Starter

Let's

Quiz~~let~~

10:00

your current knowledge…

Open a web browser and type this in the address bar

bitly.im/GIbhU

IB CS

THE BRITISH SCHOOL OF BEIJING, SHUNYI

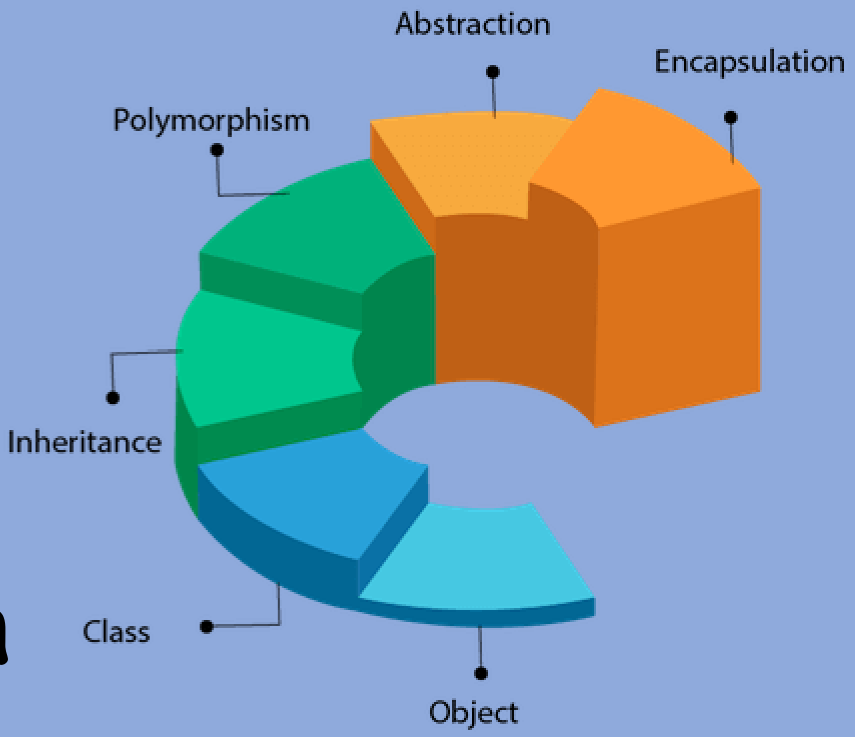
NORD ANGLIA EDUCATION

International Baccalaureate®

OOP Principles

Topic D

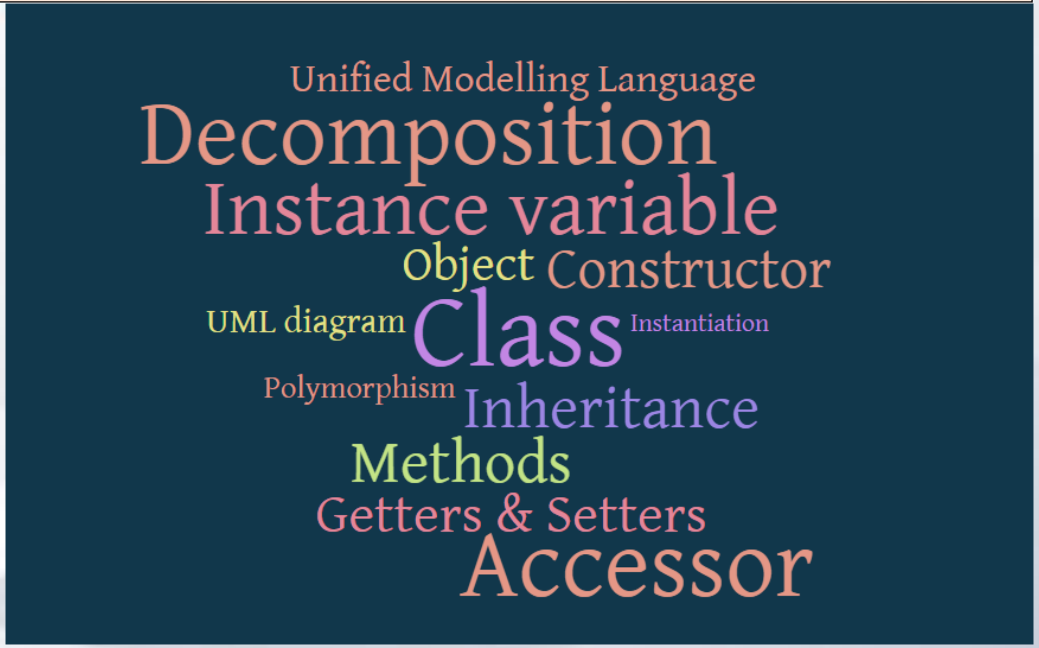
OOP Programming in Java



Mr. Teasdale

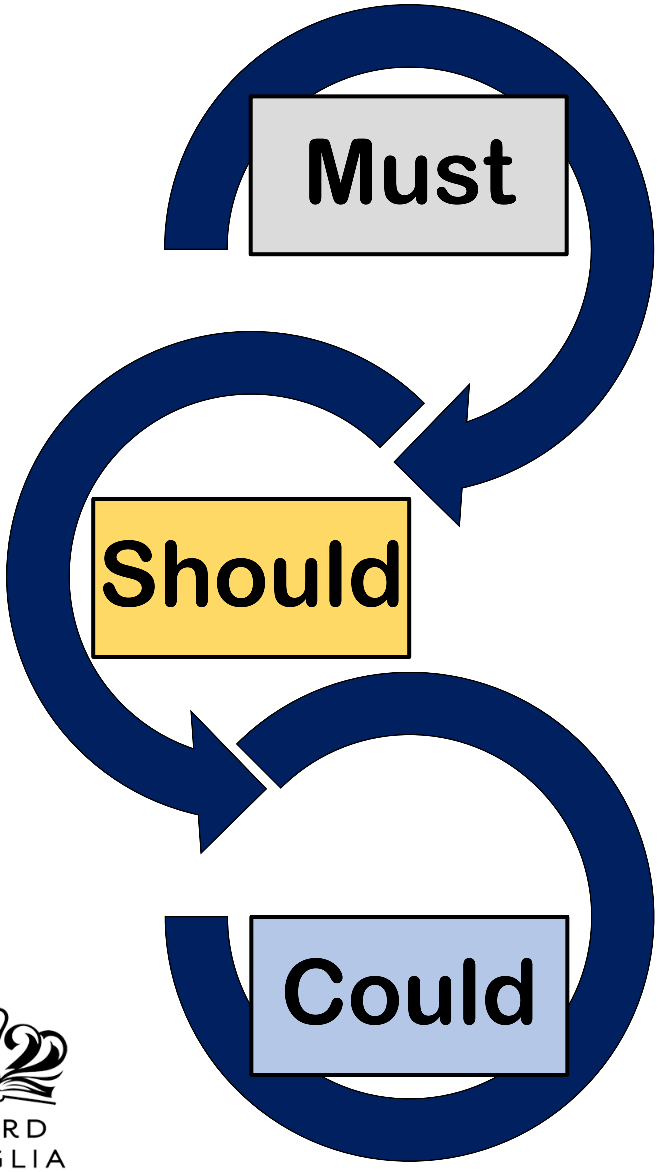
Today we are going to…

Develop an understanding of objects as a programming concept



Success Criteria

- D1.1.1 The general nature of an object



SILVER

Define the terms: object, objects' data and objects' actions

GOLD

Describe the conceptual framework of objects in programming

PLATINUM

Explain the use of objects as an abstract entity

Are there any words you recognise?

UML diagram

Instance variable

Accessor

Instantiation

Class

Inheritance

Getters & Setters

Methods Object

Constructor

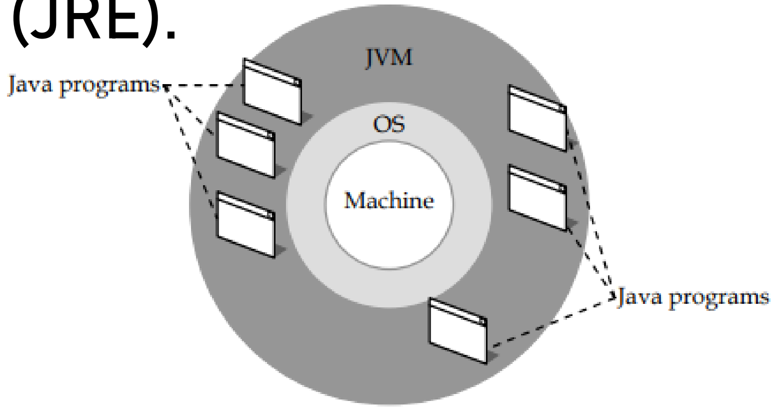
Decomposition

Polymorphism

L/A: To develop an understanding of objects as a programming concept

What is Java?

- Unlike other programming languages, compiling Java source code does not result in a machine language program. Instead, when Java source code is compiled, we get what is called Java bytecode. Java bytecode is a form of machine language instructions. However, it is not primitive to the CPU. Java bytecode runs on a program that mimics itself as a real machine. This program is called the Java Virtual Machine (JVM) or Java Run-time Environment (JRE).



Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
    - ❑ Getter/Setter
    - ❑ Accessor
    - ❑ Constructor

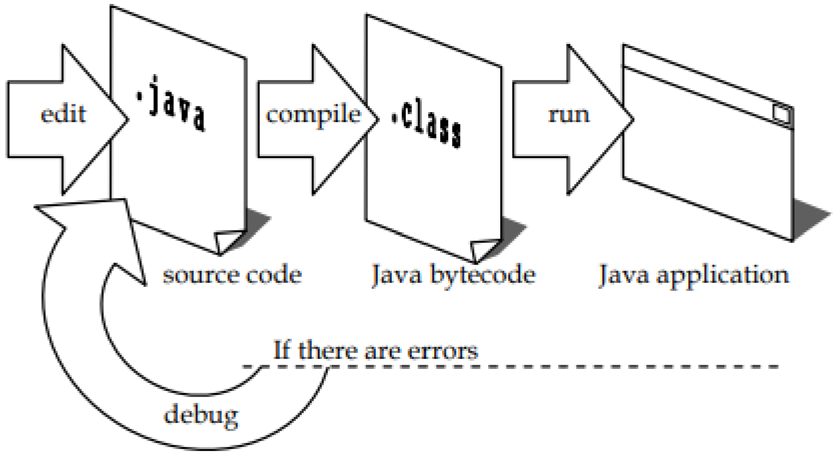
Define the terms: object, objects' data and objects' actions.
Describe the conceptual framework of objects in programming.
Explain the use of objects as an abstract entity.

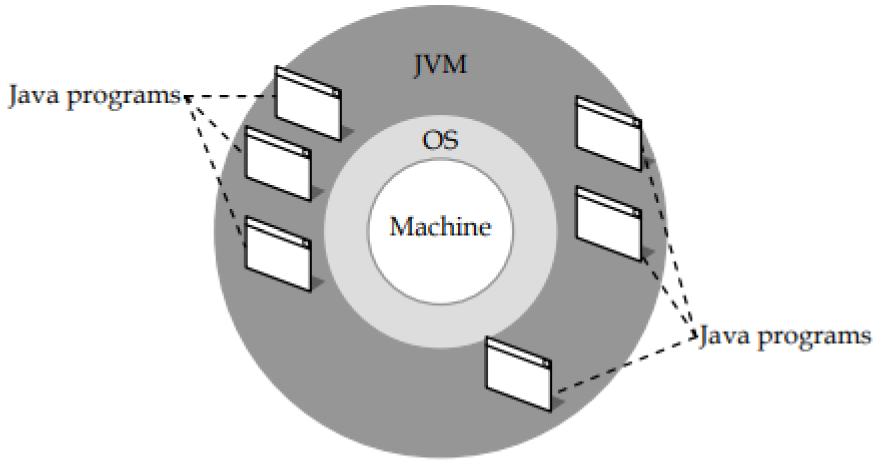
Y12 & 13 IB CS

Myran Teasdale

What is Java good?

- The architecture makes Java bytecode runs on any machines that have JVM, independent of the OSs and CPUs. This means the effort in writing Java source code for a certain program is spent once and the target program can run on any platform. (E.g. Windows, MacOS, Unix, etc.)





Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define the terms: object, objects' data and objects' actions.
Describe the conceptual framework of objects in programming.
Explain the use of objects as an abstract entity.

NORD ANGLIA EDUCATION

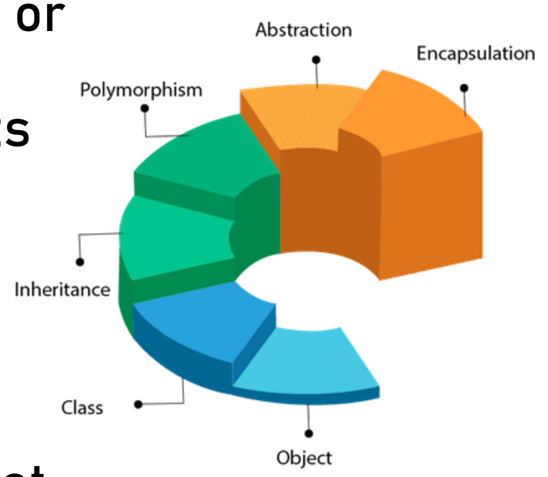
Y12 & 13 IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

What is OOP?

- Procedural programming is about writing procedures or methods that perform operations on the data, while object-oriented programming is about creating objects that contain both data and methods.
- Advantages over procedural programming:
  - OOP is faster and easier to execute
  - OOP provides a clear structure for the programs
  - OOP helps to keep the Java code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug
  - OOP makes it possible to create full reusable applications with less code and shorter development time



Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define the terms: object, objects' data and objects' actions.
Describe the conceptual framework of objects in programming.
Explain the use of objects as an abstract entity.

NORD ANGLIA EDUCATION

Y12 & 13 IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

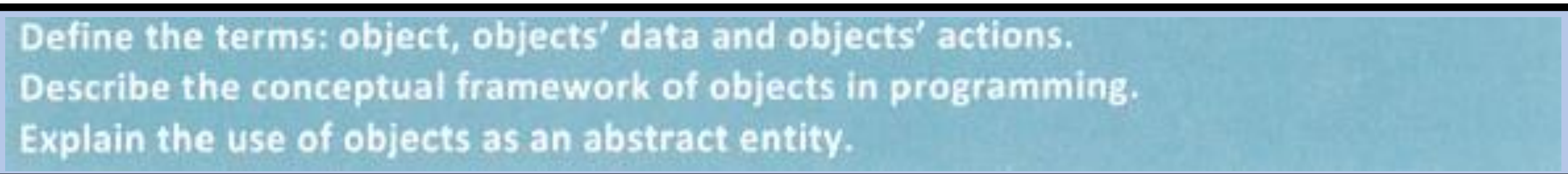
Classes and Objects

- Classes and objects are the two main aspects of object-oriented programming.
- So, a class is a template for objects, and an object is an instance of a class.
- When the individual objects are created, they inherit all the variables and methods from the class.

Vocab Chance!

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor


Define the terms: object, objects' data and objects' actions.
Describe the conceptual framework of objects in programming.
Explain the use of objects as an abstract entity.

Y12 & 13
IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

Objects and classes

class	objects
Fruit	Apple
	Banana
	Mango

Another example:

class	objects
Car	Volvo
	Audi
	Toyota

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
    - ❑ Getter/Setter
    - ❑ Accessor
    - ❑ Constructor

Define the terms: object, objects' data and objects' actions.
Describe the conceptual framework of objects in programming.
Explain the use of objects as an abstract entity.

NORD ANGLIA EDUCATION

Y12 & 13 IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

Example

5:00

```
public class Employee {
private int employeeId;
private String employeeName;
public int getSalary(int basicPay, int da, int hra) {
int salary = basicPay + da + hra;
return salary;
}
}
```

Writing the class

Creating the object

```
Employee employeeObject = new Employee();
```

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define the terms: object, objects' data and objects' actions.
Describe the conceptual framework of objects in programming.
Explain the use of objects as an abstract entity.

Y12 & 13 IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

One more...

5:00

```
public class Cat {
    /*
    Instance variables: states of Cat
     */
    String name;
    int age;
    String color;
    String breed;

    /*
    Instance methods: behaviors of Cat
     */
    void sleep(){
        System.out.println("Sleeping");
    }
    void play(){
        System.out.println("Playing");
    }
    void feed(){
        System.out.println("Eating");
    }

}
```

Writing the class

Creating the objects

```
public class Main {
    public static void main(String[] args) {
        Cat thor = new Cat();
        Cat rambo = new Cat();
    }
}
```

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define the terms: object, objects' data and objects' actions.
Describe the conceptual framework of objects in programming.
Explain the use of objects as an abstract entity.

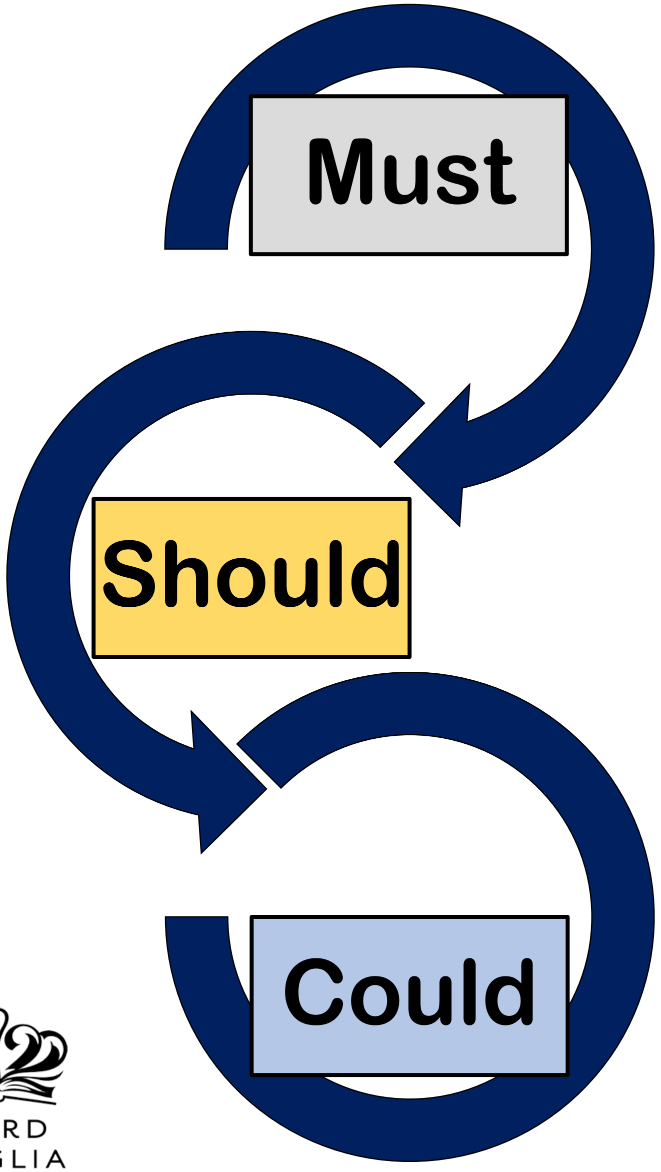
NORD ANGLIA EDUCATION

Y12 & 13 IB CS

Myran Teasdale

Success Criteria

- D.1.2 Distinguishing between object and instantiation



Define the terms: class, template and instantiation

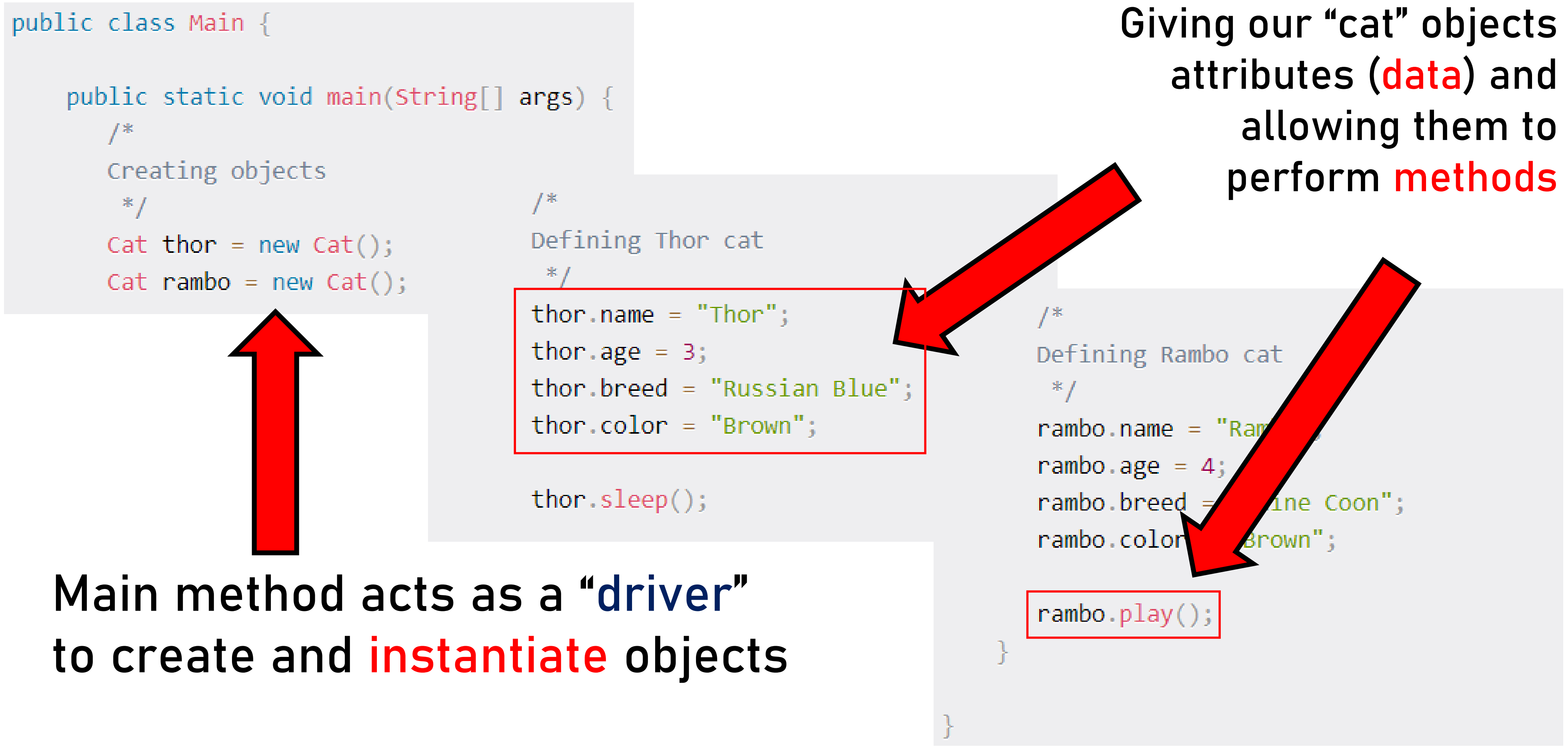
Distinguish between an object and instantiation

Discuss memory use and code definitions that relate to object and instantiation

L/A: To develop an understanding of objects as a programming concept

Giving the objects attributes

5:00



```
public class Main {

    public static void main(String[] args) {
        /*
        Creating objects
         */
        Cat thor = new Cat();
        Cat rambo = new Cat();
```

```
        /*
        Defining Thor cat
         */
        thor.name = "Thor";
        thor.age = 3;
        thor.breed = "Russian Blue";
        thor.color = "Brown";

        thor.sleep();
```

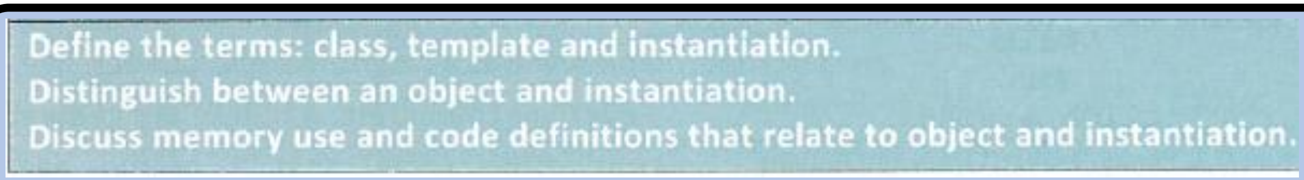
```
        /*
        Defining Rambo cat
         */
        rambo.name = "Ram
        rambo.age = 4;
        rambo.breed =     ine Coon";
        rambo.color     Brown";

        rambo.play();
    }

}
```

Giving our "cat" objects attributes (data) and allowing them to perform methods

Main method acts as a "driver" to create and instantiate objects

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
    - ❑ Getter/Setter
    - ❑ Accessor
    - ❑ Constructor



Define the terms: class, template and instantiation.
Distinguish between an object and instantiation.
Discuss memory use and code definitions that relate to object and instantiation.

Y12 & 13 IB CS

Myran Teasdale

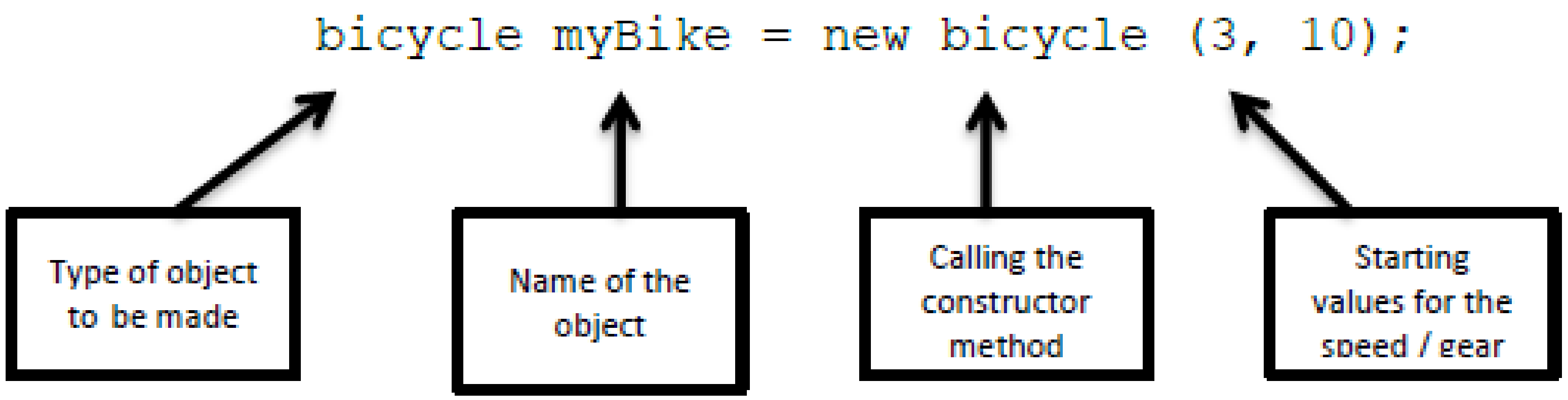
L/A: To develop an understanding of objects as a programming concept

Constructor methods

5:00

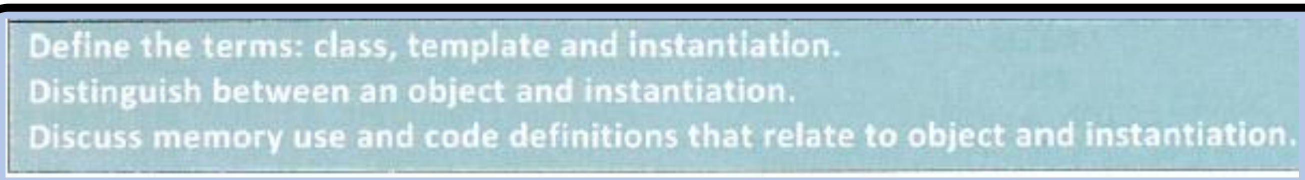
Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
    - ❑ Getter/Setter
    - ❑ Accessor
    - ❑ Constructor



```
bicycle myBike = new bicycle (3, 10);
```

Type of object to be made
Name of the object
Calling the constructor method
Starting values for the speed / gear



Define the terms: class, template and instantiation.
Distinguish between an object and instantiation.
Discuss memory use and code definitions that relate to object and instantiation.

Y12 & 13 IB CS

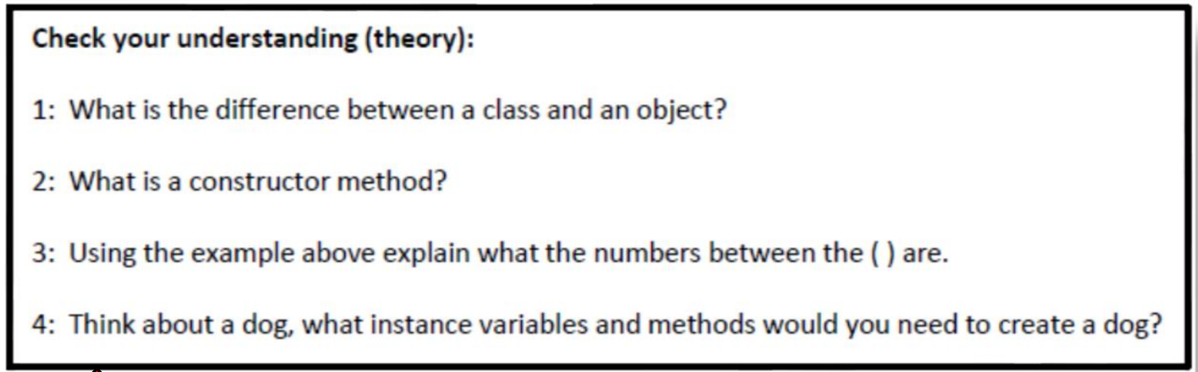
Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

Activity – Recap and Practical

repl.it/@BSBYear12CS/Critters-1

40:00



Check your understanding (theory):

1: What is the difference between a class and an object?

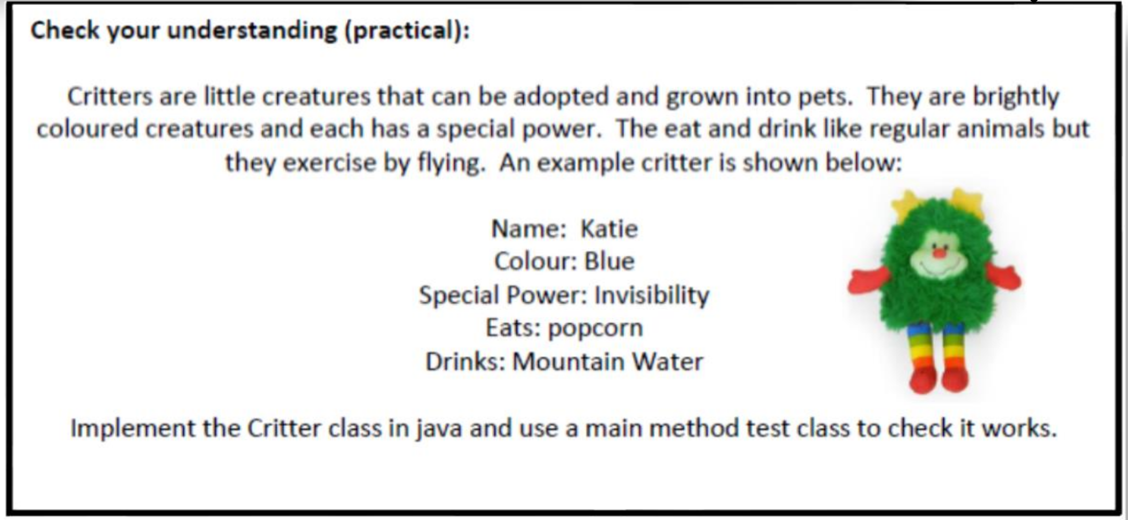
2: What is a constructor method?

3: Using the example above explain what the numbers between the ( ) are.

4: Think about a dog, what instance variables and methods would you need to create a dog?

Answer the theory questions

Then, Attempt the practical task



Check your understanding (practical):

Critters are little creatures that can be adopted and grown into pets. They are brightly coloured creatures and each has a special power. The eat and drink like regular animals but they exercise by flying. An example critter is shown below:

Name: Katie
Colour: Blue
Special Power: Invisibility
Eats: popcorn
Drinks: Mountain Water

Implement the Critter class in java and use a main method test class to check it works.

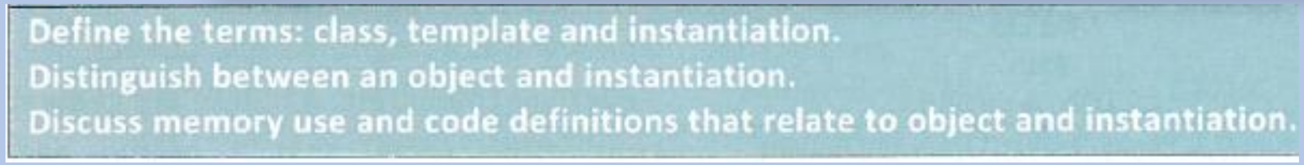
Need help?
<u>THEORY</u>

bitly.im/5yhDJ

<u>PRACTICAL</u>
There is some source code for the practical section in the workbook

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor



Define the terms: class, template and instantiation.
Distinguish between an object and instantiation.
Discuss memory use and code definitions that relate to object and instantiation.

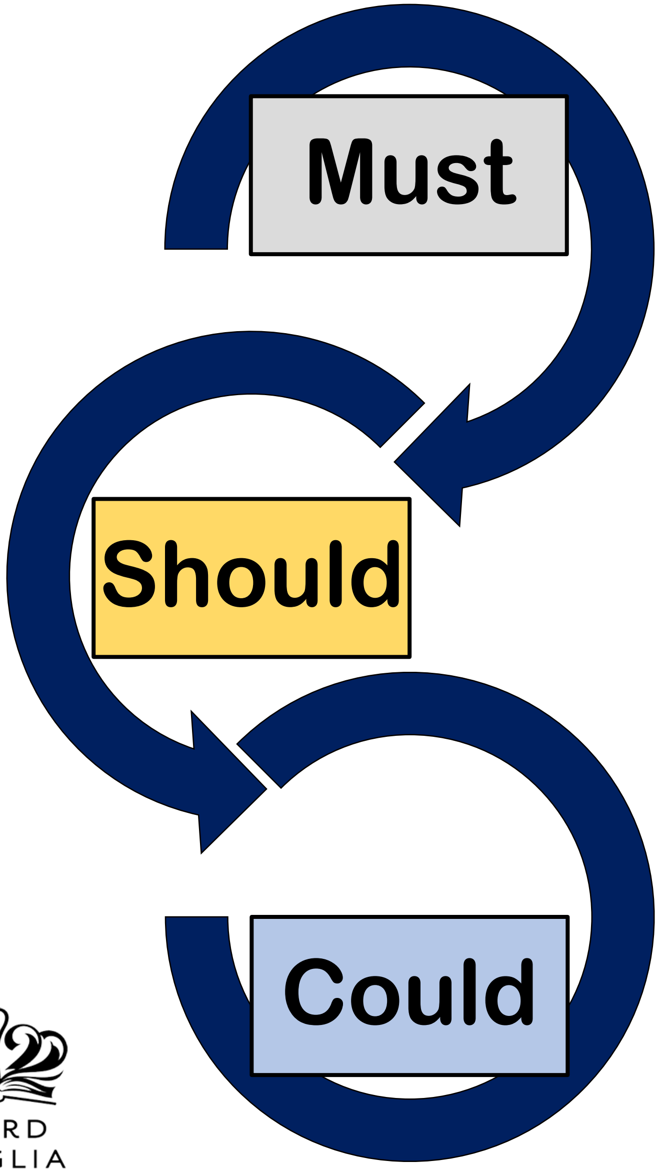
NORD ANGLIA EDUCATION

Y12 & 13 IB CS

Myran Teasdale

Success Criteria

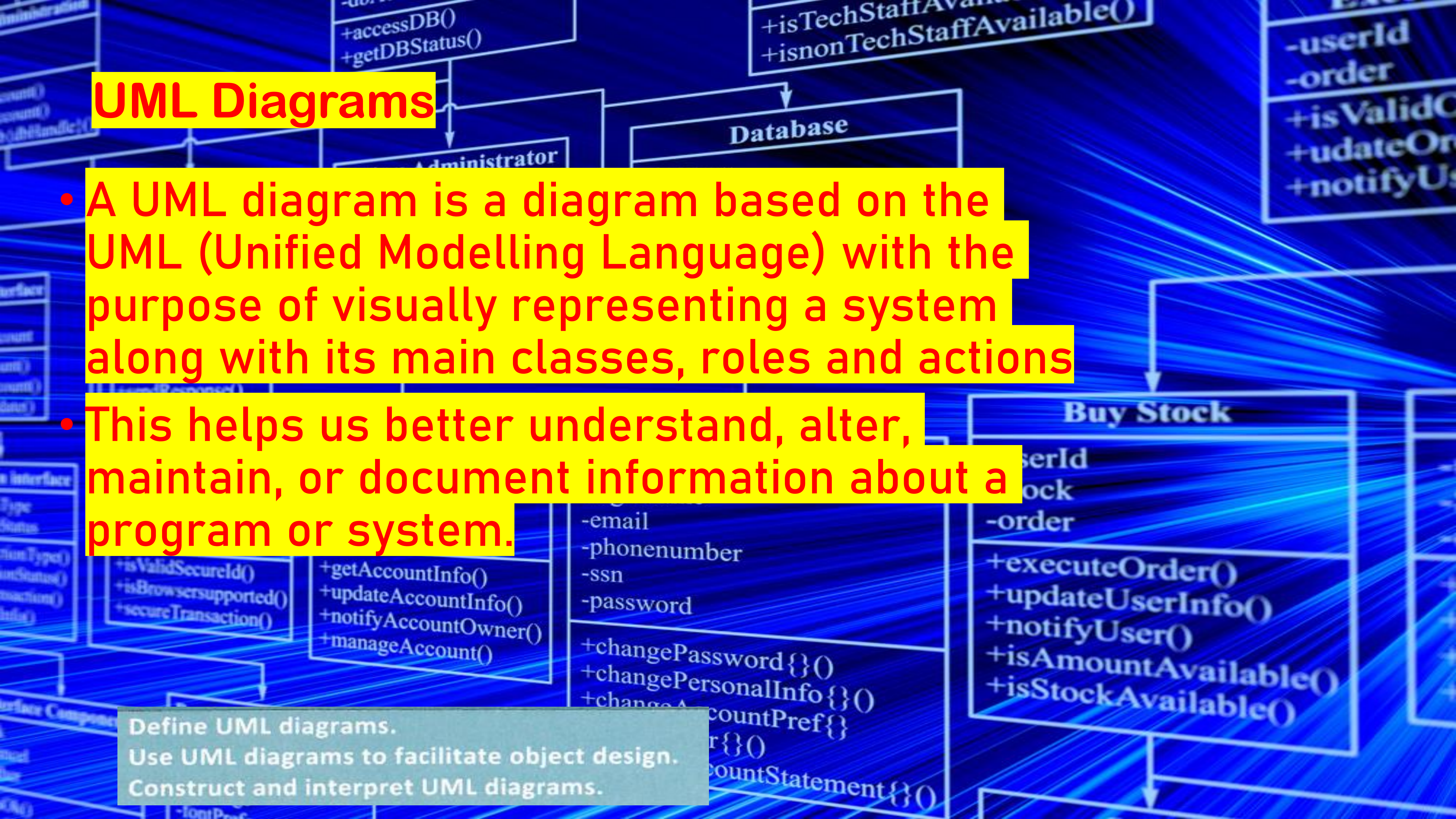
- D.1.3-4 UML Diagrams



Define UML diagrams

Use UML diagrams to facilitate object design

Construct and interpret UML diagrams



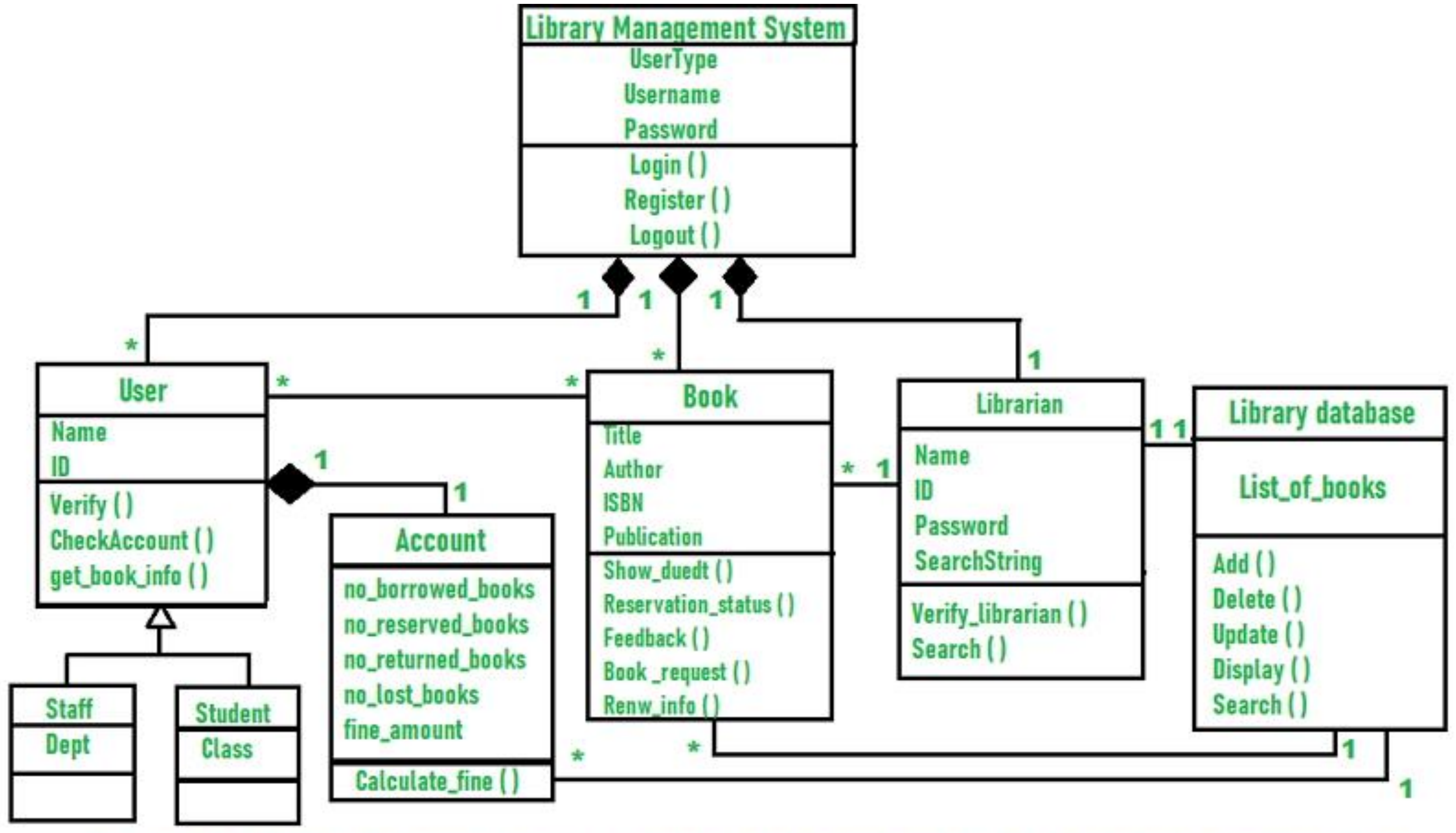
UML Diagrams

- A UML diagram is a diagram based on the UML (Unified Modelling Language) with the purpose of visually representing a system along with its main classes, roles and actions
- This helps us better understand, alter, maintain, or document information about a program or system.

Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

L/A: To develop an understanding of objects as a programming concept

They can get quite complex...



CLASS DIAGRAM FOR LIBRARY MANAGEMENT SYSTEM

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

Y12 & 13
IB CS

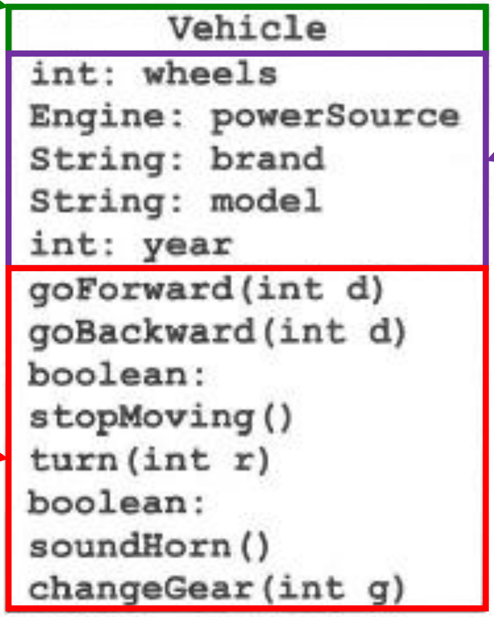
NORD ANGLIA EDUCATION

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

UML Diagrams (Structure)

Class name



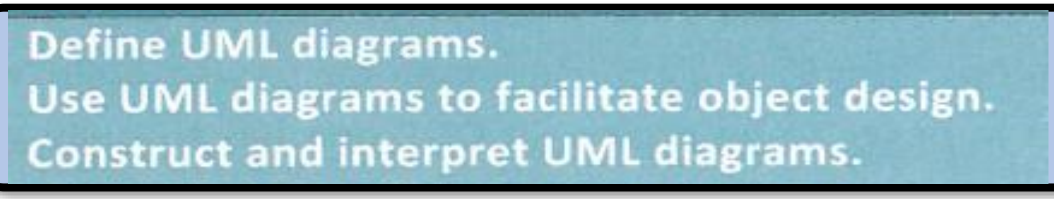
```
Vehicle
int: wheels
Engine: powerSource
String: brand
String: model
int: year
goForward(int d)
goBackward(int d)
boolean:
stopMoving()
turn(int r)
boolean:
soundHorn()
changeGear(int g)
```

The data/attributes that are passed to and from the class

Actions (methods) that the class can "do"

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor



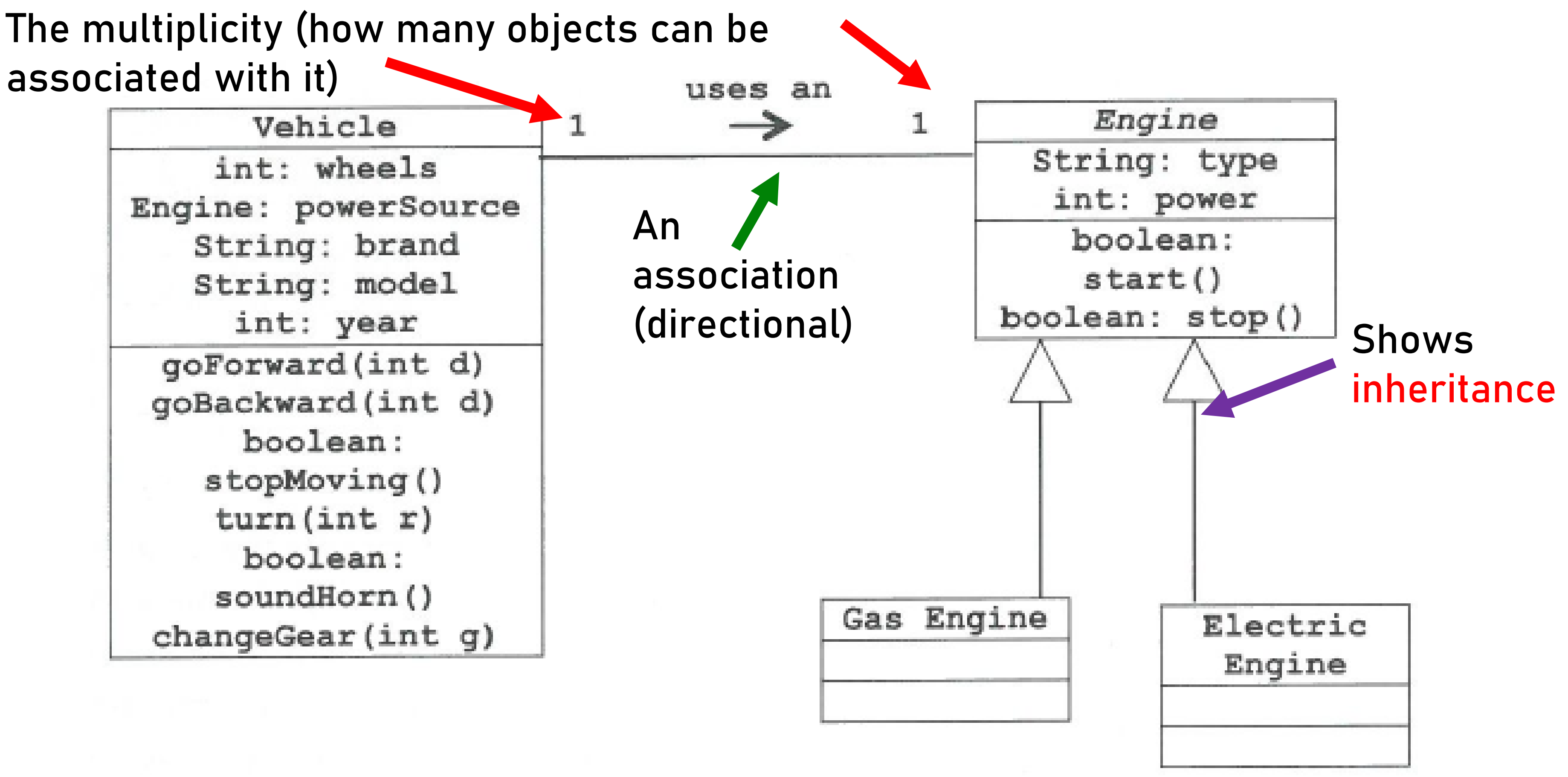
Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

Y12 & 13
IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

Associations, Multiplicities and Inheritance



Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

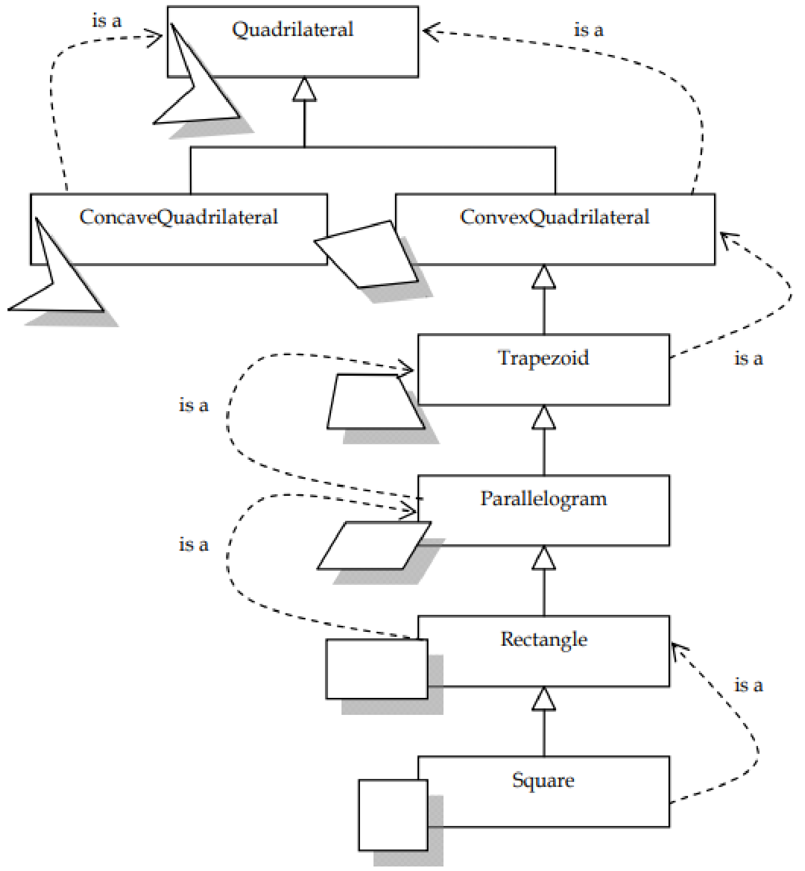
Y12 & 13 IB CS

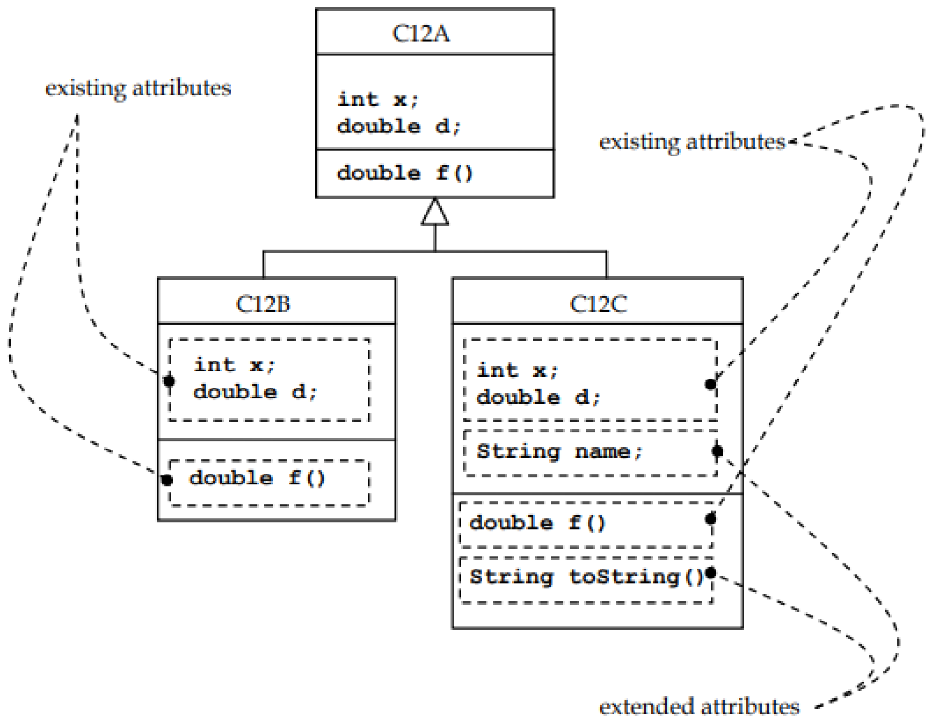
NORD ANGLIA EDUCATION

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

Associations, Multiplicities and Inheritance





Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

Y12 & 13 IB CS

Myran Teasdale

Activities: Theory

30:00

What is **inheritance**? Complete the term in your glossary

What are the **advantages** and **disadvantages** of this characteristic of OOP? Complete the table

Advantages	Disadvantages

OK, using the classes above and the information below, draw a UML diagram to show how the classes are connected including their data and actions. You should include the original superclass (Bicycle), and 3 subclasses (mountainBike, racingBike and stuntBike).

Data (attributes)

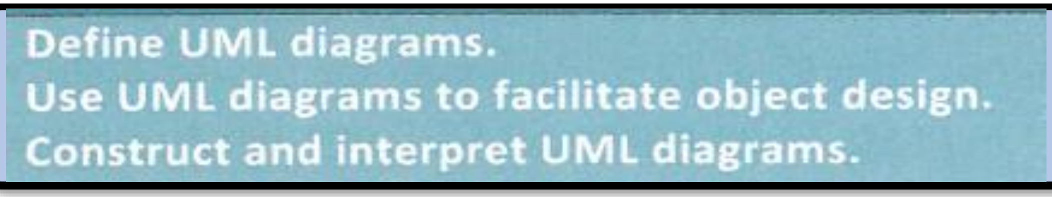
- ✓ The mountain bike has suspension
- ✓ The racing bike has wider tyres
- ✓ The stunt bike has adjustable seats

Actions (methods)

Using the above data, create actions for each of the data for the different bicycles.

Finished?

Move onto the practical tasks on the next page in the workbook



Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

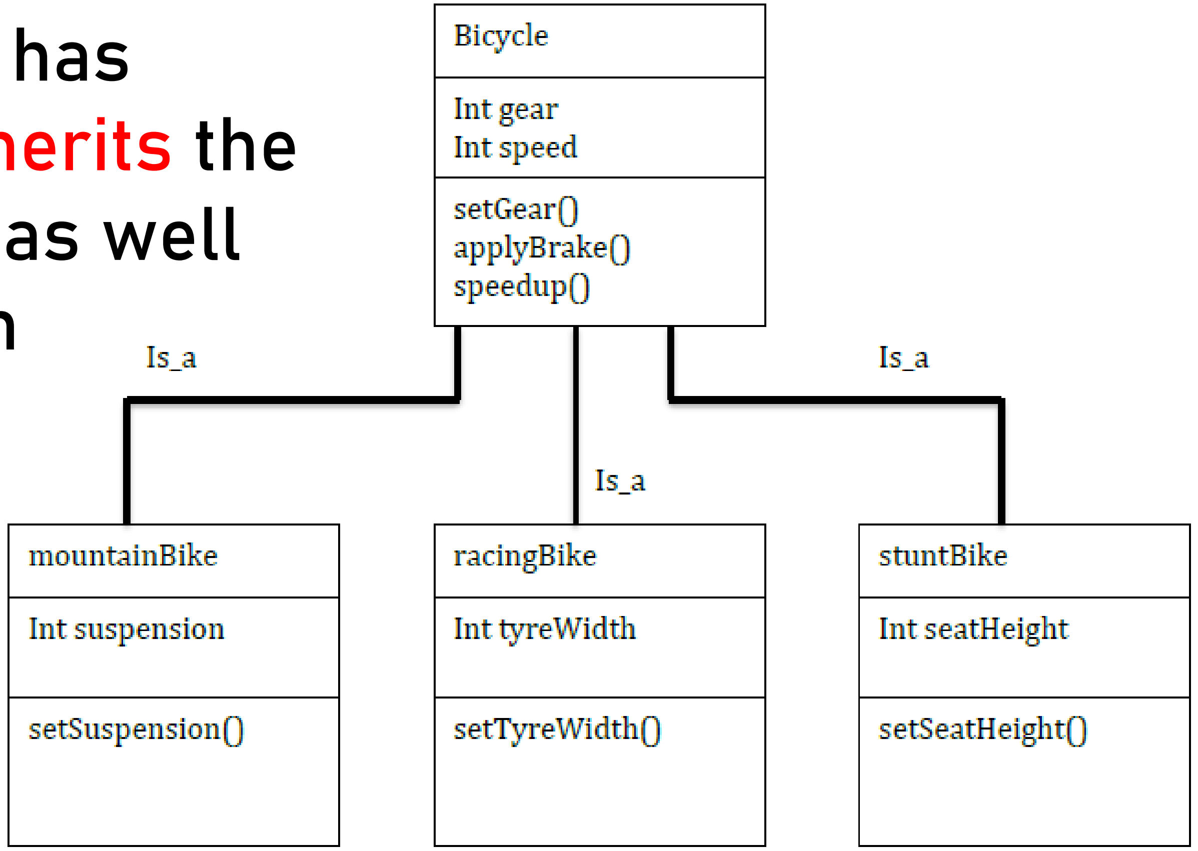
Y12 & 13
IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

UML Diagram Theory Activity - Solution

The bicycle class has classes which inherits the data and method as well as have their own



Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

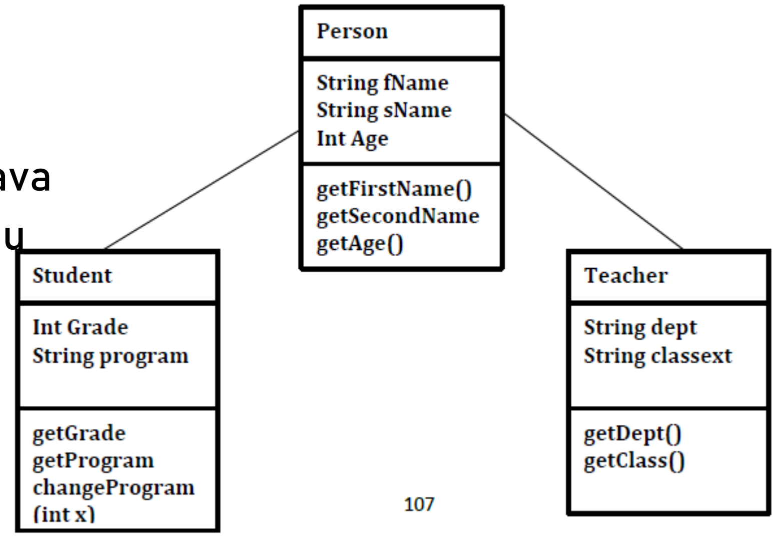
Y12 & 13 IB CS

Myran Teasdale

Activities: Practical

30:00

Try to code the following UML diagram in Java. You can see there is going to be 3 classes (Person, Student, Teacher). You should ideally create 4 separate java files for each class + 1 driver class. This is how OOP works! Alternatively, if you are struggling then just place them all into one java file. Here are the two methods and how they work theoretically.



__Method 1__

Accessing class methods outside of the Java file you must use the keywords public and extends as below:

__Bicycle.java__

- public class bicycle{}

__MountainBike.java__

- public class mountainBike extends bicycle{}

__Method 2__

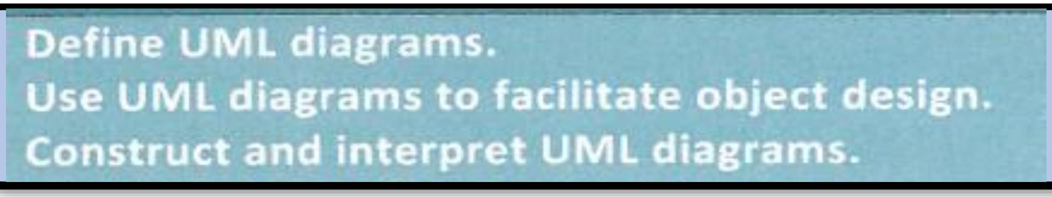
Accessing classes from within the same java file is easier but not typically used once you develop your own skills. Notice the lack of public and extends keywords:

__Bicycle.java__

- public class bicycle{}
- class mountainBike{}

Support? Ask your teacher for the password to the source code for this tasks' solution

__Source Code__ __https://repl.it/@BSBYear12CS/OOP-Person-UML-Activity__



Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

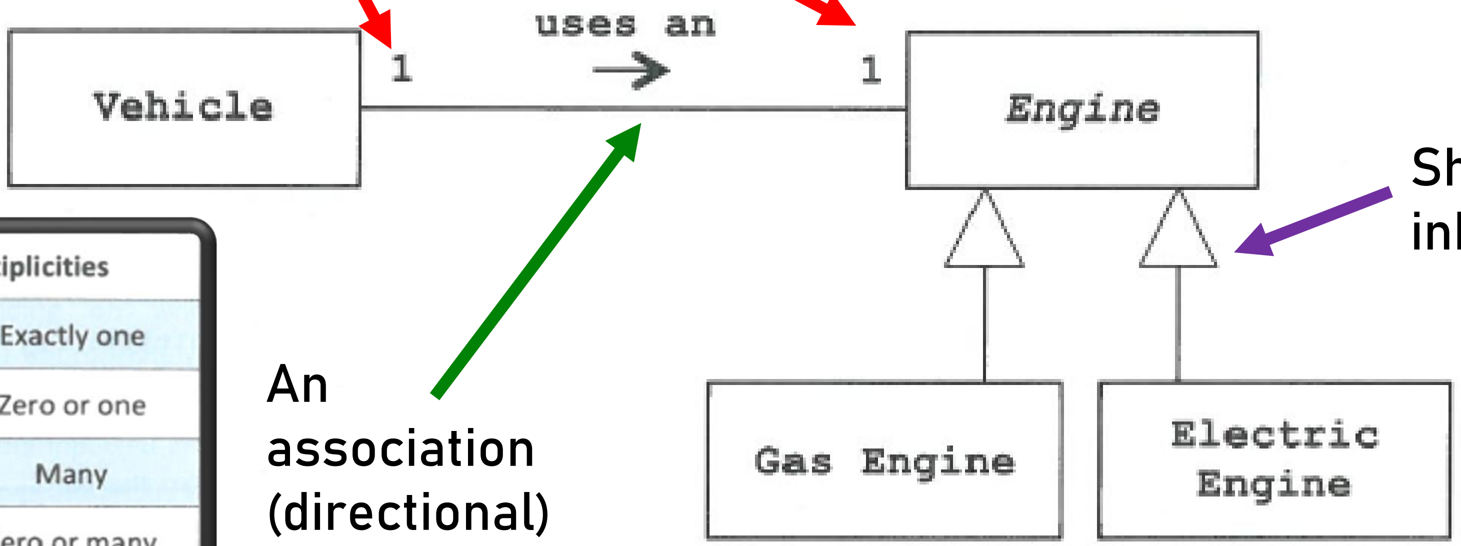
Y12 & 13
IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

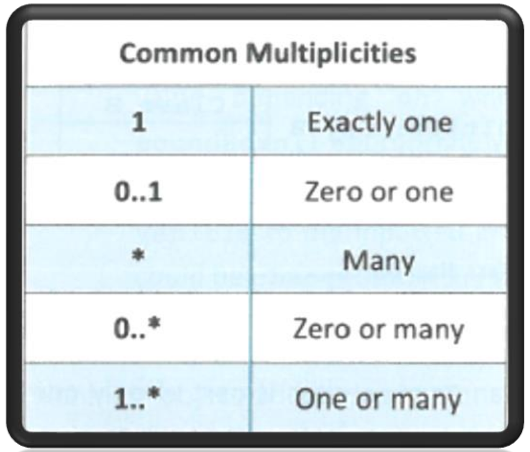
Simplifying a UML diagram

The multiplicity (how many objects can be associated with it)



Shows inheritance

An association (directional)



Common Multiplicities	
1	Exactly one
0..1	Zero or one
*	Many
0..*	Zero or many
1..*	One or many

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

Y12 & 13 IB CS

Myran Teasdale

Activities: Theory

30:00

Using the Java code in your workbooks, draw a UML Class Diagram showing the different instance variables, actions(methods), associations, multiplicities of each class.

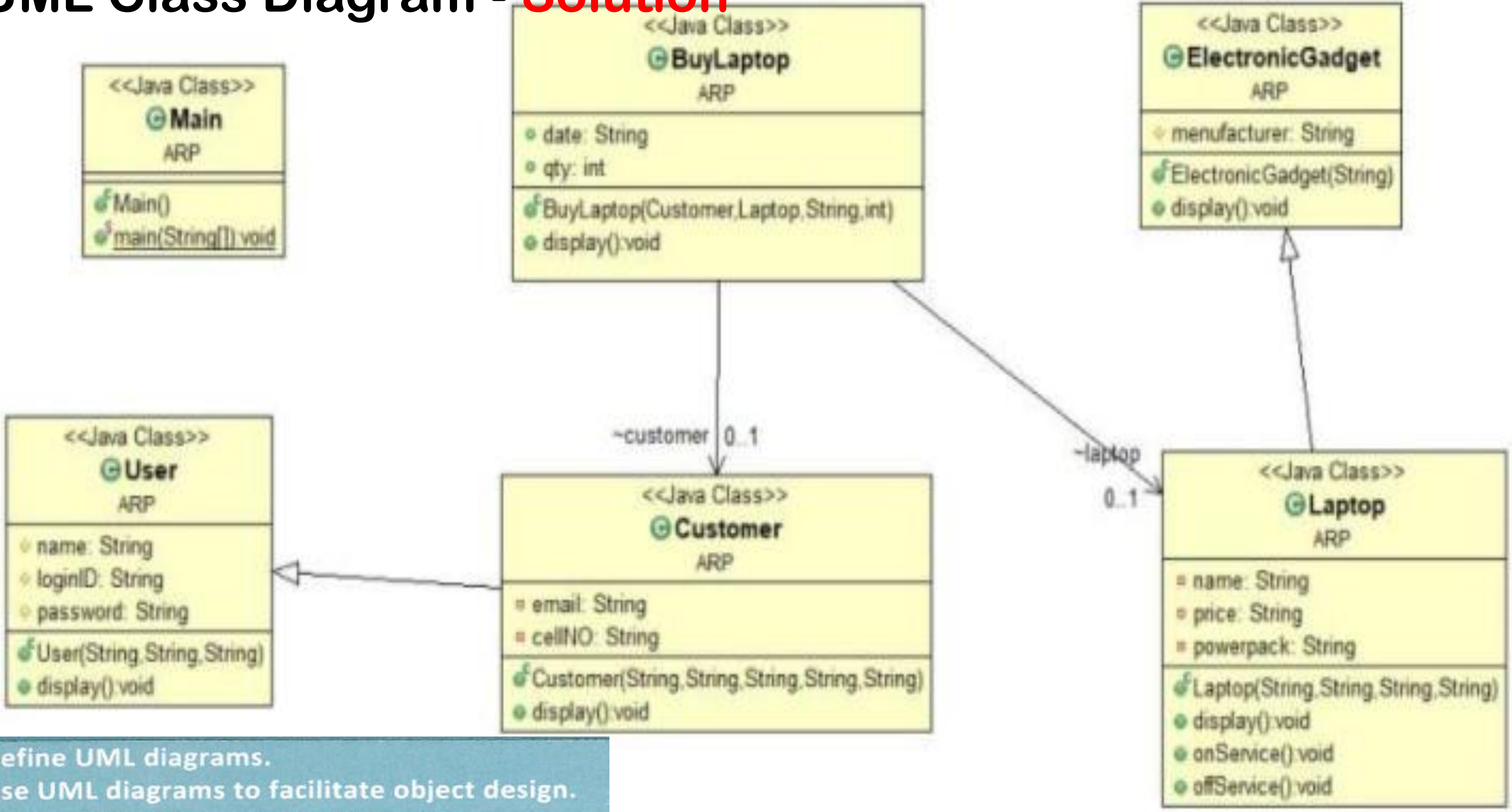
Support? https://www.javatpoint.com/uml-class-diagram

Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

Y12 & 13 IB CS

Myran Teasdale

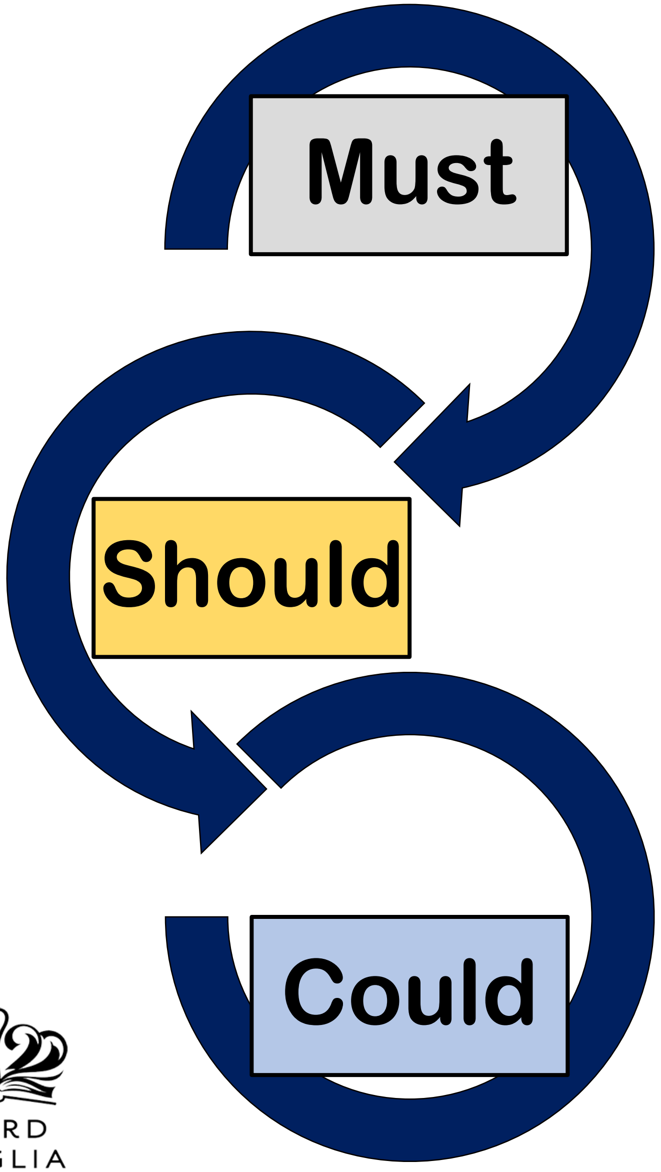
UML Class Diagram - Solution



Define UML diagrams.
Use UML diagrams to facilitate object design.
Construct and interpret UML diagrams.

Success Criteria

- D.1.5 Process of decomposition



SILVER

Describe the decomposition process of an object to several related objects

GOLD

Explain how the decomposition facilitates abstraction

PLATINUM

Use the objects’ decomposition process in real life situations

NORD ANGLIA EDUCATION

Decomposition

- What is decomposition?

- Watch the video, then feedback to the class. Complete the definition as you watch.

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

NORD ANGLIA EDUCATION

Y12 & 13 IB CS

Activity: Exam Board

Use you coursebooks for support (p.300)

TASK 1
TASK 2
TASK 3

10:00

- An school uses a computer system to record students, examination entries, record marks for each exam, and teachers involved in teaching the students.
- Draw a decomposed diagram showing distinct objects to illustrate how this computer system may be broken down into sub-systems. You should also show associations and multiplicities as per the previous topic D.1.2

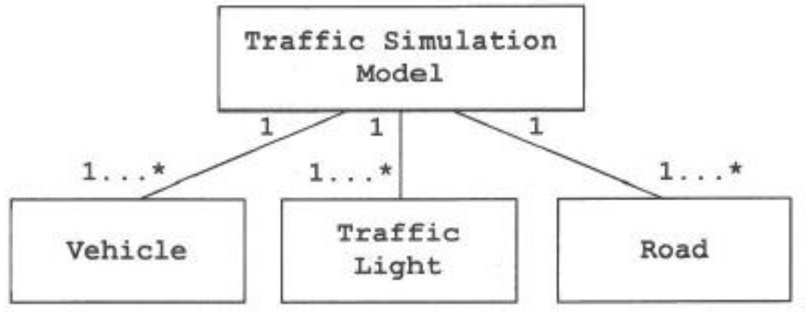


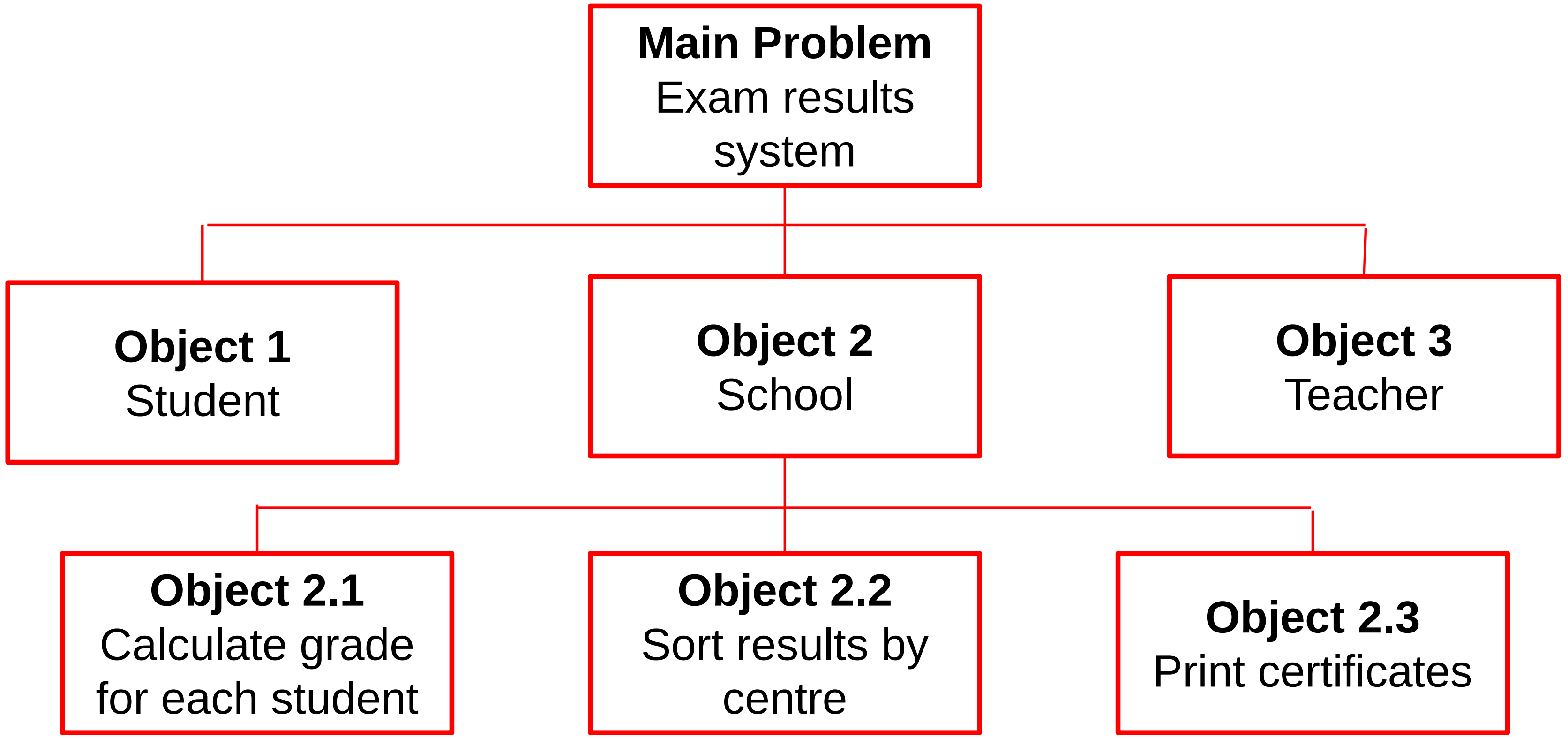
Figure D.13: The decomposition of a problem into related objects

Define the terms: class, template and instantiation.
Distinguish between an object and instantiation.
Discuss memory use and code definitions that relate to object and instantiation.

Y12 & 13 IB CS

L/A: To develop an understanding of objects as a programming concept

Decomposition Task – Possible Solution


Main Problem Exam results system
Object 1 Student
Object 2 School
Object 3 Teacher
Object 2.1 Calculate grade for each student
Object 2.2 Sort results by centre
Object 2.3 Print certificates

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
    - ❑ Getter/Setter
    - ❑ Accessor
    - ❑ Constructor

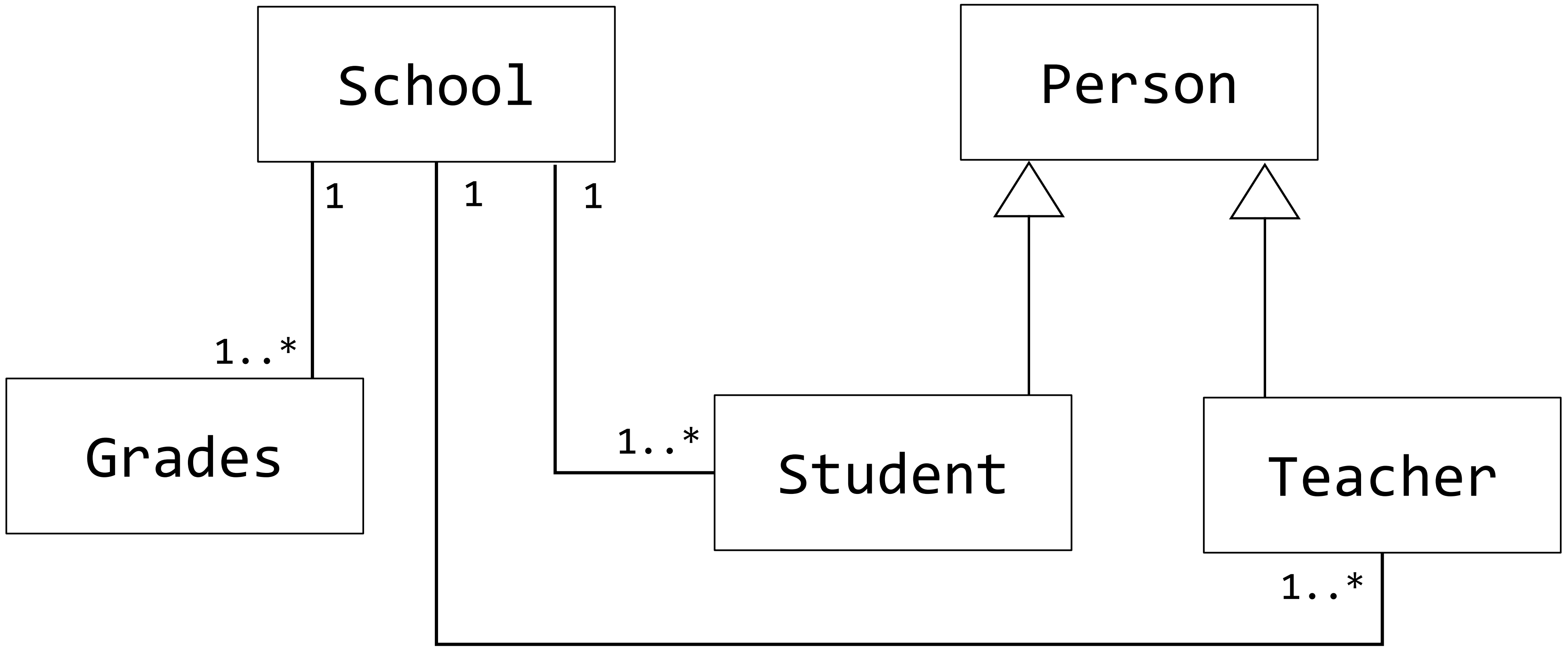
Describe the decomposition process of an object to several related objects.
Explain how the decomposition process facilitates abstraction.
Use the objects' decomposition process in real life situations.

Y12 & 13 IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

Decomposition Task – Possible Solution



Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Describe the decomposition process of an object to several related objects.
Explain how the decomposition process facilitates abstraction.
Use the objects' decomposition process in real life situations.

Y12 & 13
IB CS

Myran Teasdale

Success Criteria

- D.1.6 Relationships between objects

Must

Explain the dependency (“uses”), aggregation (“has a”), and inheritance (“is a”) relationship between objects in a given scenario

Should

Explain the dependency (“uses”), aggregation (“has a”), and inheritance (“is a”) relationship facilitate abstraction

Could

Activity: Research

20:00

Perform some **independent research** on the following terms and explain in as much detail as you can. You may use diagrams to aid your explanations.

Association

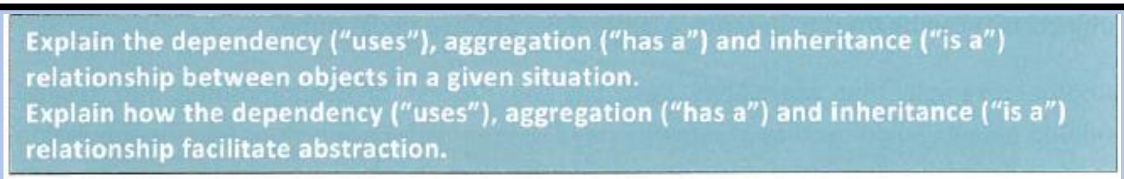
Dependency – "Uses" relationship

Aggregation – "has a" relationship

Finished?

How can all the terms you have learned facilitate abstraction?

Support: Here is a very basic definition is you are not sure how to begin to answer this question: https://bitly.im/Odjpr


Explain the dependency ("uses"), aggregation ("has a") and inheritance ("is a") relationship between objects in a given situation.
Explain how the dependency ("uses"), aggregation ("has a") and inheritance ("is a") relationship facilitate abstraction.

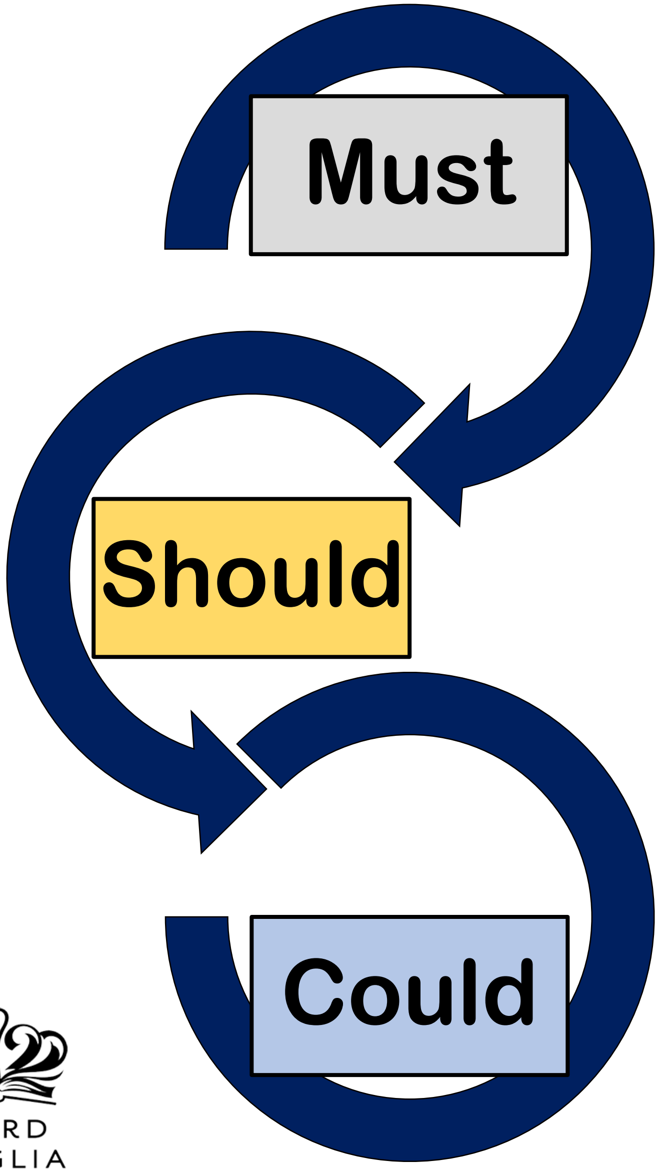
NORD ANGLIA EDUCATION

Y12 & 13 IB CS

Myran Teasdale

Success Criteria

- D.1.7 Need to reduce dependencies between objects



SILVER

Explain the negative effects that unnecessary dependencies between objects cause

GOLD

Discuss the increase of maintenance overhead because of increased dependencies

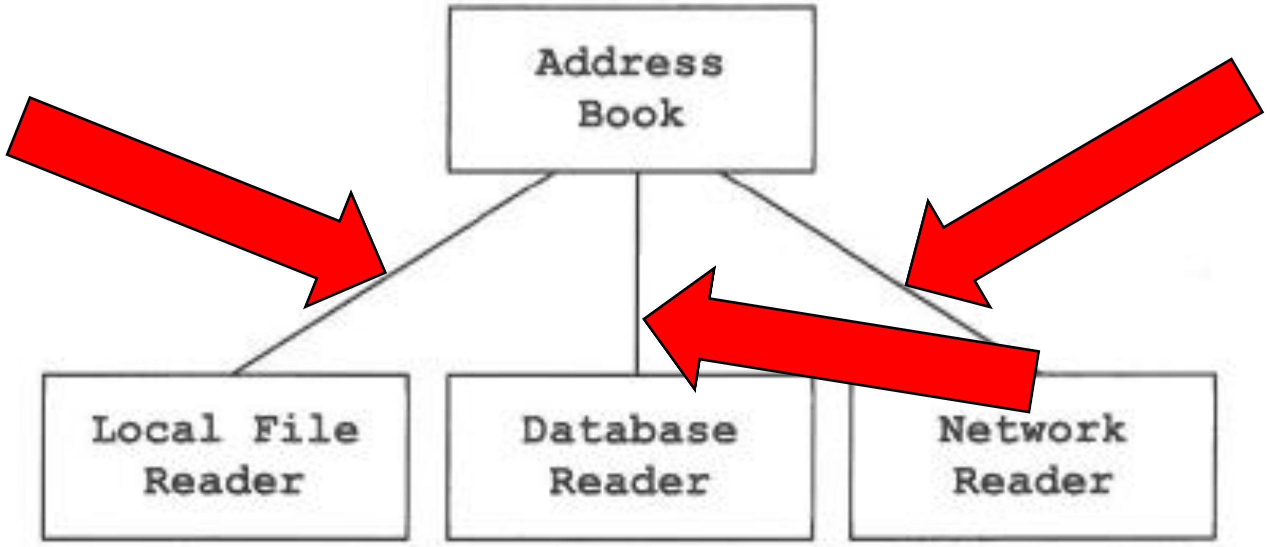
PLATINUM

L/A: To develop an understanding of objects as a programming concept

Dependencies between objects

5:00

What do you think a dependency is?



Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor

Explain the negative effects that unnecessary dependencies between objects cause.
Discuss the increase of maintenance overheads because of increased dependencies

Y12 & 13 IB CS

Myran Teasdale

L/A: To develop an understanding of objects as a programming concept

Dependencies between objects

5:00

What do you think a dependency is?

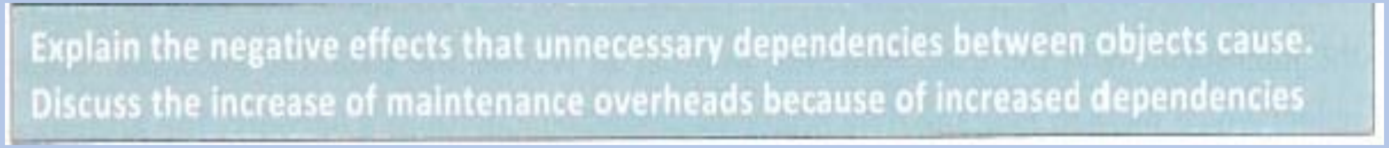
Why might they be useful?

What problems may occur if we have too many inside our programs?

How could we solve the issue of multiple unnecessary dependencies?

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
    - ❑ Getter/Setter
    - ❑ Accessor
    - ❑ Constructor

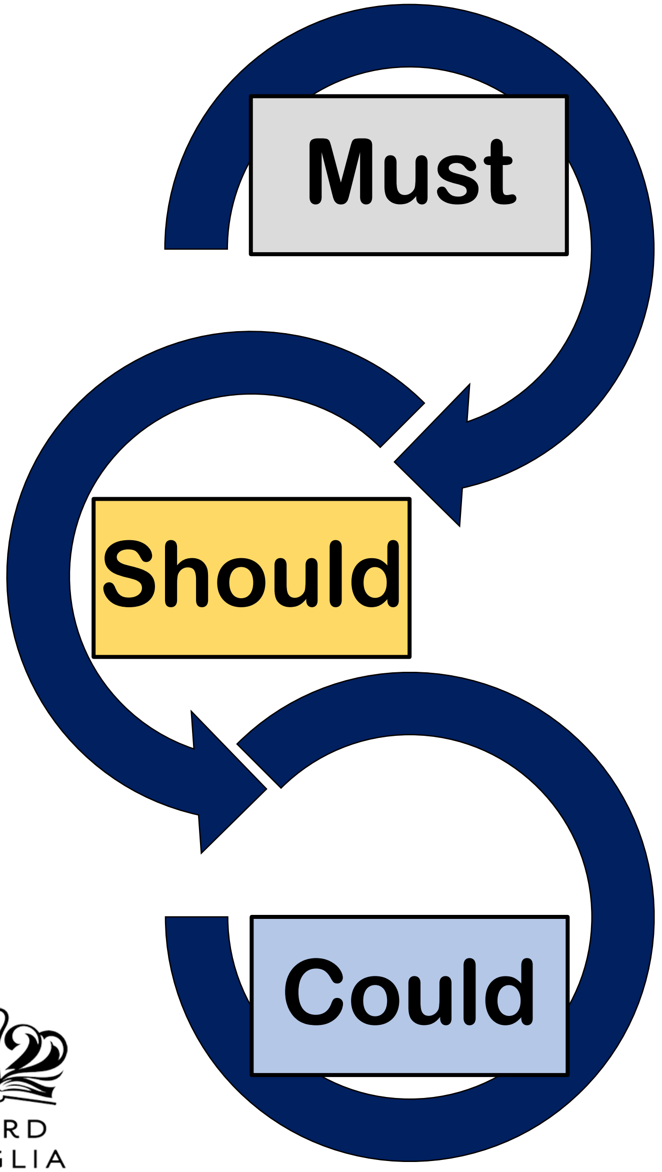

Explain the negative effects that unnecessary dependencies between objects cause.
Discuss the increase of maintenance overheads because of increased dependencies

Y12 & 13 IB CS

Myran Teasdale

Success Criteria

- D.1.8 Constructing Related Objects



SILVER

Develop objects for a given scenario

GOLD

Develop various object definitions

PLATINUM

Explain the relationships of objects to each other and to any additional classes defined by a given scenario

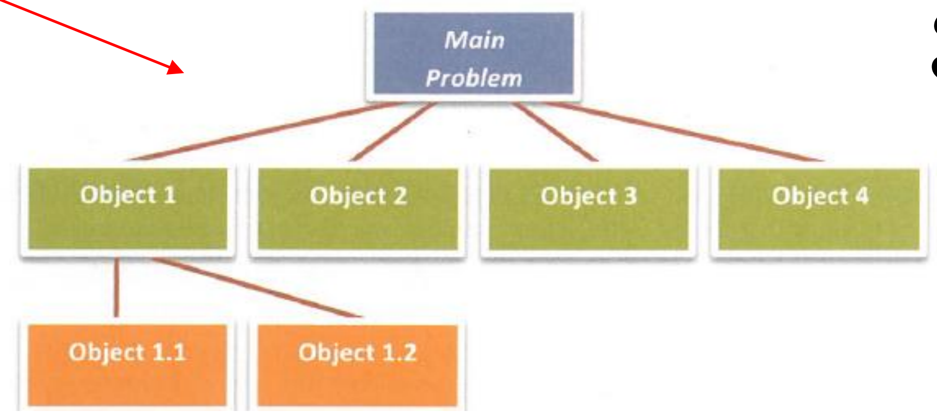
NORD ANGLIA EDUCATION

Activity: Constructing Related Objects

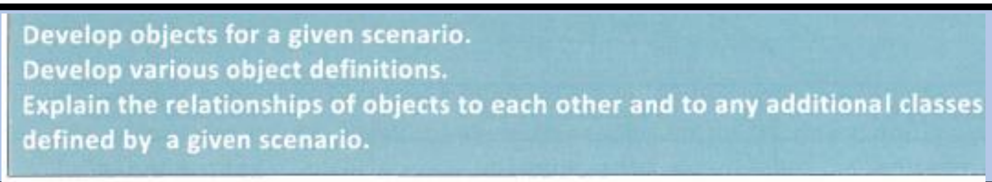
- You are going to develop a solution for a given scenario
- You will first study the scenario carefully in your workbooks
- Part 1: You will then begin to develop a simple top-down design breakdown of objects and dependencies between the objects like we saw in fig D.11 on page 17

30:00

If you need support you should look at your coursebooks p.308 for an example



- Part 2: You will then present the breakdown of your Part 1 solution as a Unified Modelling Diagram (UML) to consolidate your learning of UML.

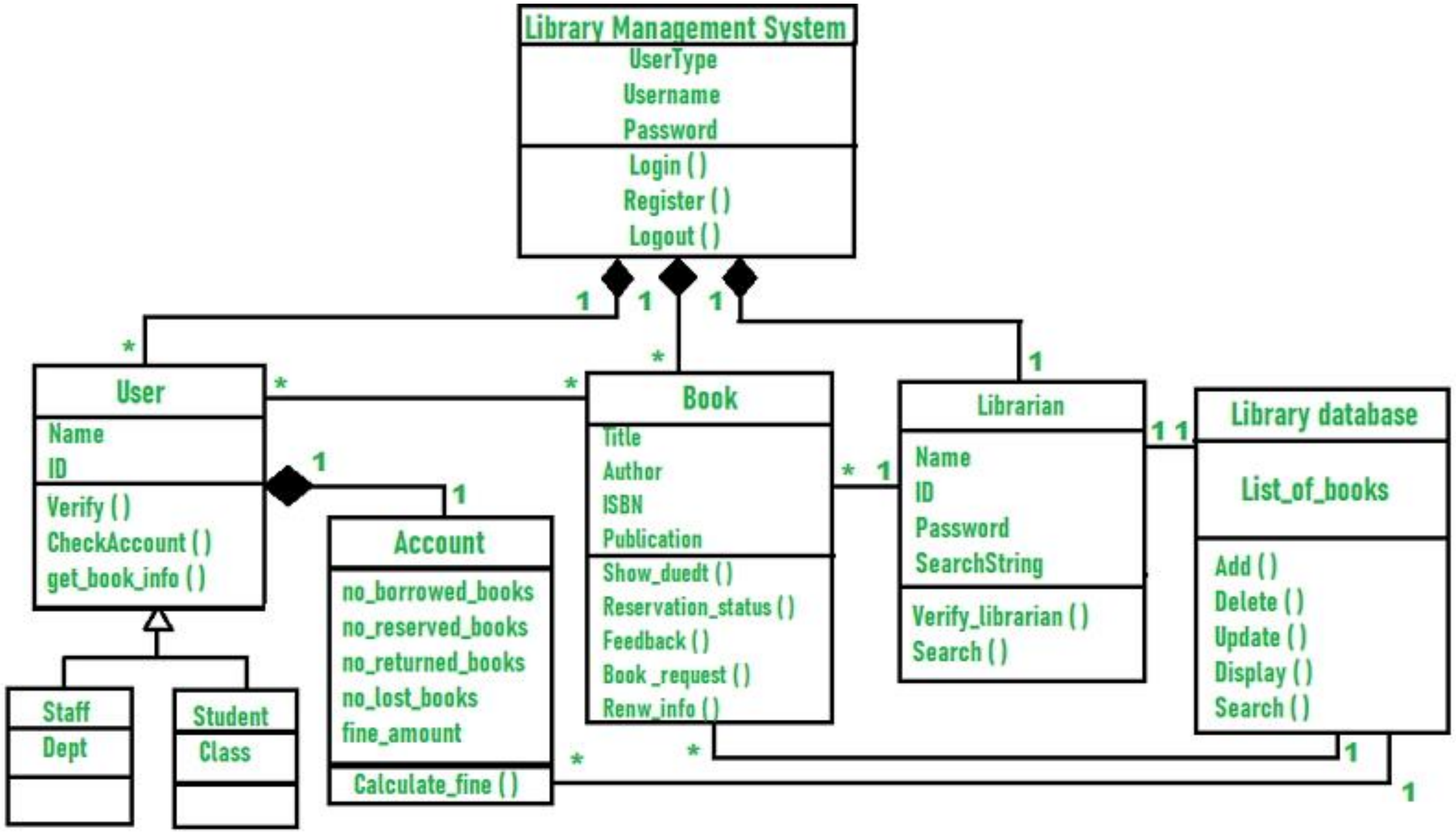


Develop objects for a given scenario.
Develop various object definitions.
Explain the relationships of objects to each other and to any additional classes defined by a given scenario.

Y12 & 13 IB CS

Myran Teasdale

Activity: Solution to the Library System



TASK 1
TASK 2
TASK 3

30:00

Develop objects for a given scenario.
Develop various object definitions.
Explain the relationships of objects to each other and to any additional classes defined by a given scenario.

Y12 & 13
IB CS

Myran Teasdale

Success Criteria

- D.1.9 Data Types

Must

Should

Could

SILVER

Explain the need of integer, real, string and Boolean data types

GOLD

Explain how real world items are representation, stored and manipulated by different data types

PLATINUM

NORD ANGLIA EDUCATION

L/A: To develop an understanding of objects as a programming concept

Activity

5:00

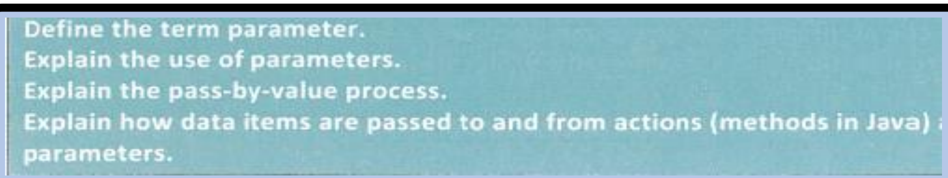
Complete the table of data types in your workbook

Type	Why it is used	Example

Extension. Decribe what primitive data types are and list 4 primitive types that the Java language uses.

Literacy Focus

- ❑ Object
- ❑ Class
- ❑ Instantiation
- ❑ UML diagram
- ❑ Decomposition
- ❑ Inheritance
- ❑ Encapsulation
- ❑ Polymorphism
- ❑ Instance variable
- ❑ Methods
  - ❑ Getter/Setter
  - ❑ Accessor
  - ❑ Constructor



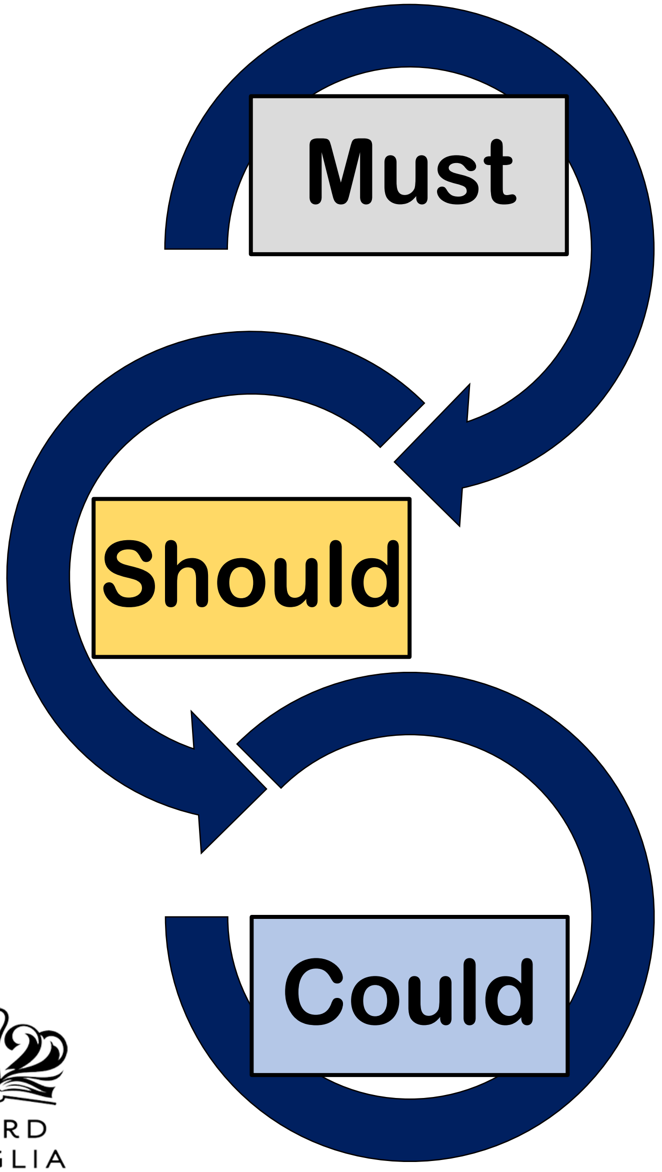
Define the term parameter.
Explain the use of parameters.
Explain the pass-by-value process.
Explain how data items are passed to and from actions (methods in Java) a
parameters.

Y12 & 13
IB CS

Myran Teasdale

Success Criteria

- D.1.10 Data items passed as parameters



SILVER

Define the term parameter

GOLD

Explain the use of parameters as well as the pass-by-value process

PLATINUM

Explain how data items are passed to and from actions (methods in Java) as parameters

L/A: To develop an understanding of objects as a programming concept

Parameters

5:00

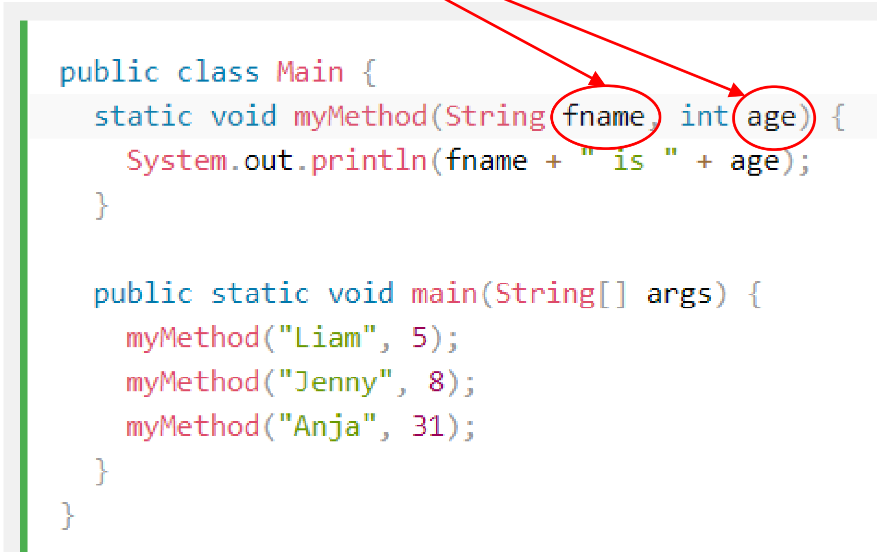
Literacy Focus

❑ Object
❑ Class
❑ Instantiation
❑ UML diagram
❑ Decomposition
❑ Inheritance
❑ Encapsulation
❑ Polymorphism
❑ Instance variable
❑ Methods
  ❑ Getter/Setter
  ❑ Accessor
  ❑ Constructor

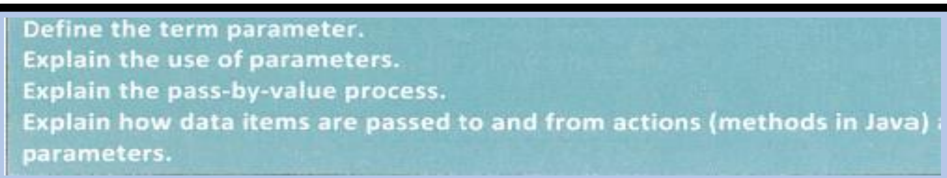
What are parameters?

Information can be passed to methods as parameter. Parameters act as variables inside the method.

Parameters are specified after the method name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma.



```
public class Main {
  static void myMethod(String fname, int age) {
    System.out.println(fname + " is " + age);
  }

  public static void main(String[] args) {
    myMethod("Liam", 5);
    myMethod("Jenny", 8);
    myMethod("Anja", 31);
  }
}
```



Define the term parameter.
Explain the use of parameters.
Explain the pass-by-value process.
Explain how data items are passed to and from actions (methods in Java)
parameters.

Y12 & 13
IB CS

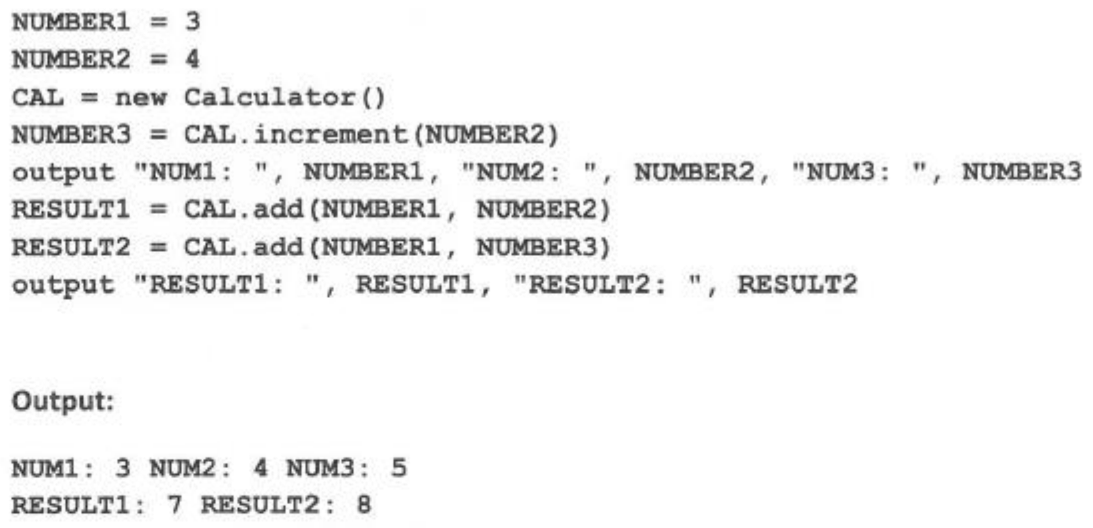
Myran Teasdale

Practical Activity: A simple Calculator!

- Implement the pseudocode algorithm found in the coursebook (p.311)
- You should be making use of a separate "Calculator" class as well as your main class to "drive" the Calculator
- The begin with create a UML diagram of what the program will look like. Be sure to draw it correctly with the class name, instance variables/attributes and the methods/actions it will perform
- Once you have done that begin to code the solution in your favourite IDE

If you need support, you should look at the previous tasks' source code on how to setup a new class along with the methods

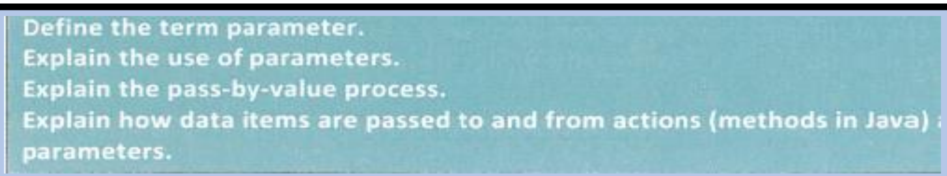
30:00



```
NUMBER1 = 3
NUMBER2 = 4
CAL = new Calculator()
NUMBER3 = CAL.increment(NUMBER2)
output "NUM1: ", NUMBER1, "NUM2: ", NUMBER2, "NUM3: ", NUMBER3
RESULT1 = CAL.add(NUMBER1, NUMBER2)
RESULT2 = CAL.add(NUMBER1, NUMBER3)
output "RESULT1: ", RESULT1, "RESULT2: ", RESULT2
```

Output:

```
NUM1: 3 NUM2: 4 NUM3: 5
RESULT1: 7 RESULT2: 8
```



Define the term parameter.
Explain the use of parameters.
Explain the pass-by-value process.
Explain how data items are passed to and from actions (methods in Java)
parameters.