Register | Login | Logout | Contact Us

Java-Success.com

Industrial strength Java/JEE Career Companion to open more doors

search here … Go

Home | Java FAQs | 600+ Java Q&As | Career | Tutorials | Member | Why?

Can u Debug? | Java 8 ready? | Top X | Productivity Tools | Judging Experience?

Home › Interview › Scala Interview Q&As › 12: Q78 – Q80 Recursion in Scala Q&As explained with diagrams

12: Q78 – Q80 Recursion in Scala Q&As explained with diagrams

Posted on October 7, 2016 by Arulkumaran Kumaraswamipillai

Q. Do functional languages handle recursion better than non-functional ones?

A. Yes because they have to. A **pure function** is a function with no side effects and no state. Not having side effects means you can't have loop counters as loop counters as they mutate state, hence it will have a side effect. Recursion is not only a good natural match for pure functional programming, but also having no side effects makes recursion more efficient as compiler level optimizations like what order the functions run in, etc can be carried out .

Q78. What will be the output of the following code? Can you explain why you got this output?

600+ Full Stack Java/JEE Interview Q&As ♥Free ♦FAQs

open all | close all

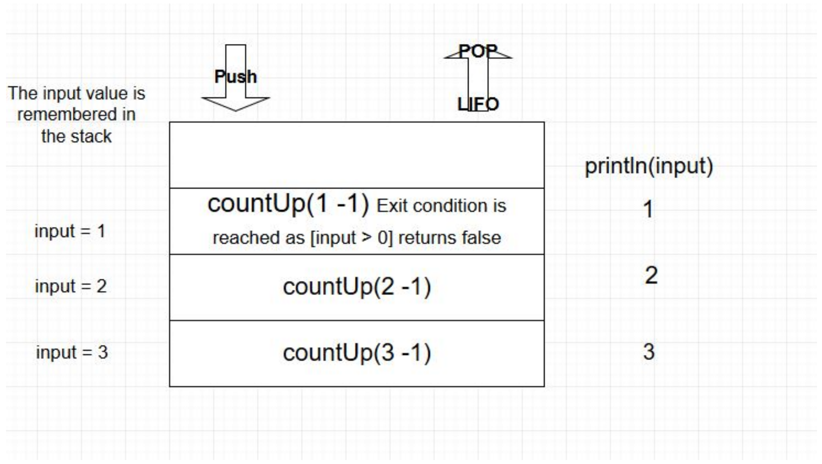
Ice Breaker Interview
Core Java Interview (
JEE Interview Q&A (3
Pressed for time? Jav
SQL, XML, UML, JSC
Hadoop & BigData Int
Java Architecture Inte
Scala Interview Q&As
Scala way of codin
01: Coding Scala
01: ♥ Q1 – Q6 Sca
02: Q6 – Q12 Scala
03: Q13 – Q18 Sca
04: Q19 – Q26 Sca
05: Q27 – Q32 Sca
06: Q33 – Q40 Sca
07: Q41 – Q48 Sca
08: Q49 – Q58 Sca
09: Q59 – Q65 Hig
10: Q66 – Q70 Pat
11: Q71 – Q77 – S

```scala

object Counting extends App {

  def countUp(input: Int): Unit = {
    if(input > 0){
      countUp(input - 1)
      println(input)
    }
  }

  countUp(3)

}

```

A78. The output is:

```

1
2
3

```

The above code makes recursive calls. The recursive call “countUp(input-1)” is made before the “println(input)”. The recursive calls are remembered and gets put into a stack, which is a LIFO (Last In First Out) data structure.


Push
POP
LIFO
The input value is remembered in the stack
println(input)
input = 1
countUp(1 -1) Exit condition is reached as [input > 0] returns false
1
input = 2
countUp(2 -1)
2
input = 3
countUp(3 -1)
3

Scala recursion – counting up

Q79. Can you write code to insert sort a list of nubers say 5, 4, 3 in ascending order using recursion?

A79.

12: Q78 – Q80 Rec
⊞ Spring, Hibernate, & I
⊞ Testing & Profiling/Sa
⊞ Other Interview Q&A f
⊞ ▶ Free Java Interviev

16 Technical Key Areas

open all | close all
⊞ Best Practice (6)
⊞ Coding (26)
⊞ Concurrency (6)
⊞ Design Concepts (7)
⊞ Design Patterns (11)
⊞ Exception Handling (3
⊞ Java Debugging (21)
⊞ Judging Experience I
⊞ Low Latency (7)
⊞ Memory Managemen
⊞ Performance (13)
⊞ QoS (8)
⊞ Scalability (4)
⊞ SDLC (6)
⊞ Security (13)
⊞ Transaction Managen

80+ step by step Java Tutorials

open all | close all
⊞ Setting up Tutorial (6)
⊞ Tutorial - Diagnosis (2
⊞ Akka Tutorial (9)
⊞ Core Java Tutorials (2
⊞ Hadoop & Spark Tuto
⊞ JEE Tutorials (19)
⊞ Scala Tutorials (1)

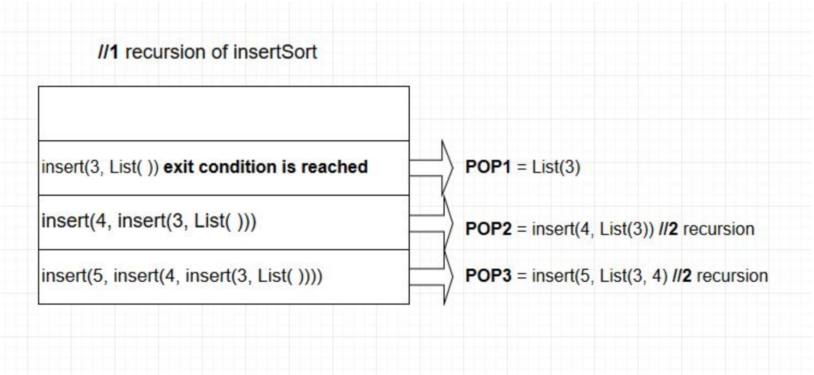
```

object Sorting extends App{

  def insertSort(xs: List[Int]) : List[Int] = {
      xs match {
        case List() => List[Int]()
        case y :: ys => insert(y, insertSort(ys))
      }
  }

  def insert(x: Int, xs: List[Int]): List[Int] =
    xs match {
      case List() => List(x)
      case y :: ys => if (x <= y) x :: xs // pre
                      else y :: insert(x, ys)  /
    }
  }

  val inputList = List(5, 4, 3)
  val result = insertSort(inputList)
  println(result)
}


```

Output:

```

List(3, 4, 5)

```

//1 recursion is evaluated as

```

insert(5, insert(4, insert(3, List())))

```


//1 recursion of insertSort
insert(3, List( )) exit condition is reached
insert(4, insert(3, List( )))
insert(5, insert(4, insert(3, List( ))))
POP1 = List(3)
POP2 = insert(4, List(3)) //2 recursion
POP3 = insert(5, List(3, 4) //2 recursion

Scala Insert Sort

POP2 is insert(4, List(3))

⊞ Spring & HIbernate Tu
⊞ Tools Tutorials (19)
⊞ Other Tutorials (45)

100+ Java pre-interview coding tests

open all | close all
⊞ Can you write code? (
⊞ ♦ Complete the given
⊞ Converting from A to I
⊞ Designing your classe
⊞ Java Data Structures
⊞ Passing the unit tests
⊞ What is wrong with th
⊞ Writing Code Home A
⊞ Written Test Core Jav
⊞ Written Test JEE (1)

How good are your .....?

open all | close all
⊞ Career Making Know-
⊞ Job Hunting & Resum

Recursion 1:

```

x = 4, xs= List(3)
y = 3, ys= List()

```

x <= y is **false**, hence executes “y :: insert(x, ys) which is “3 :: insert(4, List())”

Recursion 2:

```

x = 4, xs = List()

```

Hence <u>exit condition</u> is reached, and returns List(x) which is List(4). So “3 :: List(4)” is an output of **List(3, 4)**.

POP3 is insert(5, List(3, 4))

Recursion 1:

```

x = 5, xs = List(3, 4) //Note that in POP2 4 and
y = 3, ys = List(4)

```

x <= y is **false**, hence executes “y :: insert(x, ys) which is “**3 ::** insert(5, List(4))”

Recursion 2:

```

x = 5, xs = List(4)
y = 4, ys = List()

```

x <= y is **false**, hence executes “y :: insert(x, ys) which is “**4 ::** insert(5, List())”

Recursion 3:

```

x = 5 and y = List() // exit condition is reached

```

Now the final result will be from the

recursion 1 => 3 :: insert(5, List(4))

recursion 2 => 3 :: 4 :: insert(5, List())

recursion 3 => 3 :: 4 :: List(5)

Which gives a final sorted result of **List(3, 4, 5)**

Q80. Can you write code to merge sort a list of nubers say 5, 3, 4 in ascending and descending order using recursion?
A80.

```

object Sorting extends App{

  //outer function
  def mergeSort[T](isTrue: (T, T) => Boolean)(xs

     //inner function
     def merge(xs: List[T], ys: List[T]): List[T
       (xs, ys) match  {
         case (Nil, _) => ys
         case (_, Nil) => xs
         case (x :: xs1, y :: ys1) => if (isTrue
                                      else y :: 
       }
     }// end of inner function


     val n = xs.length / 2
     if (n == 0) xs
     else {
       val split = xs.splitAt(n);
       merge(mergeSort(isTrue)(split._1), mergeS
     }
  } // end of outer function

  val inputList = List(5, 3, 4)

  //curried functions
  val sortAsc = mergeSort ((x: Int, y: Int) => x
  val sortDesc = mergeSort ((x: Int, y: Int) => 

  val resultAsc = sortAsc(inputList);
  val resultDesc = sortDesc(inputList);

  println(resultAsc)
  println(resultDesc)
}
```

Output:

```

List(3, 4, 5)
List(5, 4, 3)

```

//1 recursion is evaluated as

```

merge(merge(isTrue)(5), merge(merge(isTrue)(3), m

```

1) merge(isTrue)(3) => returns **List(3)**

2) merge(isTrue)(4) => returns **List(4)**

3) merge(List(3), List(4)) => x = 3, y = 4, xs = List(), ys = List(). returns **List(3, 4)** after 2 iterations

4) merge(List(5), List(3, 4)) => after 3 iterations returns **List(3, 4, 5)**

Popular Member Posts

♦ 11 Spring boot interview questions & answers

850 views

♦ Q11-Q23: Top 50+ Core on Java OOP Interview Questions & Answers

769 views

001A: ♦ 7+ Java integration styles & patterns interview questions & answers

399 views

18 Java scenarios based interview Questions and Answers

387 views

♦ 7 Java debugging interview questions & answers

308 views

01b: ♦ 13 Spring basics Q8 – Q13 interview questions & answers

305 views

01: ♦ 15 Ice breaker questions asked 90% of the time in Java job interviews with hints

297 views

♦ 10 ERD (Entity-Relationship Diagrams) Interview Questions and Answers

294 views

♦ Q24-Q36: Top 50+ Core on Java classes, interfaces and generics interview questions & answers

246 views

001B: ♦ Java architecture & design concepts interview questions & answers

204 views

Bio | **Latest Posts**

Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

About Arulkumaran Kumaraswamipillai

Mechanical Eng to freelance Java developer in 3 yrs. Contracting since 2003, and attended 150+ Java job interviews, and often got 4 - 7 job offers to choose from. It pays to prepare. So, published Java interview Q&A books via Amazon.com in 2005, and sold 35,000+ copies. Books are outdated and replaced with this subscription based site.**945+** paid members. join my LinkedIn Group. **Reviews**

‹ 11: Q71 – Q77 – Scala type parametrization & variance interview Q&As

01: Coding Scala Way – Part 1 ›

Posted in Scala Interview Q&As

Empowers you to open more doors, and fast-track

Technical Know Hows

☀ Java generics in no time ☀ Top 6 tips to transforming your thinking from OOP to FP ☀ How does a HashMap internally work? What is a hashing function?
☀ 10+ Java String class interview Q&As ☀ Java auto un/boxing benefits & caveats ☀ Top 11 slacknesses that can come back and bite you as an experienced Java developer or architect

Non-Technical Know Hows

☀ 6 Aspects that can motivate you to fast-track your career & go places ☀ Are you reinventing yourself as a Java developer? ☀ 8 tips to safeguard your Java career against offshoring ☀ My top 5 career mistakes

Prepare to succeed

☀ Turn readers of your Java CV go from "Blah blah" to "Wow"? ☀ How to prepare for Java job interviews? ☀ 16 Technical Key Areas ☀ How to choose from multiple Java job offers?

Select Category ▼

© Disclaimer

The contents in this Java-Success are copy righted. The author has the right to correct or enhance the current content without any prior notice.

These are general advice only, and one needs to take his/her own circumstances into consideration. The author will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. No guarantees are made regarding the accuracy or usefulness of content, though I do make an effort to be accurate. Links to external sites do not imply endorsement of the linked-to sites.