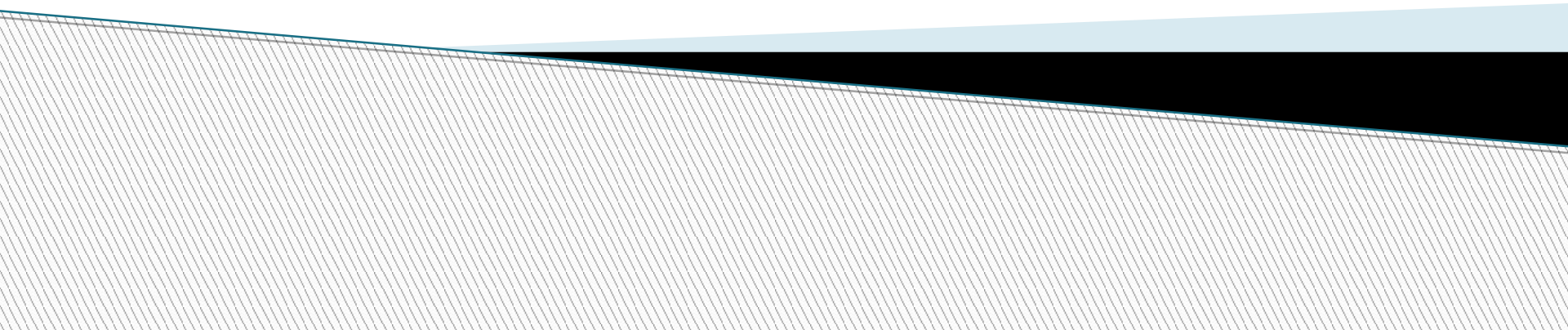


41012 Programming for Mechatronic Systems

Week 3



Overview

- Outstanding questions from last week
- Quiz
- SFS
- We continue looking at classes / scope / access specifiers
- Push into OO methodology with
 - ▢ Inheritance
 - ▢ Polymorphism

EARLY FEEDBACK SURVEY (EFS)

➤ **What is it?**

Confidential, short online survey for each subject in week 4 of semester.

➤ **Why do it?**

Give academic staff an early indication of your learning experience in each subject.

Allow refinements to be made this semester where appropriate and feasible.

➤ **How do I do it?**

Log in at www.sfs.uts.edu.au – available this week only!

➤ **What else do I need to know?**

Please be constructive in your feedback!

By participating, you could win a prize and support a charity.

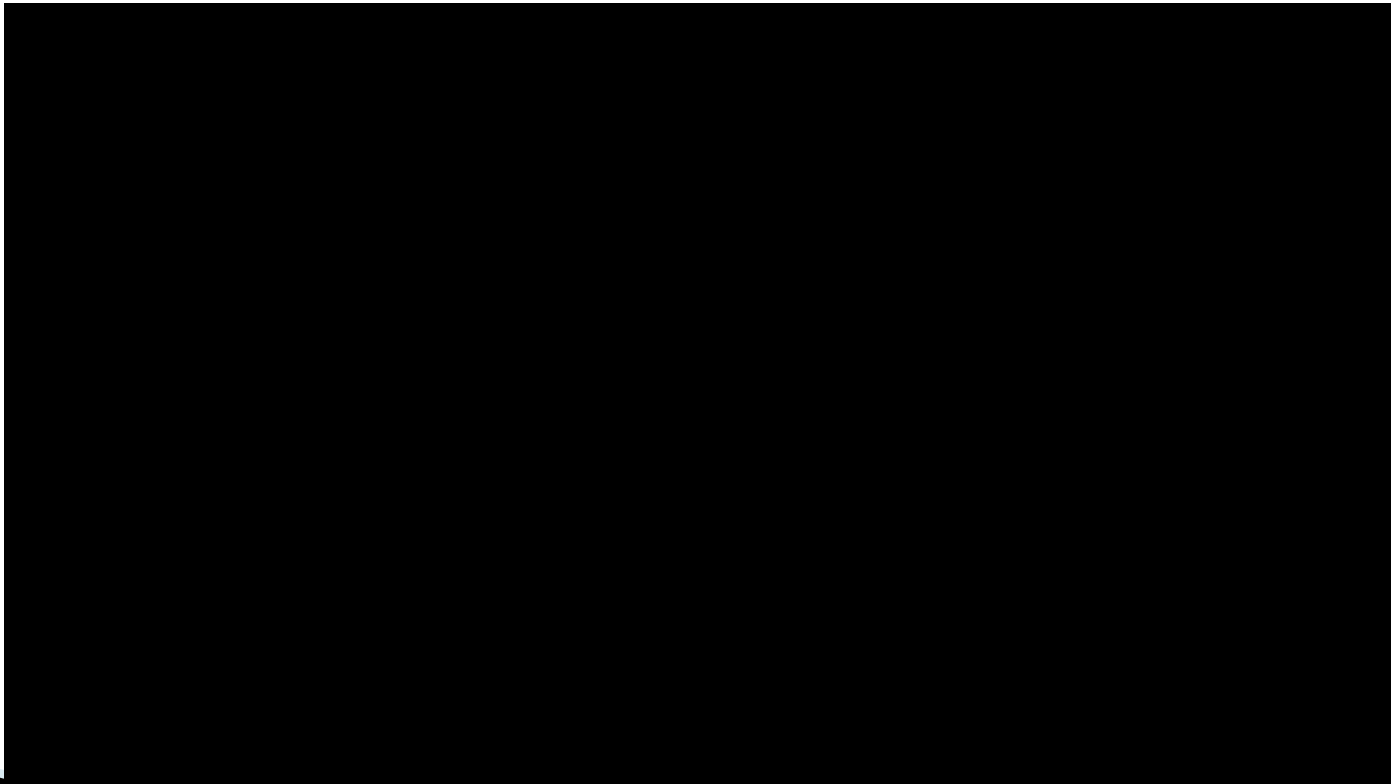
THANK YOU!



EARLY FEEDBACK SURVEY (EFS)

Here is a message from the Vice-Chancellor about why the EFS is important:

<https://youtu.be/GGe1cR01jIM>



Example Class

```
class Rectangle{  
public:  
    Rectangle();  
    void set_values (int w, int h);  
    int area (void);  
    int perimeter(void)  
private:  
    int w, h;  
};
```

Questions:

- ▶ What are the
 - Assumptions
 - Pitfalls
 - What would you change

Polymorphism

- ▶ **polymorphism** : having many forms
- ▶ Why?
- ▶ Occurs when there is a hierarchy of classes and they are related by inheritance
- ▶ A call to a member function will cause a different function to be executed
 - depending on the type of object that invokes the function.

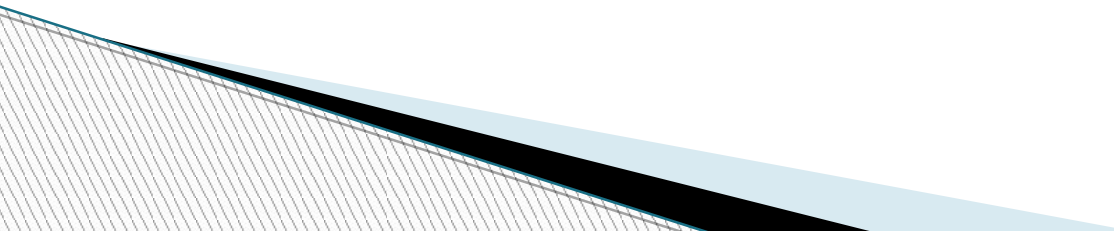
Polymorphism

```
class Rectangle {  
public:  
    Rectangle();  
    void set_values (int w);  
    void set_values (int w, int h);  
    int area (void);  
    int perimeter(void)  
private:  
    int w,h;  
};
```

Questions:

- ▶ What does a single value mean?
 - HINT: Could we assume a square?
- ▶ What might be the pitfall of this class?
 - Is this bad programming practise?

Inheritance

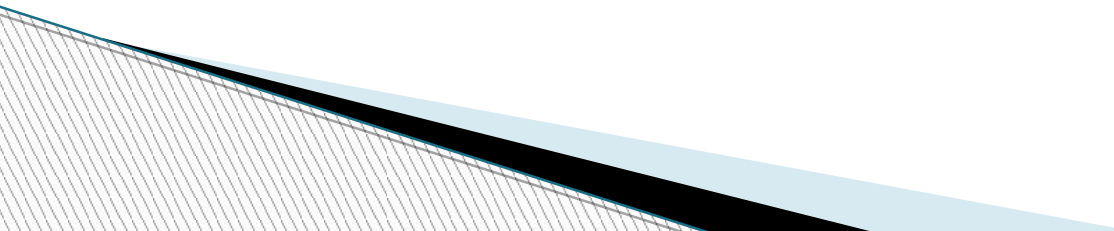
- ▶ Classes in C++ can be extended, creating new classes which retain characteristics of the base class.
 - ▶ This process, known as inheritance, involves a *base class* and a *derived class*
 - ▶ The *derived class* inherits the members of the *base class*, on top of which it can add its own members
 - ▶ Why?
- 

Inheritance (Base Class)

```
class Shape{  
public:  
    Shape();  
    int area (void);  
    int perimeter(void);  
    void printDescription(void);  
private:  
    string description;  
};
```

- ▶ class Rectangle: public Shape
- ▶ Triangle?
 - Can you allow specifying side + height ; 3 sides
- ▶ Circle??

Inheritance

- ▶ **What is inherited from the base class?**
 - ▶ A publicly derived class inherits access to every member of a base class except
 - its constructors and its destructor
 - its assignment operator members (operator=)
 - its friends
 - its private members
- 

Pure Virtual

- ▶ **What is a “pure virtual” member function?**
- ▶ A member function declaration that turns a normal class into an abstract class
- ▶ You only implement it in a derived class.

```
class Shape {
```

```
public:
```

```
virtual void draw() const = 0; // = 0 means it is  
"pure virtual"
```

```
// ...
```

```
};
```

