How well do you know lists?

Intro 1 2 3 4 5

Before we complete this activity on lists in Erlang, try this quick quiz to test yourself on what you've learned about lists so far.

QUIZ RULES

- Quizzes do not count towards your course score, they are just to help you learn
- You may take as many attempts as you wish to answer each question
- You can skip questions and come back to them later if you wish

Begin quiz

DEFINING THE 'PALINDROME' FUNCTION
VIDEO

SKIP QUIZ
GO TO STEP 2.18

Support

How well do you know lists?

Question 1

Which of the following statements about this Erlang function definition is correct?

```
f([X,Y|Xs]) -> X.
```

- ◯ Calling `f(Ys)` returns the first element of `Ys`.
- ◯ Calling `f(Ys)` returns the second element of `Ys`.
- ◯ Calling `f(Ys)` returns the first element of `Ys`, *if* `Ys` has at least two elements.
- ◯ Calling `f(Ys)` will return a warning because not all the variables in the pattern are used on the right-hand side of the definition.

INTRO SKIP QUESTION

Support

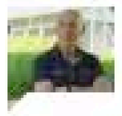
Question 1

Which of the following statements about this Erlang function definition is correct?

```
f([X,Y|Xs]) -> X.
```

- ○ Calling `f(Ys)` returns the first element of `Ys`.
- ○ Calling `f(Ys)` returns the second element of `Ys`.
- [✓ **Calling `f(Ys)` returns the first element of `Ys`, *if* `Ys` has at least two elements.**]
- ○ Calling `f(Ys)` will return a warning because not all the variables in the pattern are used on the right-hand side of the definition.

Correct



Simon Thompson LEAD EDUCATOR

That's right. `X` and `Y` are matched to the first two elements.

Note that the warning (because not all the variables in the pattern are used on the right-hand side of the definition) comes at compile time, not run time.

Support

Question 2

Which of these statements is false?

- ◯ The expression `[3|3]` leads to an error in Erlang.
- ◯ Erlang allows expressions of any type to be arguments to `[ | ]`.
- ◯ `[[3]|[]]` is a list.
- ◯ `[[]|[3]]` is a list.

PREVIOUS QUESTION

SKIP QUESTION

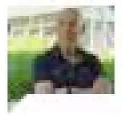
How well do you know lists? - Functional Programming in Erlang - The University of Kent

Question 2

Which of these statements is false?

- [✓ **The expression `[3|3]` leads to an error in Erlang.**]
- Erlang allows expressions of any type to be arguments to `[ | ]`.
- `[[3]|[]]` is a list.
- `[[]|[3]]` is a list.

Correct



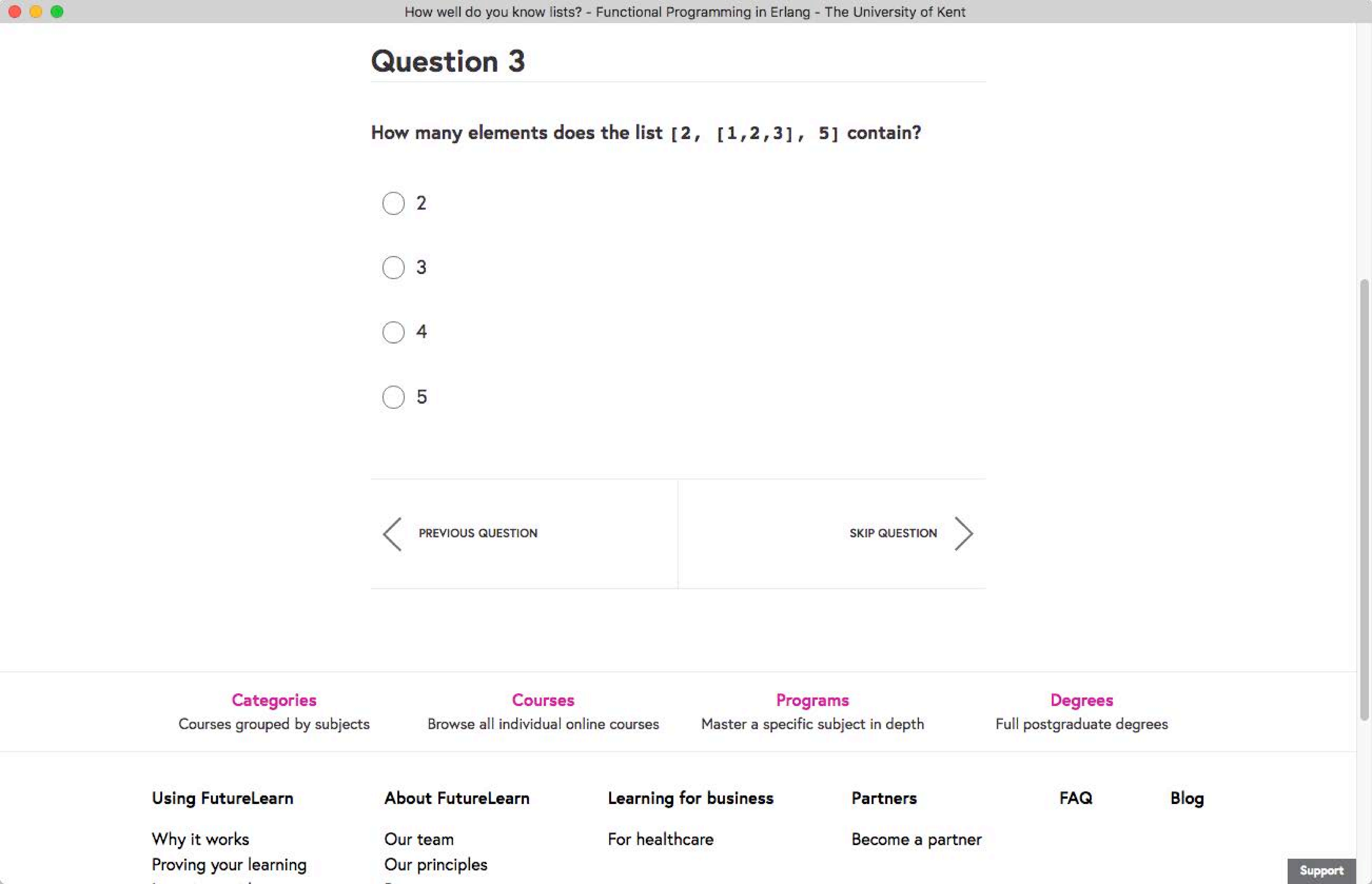
Simon Thompson LEAD EDUCATOR

That's right. It's possible to put *any* values in `[ | ]`, but in general these are called *improper* lists. To make use of lists, the argument on the right-hand side needs to be a (proper) list too. This course has only introduced proper lists, and we'll ignore improper lists from now on.

PREVIOUS QUESTION

NEXT QUESTION

Support



How well do you know lists? - Functional Programming in Erlang - The University of Kent

Question 3

How many elements does the list `[2, [1,2,3], 5]` contain?

○ 2

○ 3

○ 4

○ 5

PREVIOUS QUESTION

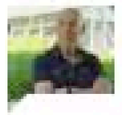
SKIP QUESTION

Question 3

How many elements does the list `[2, [1,2,3], 5]` contain?

- 2
- [✓ 3]
- 4
- 5

Correct



Simon Thompson LEAD EDUCATOR

That's right, there are three elements: 2, the list `[1,2,3]`, and `5`. The point is that the second element is just one element, not three (or two if you ignore duplicates).

PREVIOUS QUESTION

NEXT QUESTION

Support

Question 4

Which of the following statements about this function is false?

```
foo([0|Xs]) -> 1 + foo(Xs);
foo([X|Xs]) -> foo(Xs);
foo([])     -> 0.
```

- ○ Its argument is a list of numbers.
- ○ Its result is an integer.
- ○ It will produce a warning when compiled.
- ○ It is not a tail recursion.

PREVIOUS QUESTION   SKIP QUESTION

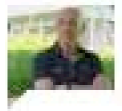
Question 4

Which of the following statements about this function is false?

```
foo([0|Xs]) -> 1 + foo(Xs);
foo([X|Xs]) -> foo(Xs);
foo([])     -> 0.
```

- [✓ **Its argument is a list of numbers.**]
- Its result is an integer.
- It will produce a warning when compiled.
- It is not a tail recursion.

Correct



Simon Thompson LEAD EDUCATOR

That's right. The function can be applied to lists containing *any* values, in fact - it just counts the zeroes that occur; that's Erlang typing for you!

Question 5

Finally, an advanced level question! Which of the following statements about the time complexity of calling `foo(Xs)` is true?

```
foo([X|Xs]) -> bar(foo(Xs),[X]);
foo([])     -> [].

bar([],Ys) -> Ys;
bar([Z|Zs],Ys) -> [Z|bar(Zs,Ys)].
```

- ◯ It is logarithmic in the length of `Xs`.
- ◯ It is linear in the length of `Xs`.
- ◯ It is quadratic in the length of `Xs`.
- ◯ It is exponential in the length of `Xs`.

PREVIOUS QUESTION

CONSOLIDATION: FUNCTIONS OVER LISTS
ARTICLE

Question 5

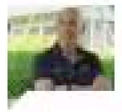
Finally, an advanced level question! Which of the following statements about the time complexity of calling `foo(Xs)` is true?

```
foo([X|Xs]) -> bar(foo(Xs),[X]);
foo([])     -> [].

bar([],Ys) -> Ys;
bar([Z|Zs],Ys) -> [Z|bar(Zs,Ys)].
```

- It is logarithmic in the length of `Xs`.
- It is linear in the length of `Xs`.
- [✓ **It is quadratic in the length of `Xs`.**]
- It is exponential in the length of `Xs`.

Correct



Simon Thompson LEAD EDUCATOR

That's right. `bar` is linear in its first argument, but it is called recursively for each `N` length less than the length of `Xs`, hence the complexity is quadratic.

Support