Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:14:55

Question **1**

Not yet answered

Marked out of 1.00

Flag question

Select the snippets that are valid Haskell code

a.
```
inc : Num a => a -> a
inc a = a + 1
```

b.
```
len l = case l of
  [] -> 0
  (_:xs) -> 1 + len xs
```

c.
```
len l = case l of
  [] -> 0
  (_::xs) -> 1 + len xs
```

d.
```
inc :: Num a => a -> a
inc a = a + 1
```

Next page

◄ L11-Code | Jump to... | L12-Synopsis-Lazy-evaluation ►

Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:14:12

Question **2**

Not yet answered

Marked out of 1.00

Flag question

Given the following function definition:

```
f :: [Int] -> Int
f [1, 2] = 1
f [_, _] = 2
f [3, 4] = 3
```

the result of the following function call is:

```
f [3, 4]
```

Answer:

Previous page

Next page

◄ L11-Code

Jump to...

L12-Synopsis-Lazy-evaluation ►

Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:13:39

Question **3**

Not yet answered

Marked out of 1.00

Flag question

Which function describes best the each of the following list comprehensions?

[take 3 | x <- xs] Choose...

[x | x <- xs, x `div` 3 == 2] Choose...

[x^2 | x <- xs] Choose...

Choose...
take
map
foldl
filter

Previous page

Next page

◄ L11-Code

Jump to...

L12-Synopsis-Lazy-evaluation ►

Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:12:58

Question **4**

Not yet answered

Marked out of 1.00

Flag question

Which of the following are examples of **valid** ways to create local definitions in Haskell?

- [ ] a. y * 2 with y = 5
- [ ] b. local y = 5 in y * 2
- [ ] c. y * 2 where y = 5
- [ ] d. let y = 5 in y * 1

Previous page

Next page

◄ L11-Code

Jump to...

L12-Synopsis-Lazy-evaluation ►

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:12:27

Question **5**

Not yet answered

Marked out of 1.00

Flag question

Select the function that uses pattern guards correctly to implement the filter function:

a.
```
filter _ [] = []
filter p (x:xs) =
  if p x then x:filter p xs
  otherwise filter p xs
```

b.
```
filter _ [] = []
filter p (x:xs)
  | p x = x:filter p xs
  | otherwise = filter p xs
```

c.
```
filter _ [] = []
filter p (x:xs)
  | p x -> x:filter p xs
  | else -> filter p xs
```

Previous page

Next page

◄ L11-Code

Jump to...

L12-Synopsis-Lazy-evaluation ►

Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:11:57

Question **6**

Not yet answered

Marked out of 1.00

Flag question

Select all the **true** statements about the bottom value:

- [x] a. The compiler won't compile (i.e. will show an error) programs that contain the bottom value
- [ ] b. The bottom value can be assigned to any type
- [x] c. In Haskell, undefined is the bottom value
- [ ] d. In Haskell, Nothing is the bottom value

Previous page

Next page

◄ L11-Code

Jump to...

L12-Synopsis-Lazy-evaluation ►

Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:11:41

Question **7**

Not yet answered

Marked out of 1.00

Flag question

The following list comprehension:

```
[(x, y) | x <- ['a', 'b']; y <- [1, 2]]
```

a. Returns [('a',1),('a',2),('b',1),('b',2)]

b. Fails to combile because x and y have different types

c. Fails to combile because the syntax is invalid

d. Returns [('a', 1), ('b', 2)]

Clear my choice

Previous page

Next page

◄ L11-Code

Jump to...

L12-Synopsis-Lazy-evaluation ►

Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:11:16

Question **8**

Not yet answered

Marked out of 1.00

Flag question

Select all the **true** statements about type classes

- [x] a. Type classes are used to abstract common behavior for various types (like Java interfaces)
- [x] b. We can implement type classes defined by the standard library for our own types
- [ ] c. All type class implementations for a data type must be in the module where the data is defined
- [ ] d. Type classes are used to define classes, types that also have methods and private fields

Previous page

Next page

◄ L11-Code

Jump to...

L12-Synopsis-Lazy-evaluation ►

Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:11:02

Question **9**

Not yet answered

Marked out of 1.00

Flag question

Select the correct functions such that the definition of m3 below multiplies 3 numbers wrapped in Maybe

mul3 a b c = a * b * c

m3 a b c = [ ] [ ] a [ ] b [ ] c

<*>
mul3
`fmap`

Previous page

Next page

◄ L11-Code

Jump to...

L12-Synopsis-Lazy-evaluation ►

Functional Programming, Fall, 2022/2023

Quiz navigation

Basic
1 2 3 4 5

Intermediate
6 7 8

Advanced
9 10

Finish attempt ...

Time left 0:10:03

Question **10**

Not yet answered

Marked out of 1.00

Flag question

Given the following code:

```
newtype All = All Bool
instance Semigroup All where
   (All a) <> (All b) = All (a && b)
instance Monoid All where
   mempty = All True
```

The result of the following expressions is:

```
foldl (<>) mempty (map (\x -> All (x `mod` 2 == 0)) [1, 2, 3])
```

Choose...

```
foldl (<>) mempty (map (\x -> All (x >= 2)) [2, 3, 4])
```

Choose...

Choose...
All False
False
All True
True

Previous page

Finish attempt ...

Question **1**

Correct

Mark 1.00 out of 1.00

Flag question

Select the snippets that are valid Haskell code

- [ ] a.
```
inc : Num a => a -> a
inc a = a + 1
```
- [x] b.
```
len l = case l of
  [] -> 0
  (_:xs) -> 1 + len xs
```
- [ ] c.
```
len l = case l of
  [] -> 0
  (_::xs) -> 1 + len xs
```
- [x] d.
```
inc :: Num a => a -> a
inc a = a + 1
```

Your answer is correct.

Question **2**

Incorrect

Mark 0.00 out of 1.00

Flag question

Given the following function definition:

```
f :: [Int] -> Int
f [1, 2] = 1
f [_, _] = 2
f [3, 4] = 3
```

the result of the following function call is:

```
f [3, 4]
```

Answer: 3 ✖

Question **3**

Partially correct

Mark 0.67 out of 1.00

Flag question

Which function describes best the each of the following list comprehensions?

[take 3 | x <- xs]	take ✖
[x | x <- xs, x `div` 3 == 2]	filter ✔
[x^2 | x <- xs]	map ✔

Question **4**

Correct

Mark 1.00 out of 1.00

Flag question

Which of the following are examples of **valid** ways to create local definitions in Haskell?

- [ ] a. y * 2 with y = 5
- [ ] b. local y = 5 in y * 2
- [x] c. y * 2 where y = 5 ✔
- [x] d. let y = 5 in y * 1 ✔

Your answer is correct.

Question **5**

Correct

Mark 1.00 out of 1.00

Flag question

Select the function that uses pattern guards correctly to implement the filter function:

- ( ) a.
```
filter _ [] = []
filter p (x:xs) =
  if p x then x:filter p xs
  otherwise filter p xs
```
- (•) b. ✔
```
filter _ [] = []
filter p (x:xs)
  | p x = x:filter p xs
  | otherwise = filter p xs
```
- ( ) c.
```
filter _ [] = []
filter p (x:xs)
  | p x -> x:filter p xs
  | else -> filter p xs
```

Question **6**

Incorrect

Mark 0.00 out of 1.00

Flag question

Select all the **true** statements about the bottom value:

- [x] a. The compiler won't compile (i.e. will show an error) programs that contain the bottom value ✖
- [ ] b. The bottom value can be assigned to any type
- [x] c. In Haskell, undefined is the bottom value ✔
- [ ] d. In Haskell, Nothing is the bottom value

Your answer is incorrect.

Question **7**

Incorrect

Mark 0.00 out of 1.00

Flag question

The following list comprehension:

```
[(x, y) | x <- ['a', 'b']; y <- [1, 2]]
```

- (•) a. Returns [('a',1),('a',2),('b',1),('b',2)] ✖
- ( ) b. Fails to combile because x and y have different types
- ( ) c. Fails to combile because the syntax is invalid
- ( ) d. Returns [('a', 1), ('b', 2)]

Your answer is incorrect.

Question **8**

Correct

Mark 1.00 out of 1.00

Flag question

Select all the **true** statements about type classes

- [x] a. Type classes are used to abstract common behavior for various types (like Java interfaces) ✔
- [x] b. We can implement type classes defined by the standard library for our own types ✔
- [ ] c. All type class implementations for a data type must be in the module where the data is defined
- [ ] d. Type classes are used to define classes, types that also have methods and private fields

Your answer is correct.

Question **9**

Incorrect

Mark 0.00 out of 1.00

Flag question

Select the correct functions such that the definition of m3 below multiplies 3 numbers wrapped in Maybe

```
mul3 a b c = a * b * c
```

m3 a b c = `fmap` <*> a <*> b <*> c

Your answer is incorrect.

Question **10**

Partially correct

Mark 0.50 out of 1.00

Flag question

Given the following code:

```
newtype All = All Bool
instance Semigroup All where
   (All a) <> (All b) = All (a && b)
instance Monoid All where
   mempty = All True
```

The result of the following expressions is:

```
foldl (<>) mempty (map (\x -> All (x `mod` 2 == 0)) [1, 2, 3])
```

All True ✖

```
foldl (<>) mempty (map (\x -> All (x >= 2)) [2, 3, 4])
```

All True ✔

Your answer is partially correct.

You have correctly selected 1.