## CSPB 3155 - Reckwerdt - Principles of Programming Languages

<u>Dashboard</u> / My courses / <u>2244:CSPB 3155</u> / <u>Week 4: Functors and Operational Semantics</u> / <u>Spot Exam # 1</u>

Started on	Thursday, 13 June 2024, 7:08 PM
State	Finished
Completed on	Thursday, 13 June 2024, 7:22 PM
Time taken	14 mins 16 secs

**Grade 6.97** out of 10.00 (**70**%)

Correct

Mark 2.00 out of 2.00

```
def foo(x: Int): Int = x + 5

def bar(y: String): String = "Hello" + y

def baz(z: String): Int = {
    val v1 = bar(z)
    val v2 = v1.size
    foo(v2)
}
```

Which of the following statements about these functions are true?

Se	lect	one	or	moi	e:

- a. baz takes in an input string and returns the number equaling the size of input string plus 10 💎
- b. Calling bar(foo(z)) will result in a type mismatch error because the output type of foo (Int) is not the same as the input type of bar (String).
- c. Changing the last statement of baz from foo(v2)

  to
  return foo(v2)

does not change the output of this function.

- d. Calling foo(bar(z)) will result in a type mismatch error because the output type of bar (String) is not the same as the input type of foo (Int).
- e. The value v1 in baz will have the type Int

Your answer is correct.

The correct answers are: baz takes in an input string and returns the number equaling the size of input string plus 10, Calling foo(bar(z)) will result in a type mismatch error because the output type of bar (String)

is not the same as the input type of foo (Int)., Calling bar(foo(z)) will result in a type mismatch error because the output type of foo (Int) is not the same as the input type of bar (String)., Changing the last statement of baz from

foo(v2)

to

return foo(v2)

does not change the output of this function.

Correct

Mark 1.00 out of 1.00

What is the type of the following scala function?	
<pre>def foo(x: String) : String= {</pre>	
x + x	
}	
Select one:	
a. Int => Int	
b. String => Int	
c. String	
od. String => String	•
e. Int => String	
Your answer is correct.	
The correct answer is: String => String	

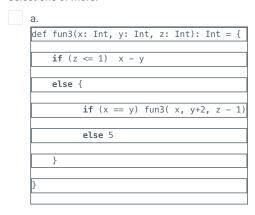
Partially correct

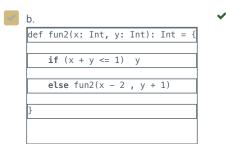
Mark 0.50 out of 3.00

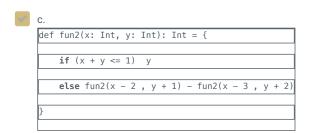
From the scala functions below, select all tail recursive functions. Make sure that all tail recursive functions are selected and no non-tail recursive functions are selected.

×

## Select one or more:







d.
 def fun1(x: Int): Int = {
 if (x <= 1) 1
 else fun1(fun1(10))
}</pre>

e.

```
Your answer is partially correct.

You have correctly selected 1.

The correct answers are:

def fun2(x: Int, y: Int): Int = {

    if (x + y <= 1) y

    else fun2(x - 2 , y + 1)

}

def fun3(x: Int, y: Int, z: Int): Int = {

    if (z <= 1) x - y

    else {

        if (x == y) fun3(x, y+2, z - 1)

        else 5

}
```

Partially correct

Mark 3.47 out of 4.00

Consider the grammar given below. Complete the scala definition by dragging and dropping appropriate choices into the provided gaps.

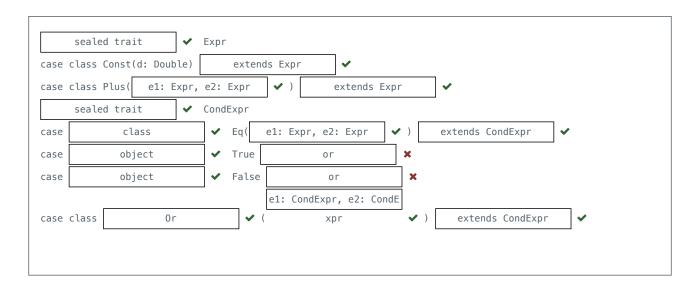
 $\begin{array}{ccc} Expr & \Rightarrow & Const(Double) \\ & | & Plus(Expr, Expr) \end{array}$ 

 $CondExpr \Rightarrow Eq(Expr, Expr)$ 

True

| False

 $| \quad {\rm Or}(CondExpr,CondExpr) \\$ 



extends CondExpr	extends Expr	sealed trait	case class
class	object	e1: Expr, e2: Expr	e1: CondExpr, e2: CondExpr
e1: Expr, e2: CondExpr	and	And	or
Or	not	Not	

Your answer is partially correct.

You have correctly selected 13.

The correct answer is:

Consider the grammar given below. Complete the scala definition by dragging and dropping appropriate choices into the provided gaps.

Expr ⇒ Const(Double)

| Plus(Expr, Expr)

CondExpr ⇒ Eq(Expr, Expr)

| True

| False

| Or(CondExpr, CondExpr)

```
[sealed trait] Expr
case class Const(d: Double) [extends Expr]
case class Plus([e1: Expr, e2: Expr]) [extends Expr]
[sealed trait] CondExpr
case [class] Eq([e1: Expr, e2: Expr]) [extends CondExpr]
case [object] True [extends CondExpr]
case [object] False [extends CondExpr]
case class [Or]([e1: CondExpr, e2: CondExpr]) [extends CondExpr]
```