Exercise 1
○

Property Based Testing
○○○○○○

Proofs and Tests
○○○

Homework
○

Consultations
○

# COMP9141

Curtis Millar

CSE, UNSW (and Dig6

17 June 2020

## Exercise 1

Assignment Project Exam Help

1. **Simple Pic**
https://eduassistpro.github.io/
2. **Moving O**
3. **Generating a Picture**: generate pictures of circle     rclePic

Add WeChat edu_assist_pro

# Property Based Testing

**Key idea**: Generate random input values, and test properties by running them.

**Example (Quic**

```
prop_revers
    reverse (
```

Haskell's *QuickCheck* is the first library ever invented for p
concept has since been ported to Erlang, Scheme, Common-Li
Java, Scala, F#, OCaml, Standard ML, C and C++.

# Mersenne Prime Example

**Example (Demo Task)**

- The $n^{th}$ Mersenne number $M_n = 2^{n-1}$.
- $M_2$, $M_3$,
- **Conjectu**

Let's try using QuickCheck to answer this question.

After a small number of guesses and fractions of a second, QuickCheck
counter-example to this conjecture: 11.

It took humanity about two thousand years to do the same.

# Semigroup and Monoid Properties

Last week we proved by hand that a list forms a semigroup with ++ as its associative operator and a monoid with [] as its identity element.

We can show the same properties much faster (although less completely) with property based testing.

**QuickCheck Pr**

```
-- Semigroup laws
prop_listAssociative xs_yz zs = ((xs ++ ys) ++ zs) == (xs ++ (ys +

-- Monoid laws
prop_listLeftIdentity xs = xs == [] ++ xs
prop_listRightIdentity xs = xs == xs ++ []
```

## Reverse Involution

Last week we also pr                                        *nvolution.*
This took over twe
Let's see how long it

### QuickCheck Property

```
prop_reverseInvolution xs = reverse (reverse xs) =
```

# Ransom Note Example

### Example (Demo Task)

Given a magazine (String form) fr

`canMakeRans`

1. Write a specification
2. Create an efficient implementation
3. Test the implementation

In Haskell.

# Graphics

Write some specifications for the following functions, use them to create properties, and then test an implementation.

1. Horizonta
2. Vertical flip
3. Rotate 180 d

> **Example (Demo Task)**
>
> Implement the above for a single Path. (You might
> for other PictureObject constructors or for an entire                    ice.) In
> Haskell.

## Proofs

Proofs:

- Proofs mus                                                                    f
  the softwar

- Proof comp                                                              cally

- If software is incorrect, a proof attempt might simply bec
  always get constructive negative feedback.

- Proofs can be labour and time intensive ($
  knowledge ($$$).

# Testing

Compared to proofs:

- Tests typically run the actual program, so requires fewer assumptions about the language se
- Test comp specificati
- Incorrect software when tested leads to immediate, de
- Testing is typically cheaper and faster than proving.
- Tests care about efficiency and computability, unlike

We lose some assurance, but gain some convenience ($$$).

## Verification versus Validation

# Assignment Project Exam Help

”https://eduassistpro.github.io/

Testing is essential but is insufficient for safety-critical applic

# Add WeChat edu_assist_pro

Exercise 1
○

Property Based Testing
○○○○○○

Proofs and Tests
○○○

**Homework**
●

Consultations
○

## Homework

Assignment Project Exam Help

1. Last week's
2. The second https://eduassistpro.github.io/
3. This week's quiz is also up, it's due next Friday (in 9 days).

Add WeChat edu_assist_pro

## Consultations

Assignment Project Exam Help

https://eduassistpro.github.io/

Be ready to share your screen with REPL (`ghci` or `stack repl`) and editor set up.

Add WeChat edu_assist_pro