
esrever em

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def reverseEachWord(s: String) =
s.split(' ')
  .map(_.reverse)
  .mkString(" ")
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Generate a strings frequency map

Input:
letters

Output:
Map(e->2, s->1,
t->2, l->1, r->1)

Exercise

Input:
letters

Output:
Map(e->2, s->1,
t->2, l->1, r->1)

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def frequencyMap(s: String) =
s.groupBy(c => c)
  .mapValues(_.length)
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a Seq of AnyRef transform only Mappy typed objects as “mappy”

Input:
Seq(Classy(2),
Mappy(1),
Sassy("Lassie"))

Output:
Seq(Classy(2),
“mappy”,
Sassy(“Lassie”))

Exercise

Input:
Seq(Classy(2),
Mappy(1),
Sassy("Lassie"))

Output:
Seq(Classy(2),
“mappy”,
Sassy(“Lassie”))

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def mappy(seq: Seq[AnyRef]) =
seq.collect {
   case _: Mappy => "mappy"
   case v => v }
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a num, repeatedly add all its digits until the result has only one digit.

Input:
338

Output: 5
3+3+8 = 14
1+4 = 5

Exercise

Input:
338

Output: 5
3+3+8 = 14
1+4 = 5

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def calculateDigit(num: Int): Int =
if (num <10)
  num
else
  calculateDigit(
   num
   .toString
   .map(_.asDigit)
   .sum)}
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a num, and a Seq split the list to two list, such that the first list will have n elements

Input:
2, Seq(1, 2, 4, 325, 23, 52)

Output:
Seq(Seq(1, 2), Seq(4, 325, 23, 52))

Exercise

Input:
2, Seq(1, 2, 4, 325, 23, 52)

Output:
Seq(Seq(1, 2), Seq(4, 325, 23, 52))

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def split(n: Int, l: Seq[Int]) =
Seq(l.take(n), l.drop(n))
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Implement .map on Seq of strings

Input:
Seq("iphone5",
"iphone6"),
x => x + "s"

Output:
Seq("iphone5s",
"iphone6s"),

Exercise

Input:
Seq("iphone5",
"iphone6"),
x => x + "s"

Output:
Seq("iphone5s",
"iphone6s"),

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def map[T](seq: Seq[String],
           fun: String => T) =
 seq.foldLeft(Seq(): Seq[T])(
  (s, v) =>  s :+ fun(v))
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Given a string make sure that parentheses '(', ')' are opened/closed in the correct order

Input:
(((())()))

Output:
true

Exercise

Input:
(()())

Output:
true

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def validate(str: String): Boolean =
validateCharSeq(str.toList, 0)

def validateCharSeq(str: Seq[Char], count: Int): Boolean =
if (count < 0) false
else
  str match {
    case Nil =>
      count == 0
    case head :: tail if head == '(' =>
      validateCharSeq(tail, count + 1)
    case head :: tail if head == ')' =>
      validateCharSeq(tail, count - 1)
    case _ :: tail =>
      validateCharSeq(tail, count)}
```

Exercise

To join go to: https://bit.ly/2S2GbIO

Find a lists median of a non even list

Input:
Seq(1,3,6,5,4)

Output:
4

Exercise

Input:
Seq(1,3,6,5,4)

Output:
4

Possible solution

To join go to: https://bit.ly/2S2GbIO

```
def median(s: Seq[Int]) =
s.sorted.apply(s.length/2)
```

WiX Engineering

Thank You

 Amita@wix.com

 @Amita

 linkedin/Amita

 github.com/Amita

Exercise

Given a sequence of numbers, find it's sub arithmetic sequences with at least 3 numbers

Input:
Seq(1,2,3,4)

Output: 3
Seq((1,2,3),(2,3,4),
(1,2,3,4))

Possible solution

Go Epic!

Exercise

Input:
Seq(1,2,3,4)

Output: 3
Seq((1,2,3),(2,3,4),
(1,2,3,4))

Possible solution

```
def numberOfArithmeticSlices(list: Seq[Int]) = {
for {
  num <- 3 to list.length
  testGroup <- list.sliding(num)
  if testGroup.sliding(2)
     .forall(pair =>
       pair.head - pair(1) ==
             testGroup.head- testGroup(1))
} yield testGroup}
```