Quiz (Week 12)

These questions are general revision for the final exam, not for the content of the Week 11 or 12 lectures.

Question 1

Which statement is correct about the following code snippet?

```
data Length unit = Length Double
data Meter
data Feet
data Yard
addLength :: Length a -> Length a -> Length a
addLength (Length l1) (Length l2) = Length (l1 + l2)
```

1. ◯ The types `Meter`, `Feet` and `Yard` are phantom types
2. ◯ This code will cause a compiler error as the definition of `Meter`, `Feet` and `Yard` are incomplete.
3. ◯ The type `Length` is a phantom type
4. ◯ The type of `addLength` is a phantom type.

Question 2

Choose the correct type/kind annotation.

1. ◯ `Maybe :: *`
2. ◯ `Maybe :: * -> *`
3. ◯ `Maybe :: a -> Just a`
4. ◯ `Maybe :: a -> Maybe a`

Question 3

Given the below definitions:

```
data Nat
 = Z
 | S Nat
```

```
data SNat n where
 Zero :: SNat 'Z
 Succ :: SNat n -> SNat ('S n)
```

Choose the correct type/kind annotation:

1. ◯ `SNat :: Nat -> Nat`
2. ◯ `SNat :: * -> *`
3. ◯ `SNat :: Nat -> *`
4. ◯ None of the above

Question 4

Which statement about this function definition is correct?

```
strange f b n = (f (not b), f (n + 1))
```

1. ◯ This would not type check in Haskell, unless some language extensions are enabled.
2. ◯ This would not type check in Haskell, no matter which language extensions are enabled, because there is no function which can be applied to boolean as well as numeric values.
3. ◯ This would type check in Haskell, no need for language extensions, as `f` could, for example, be the identity function, which can be applied to values of any type.

Question 5

Which one of the following statements is correct? (using the definition of `Vec` and so on from the lecture, exercises)

```
tailV :: Vec (S n) a -> Vec n a
tailV (VCons _ xs) = xs
```

1. ◯ `tailV` is a partial function
2. ◯ `tailV` is a higher order function
3. ◯ `tailV` is a polymorphic function
4. ◯ `tailV` is a type family

Question 6

Select all true statements below.

1. ☐ Property-based testing is more compact than a set of related unit tests.
2. ☐ Repeated property-based testing can improve code coverage.
3. ☐ Property-based testing encourages formal specification of requirements.
4. ☐ Property-based testing helps to prevent bugs through static checking.
5. ☐ Property-based testing helps to prevent bugs through dynamic checking.

Question 7

Select all true statements below.

1. ☐ A partial function is a function that has not been applied to all of its arguments yet.
2. ☐ A partial function is a function that does not return an output for every input in the domain.
3. ☐ A partial function is a function that has not been fully implemented yet.
4. ☐ A partial function is a potential source of bugs, as the type does not fully describe the set of valid inputs to the function.

Question 8

Each of the below is a method to specify and check logical properties of a program. Select all of the methods that rely on dynamic checking. Some methods may additionally rely on static checks, but they should still be selected if they rely on any dynamic checking.

1. ☐ Property-based testing
2. ☐ Proof checking and interactive theorem provers.
3. ☐ Static analysis.
4. ☐ Assertions and Design by Contract.
5. ☐ Type-checking.

Due: Friday, June 1, 11:59:59 pm

Upon clicking submit, you will be prompted for your zID and zPass. Please make sure that your answers are final and that you have answered every question.

If there is a problem, please contact the course administrator.

Submit

You can click here to check if you have submitted already.