



**Edith Cowan University**  
2024 ATAR Revision Seminar

**ATAR Computer Science**

Curriculum Dot points  
Examination and study tips  
Revision notes  
Examination questions

Prepared and presented by  
**Chris Anderson**

# Contents

Programming .....	5
Framework for Development .....	5
Key Points .....	5
Key Skills and Concepts .....	6
Data Types .....	6
Programming Concepts .....	7
Pseudocode .....	8
Common Commands .....	9
Control Structures .....	10
Modularisation .....	11
Exercises .....	13
Structured Algorithms .....	15
Big O notation .....	15
Linear Search .....	16
Binary Search .....	16
Bubble Sort .....	17
Insertion Sort .....	17
Selection Sort .....	18
Time Complexity for Common Algorithms .....	19
Error Checking .....	19
Types of Errors .....	19
Desk Checking .....	19
Completing a Trace Table .....	19
Example .....	19
Exercises .....	20
Object-Oriented Programming .....	22
Key Concepts .....	22
Sample Pseudocode .....	23
Defining a Class for each Class at School .....	24
Exercises .....	25
Networks and Communication .....	29
Terminology .....	29
Models of Networking .....	29
OSI Model .....	30
TCP/IP Model .....	31

Internet Protocol .....	34
Exercises .....	38
Network Components .....	42
Device Roles.....	42
Other Common Devices (not in current syllabus) .....	43
Transmission Media.....	43
Exercises .....	44
Network Diagrams.....	45
CISCO Symbols.....	45
Example .....	46
Exercises .....	47
Cyber Security.....	50
Network Threats.....	50
External Network Threats.....	50
Internal Network Threats .....	51
Security methods.....	52
Exercises .....	53
Cryptography.....	56
Key Points .....	56
Exercises .....	57
Data Management.....	58
Core Concepts .....	58
ACID .....	59
Data Integrity.....	59
Exercise.....	60
Normalisation .....	61
Data Anomalies.....	61
Normalisation to 3 <sup>rd</sup> Normal Form .....	62
Normalisation Example .....	63
Exercises .....	66
Entity Relationship Diagrams .....	69
Example .....	69
Creating an ERD .....	69
Many-to-many relationships .....	69
Sample ERD.....	71
Exercises .....	72
Using SQL.....	77
Development Issues .....	78

Exercise .....	79
Exam Technique .....	81
Sitting the Exam.....	81
Reading time.....	81
Start paper .....	81
Preparation.....	81

# Programming

## Framework for Development

### Key Points

- The framework for development provides a way to structure, plan and control the process of developing a system.
- For the purposes of the ATAR syllabus, the stages are:
  - investigate
    - problem description
    - define requirements
    - development schedule (including Gantt charts)
  - design
    - design data structures
    - design and test algorithm
  - develop
    - develop and debug code
    - unit testing and use of live data
  - evaluate
    - user acceptance testing
    - developer retrospective
- This framework provides the main steps that all projects need to cover, regardless of which specific methodology is being used
- There are two main categories of development processes:
  - Linear
  - iterative
- **Linear:** each stage is completed before the next stage starts.
  - Highly structured
  - Good for large projects
  - Less flexible
  - Easier to follow for less experienced developers
- **Iterative:** some stages (usually design and develop) are repeated several times.
  - More flexible
  - Greater involvement from the end user/client
  - Allows for changes to be made to original design
  - Good for smaller projects
  - Requires highly skilled staff
  - Requires regular input from clients

---

## Key Skills and Concepts

---

- program control structures, including:
  - sequence
  - selection
  - iteration
    - post-test
    - pre-test
    - fixed length
- characteristics of data types used in solutions, including:
  - integer
  - float
  - string
  - Boolean
- modular coding using functions, parameters and arguments
  - scope of variables (Global, Local)
- characteristics of the following data structures:
  - arrays
    - one-dimensional arrays
    - two-dimensional arrays
  - dictionaries
- different types of operators
  - arithmetic operators (+, -, \*, /, %)
  - relational operators (==, !=, >, =, <=)
  - logical operators (AND, OR, NOT)
- complex logical expressions
  - Boolean operators (AND, OR, NOT)
  - logical order of precedence

### Data Types

- Data types:
  - **Integer**: a whole number. The number of bytes used to represent an integer determines the size of the integers that can be stored.
  - **Float** (floating point number): number that includes a decimal place
  - **Boolean**: used to represent data that is either *True* or *False*
  - **Character**: a single unit of data used to store a single alphanumeric entry, such as a letter, digit or punctuation. In ASCII, this is a single byte.
  - **String**: a collection of characters
- Data structures:
  - **Array**: stores multiple values of the same data type
  - **Two-dimensional array**: an array of arrays. Useful to store things such as tables or grids. All values should be of the same data type
  - **Dictionary**: stores data as key-value pairs. Each key must be unique, but values can be repeated.

## Programming Concepts

- **Constants:** named memory location whose literal value does not change throughout the execution of a program
- **Variables:** named memory location whose contents can change and be used during the execution of a program
- **Scope of Identifiers:** the scope of an identifier indicates in which sections of the code an identifier is accessible
  - **Local:** are only accessible within the section of code that they have been declared
  - **Global:** are accessible throughout the entire program
- **Modularisation:** the process of breaking a program down into several smaller sections that can be used to perform a specific purpose. Values can be passed into functions for use within that function.
  - **Parameters:** named in the function definition that acts as a placeholder for the data that can be used by the function. Similar to a local variable but uses value that is passed in when the function is called.
  - **Arguments:** the actual value that is passed to the function when it is called.
  - **Types of parameters:** not strictly in the syllabus, but useful knowledge:
    - **Value parameters:** a copy of the actual data is passed to the module that is being called. Any changes to the parameter inside the module do not affect the original value
    - **Reference parameters:** a pointer to the variable's memory location is passed to the module being called. Any changes to the parameter cause the original value to be changed

---

## Pseudocode

---

- use pseudocode to represent a programming solution
  - apply, using pseudocode and a programming language, the following programming concepts:
    - constants
    - variables (local, global, parameters)
    - naming conventions for variables
    - stubs
    - statements
    - modularisation
    - functions
    - parameter passing (value and reference)
    - one-dimensional arrays
  - apply, using pseudocode and a programming language, the following control structures:
    - sequence
    - selection
      - one-way (if then)
      - two-way (if then else)
      - multi-way (case, nested if)
    - iteration
      - test first (while)
      - test last (repeat until)
      - fixed (for)
- 
- Pseudocode uses a form of structured English to describe an algorithm
  - Some tips for writing good pseudocode:
    - Make sure your algorithm is language independent. Good pseudocode should be able to be translated into any suitable programming language
    - Write only one statement per line
    - Use capital letters for keywords
    - Make sure you indent your code to show the structure of the code
    - Clearly indicate the end of all structural elements (e.g. functions) and control structures (e.g. IF...END IF)
    - Avoid making your code unnecessarily complex
    - When naming variables:
      - Begin the variable name with lower case letters
      - Should always begin with a letter
      - Names should only contain letters or numbers
      - Use a capital letter to begin new word (Camel Case)
      - Do not add spaces to a name
    - Use the symbol = to indicate an assignment statement
    - Use the symbol == to indicate a comparison statement
    - Initialise all variables at the start of each function
    - Constants should be clearly indicated (use the CONST keyword)
    - Global variables should be declared outside the main function to indicate that they are global. Variables declared inside a function should be considered local to that function.
  - **NOTE:** There is no set standard for exactly how to write pseudocode, but you need to make sure it is clear and easy for the marker to understand. The sections below show the recommended way of writing pseudocode.



## Common Commands

User input	INPUT(num)
User output	OUTPUT("Hello world!") PRINT("Hello world!")
Assignment	=
Equals (comparison)	==
Not equal to	!=
Greater than	>
Greater than or equal to	>=
Less than	<
Less than or equal to	<=
Integer division	// or DIV    e.g. 7 // 2 = 3
Modulus (remainder)	% or MOD    e.g. 7 % 2 = 1
OR	x < 1 OR x > 10
AND	x > 1 AND x < 10
Arrays	scores = [] scores[0] = 15 scores[1] = 16  scores.append(12) # add element to end of array scores.length    # gives the number of elements in an array
Dictionaries	costOfGear = {}    # empty dictionary costOfGear = { "mask": 2, "wetsuit": 5, "BCD": 7, "tank": 5 } costOfGear["fins"] = 2    # add new key:value pair costOfGear["wetsuit"] = 6    # update value of wetsuit  cost = costOfGear["mask"]    # value of cost will be 2  costOfGear.keys    # array of all keys in the dictionary costOfGear.values    # array of all values in the dictionary costOfGear.items    # array of all key/value pairs in the dictionary

## Control Structures

- Control structures are used to control the execution flow of a program, allowing a program to branch or repeat lines of code.

### Selection

One-way selection	If a condition is TRUE, run some code, otherwise no code is run, and the statement terminates	<pre> speed = 50 IF speed &gt; 60 THEN     PRINT("You are speeding") END IF </pre>
Two-way selection	If a condition is TRUE, run some code, otherwise if the condition is FALSE run some other code.	<pre> speed = 50 IF speed &gt; 60 THEN     PRINT("You are speeding") ELSE     PRINT('You are not speeding') END IF </pre>
Multi-way selection	Test a condition against a variety of different outcomes and perform an action based on which outcome is TRUE	<p><b>Method 1 – IF...ELSE IF...ELSE</b></p> <pre> speed = 50 IF speed &lt; 20 THEN     PRINT("You are going too slow") ELSE IF speed &gt; 60 THEN     PRINT("You are speeding") ELSE     PRINT('You are not speeding') END IF </pre> <p><b>Method 2 – CASE statement</b></p> <pre> colour = 'red' CASE colour OF     'red': PRINT("Stop")     'yellow': PRINT ("Slow down")     'green': PRINT("Go")     OTHER: PRINT("Incorrect colour") END CASE </pre>

### Iteration (or Repetition)

Test-first loop (WHILE)	A series of instructions are executed while the test condition is TRUE. The condition is tested before the instructions are executed.	<pre> num = 0 WHILE num &lt; 10     PRINT("The number is " + num)     num = num + 1 END WHILE </pre>
Test-last loop (REPEAT UNTIL)	A series of instructions are executed until the test condition is true (that is, it continues executing while the condition is FALSE). The condition is tested at the end of each loop.	<pre> num = 0 REPEAT     PRINT("The number is " + num)     num = num + 1 UNTIL num = 10 </pre>
Fixed loop (FOR)	Instructions are repeated a fixed number of times. These loops are used when the number of required iterations is known.	<pre> FOR num = 1 TO 10     PRINT("The number is " + num) END FOR </pre>

## Modularisation

- Modularisation is a methodology that involves breaking a problem down into smaller, less complex parts.
- Benefits of modularisation include:
  - Allows code to be reused, thereby reducing code repetition
  - Allows more people to work on a project – each person can work on separate modules
  - Breaking a large, complex problem down into smaller problems makes it easier to solve
  - Makes it easier to read algorithms and programs
  - Makes it quicker and easier to find errors

### Example

Without Modularisation	With Modularisation
<pre> FUNCTION Main   INPUT(length)   INPUT(height)   area1 = length * height    INPUT(length)   INPUT(height)   area2 = length * height    INPUT(length)   INPUT(height)   area3 = length * height    total = area1 + area2 + area3   PRINT("The total area is", total) END Main </pre>	<pre> FUNCTION Main   INPUT(length)   INPUT(height)   area1 = CalculateArea(length, height)    INPUT(length)   INPUT(height)   area2 = CalculateArea(length, height)    INPUT(length)   INPUT(height)   area3 = CalculateArea(length, height)    total = area1 + area2 + area3   PRINT("The total area is", total) END Main  FUNCTION CalculateArea(length, height)   area = length * height   RETURN area END CalculateArea </pre>

- The code on the left repeats the same lines of code three times where it calculates the area based on the length and height.
- The code on the right reduces this repetition by moving those lines of code to a separate function.
- Functions should perform a single task.
- When required to return a value, use the RETURN keyword. Avoid the use of reference parameters to return a value wherever possible.

### Parameter Passing

- We use **parameters** to access values that are passed into a function. Parameters are placeholders that are defined in the function definition.
- **Arguments** are the actual values that are passed in when the function is called. These values are placed in the parameters to be used inside the function.

- The following content is not in the current syllabus, but is useful information that might aid in understanding the behaviour of your code:
  - There are two types of parameters:
    - **Value parameters:** a copy of the actual data is passed to the module that is being called. Any changes to the parameter inside the module do not affect the original value
    - **Reference parameters:** a pointer to the variable's memory location is passed to the module being called. Any changes to the parameter cause the original value to be changed
  - In most programming languages, simple data types (including strings) will be passed by value and complex data types (such as arrays and records) will be passed by reference.



## Question 2

### Use the following information to answer parts (a) and (b)

K-Clothing is a small retail store that currently has 4 employees. To ensure that only authorised people can access their sales system, they have designed a login screen.

The system uses a global dictionary, *employees*, to store the details of each employee:

`employees = {}`

- (a) Currently the *employees* dictionary is empty and the manager needs to enter the details for each current employee.

Using pseudocode, write a function called *EnterDetails* that will allow the user to enter the username and password for each employee.

---

---

---

---

---

---

---

---

---

---

- (b) As the store expands its business, they want to add new employees to the list. Write a FUNCTION *AddEmployee* that will take in an employee name and password as parameters and add them to the array of employees. Your algorithm should check that the employee is not already in the system before adding their details.

If the new employee is already in the system, the FUNCTION should print 'Employee already in system'. Otherwise, the FUNCTION should print 'Employee successfully added'.

---

---

---

---

---

---

---

---

---

---

## Structured Algorithms

### Big O notation

- Big O notation is a way to look at the efficiency of algorithms so that we can compare their performance.
- Works out how long an algorithm as the input size increases and looks at the performance in the *worst case scenario* (as the input size heads towards infinity)
- Five most common Big O run times are:
  - $O(\log n)$ , also known as log time. Example: Binary search.
  - $O(n)$ , also known as linear time. Example: Simple search.
  - $O(n * \log n)$ . Example: A fast sorting algorithm, like quicksort.
  - $O(n^2)$ . Example: A slow sorting algorithm, like selection sort.
  - $O(n!)$ . Example: A really slow algorithm, like the traveling salesperson.

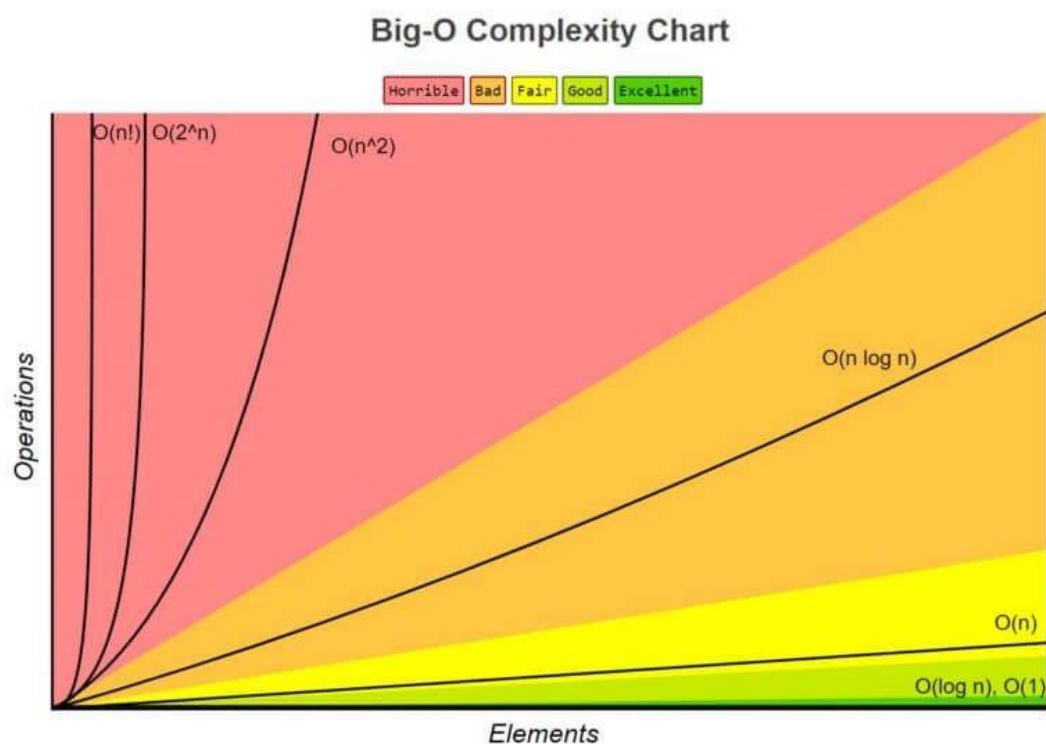


Image from <https://www.bigocheatsheet.com/>

## Linear Search

The linear search will go through an array and check each element for the target until it is found. If it does not find the target, it will move through the array until the end.

The algorithm below will return the index of the target element if it is found. If the target element is not found it will return -1.

```
FUNCTION LinearSearch(searchArray, target)
  index = 0
  position = -1
  WHILE index < searchArray.length AND position == -1
    IF searchArray[index] = target THEN
      position = index
    END IF
    index = index + 1
  END WHILE
  RETURN position
END LinearSearch
```

## Binary Search

The binary search works by comparing the middle element of an array to the target element. If a match is not found, then the element array is split into two. If the element is less than the middle element, then the sub-array continues the search until the numbers can be split.

**NOTE:** The binary search requires the array to be sorted to work properly.

```
FUNCTION BinarySearch(searchArray, target)
  position = -1
  lowerBound = 0
  upperBound = searchArray.length - 1
  WHILE lowerbound <= upperBound AND position == -1
    midpoint = (lowerBound + upperBound) / 2
    IF searchArray[midpoint] < target THEN
      lowerBound = midpoint + 1
    ELSE IF searchArray[midpoint] > target THEN
      upperBound = midpoint - 1
    ELSE
      position = midpoint
    END IF
  END WHILE
  RETURN position
END BinarySearch
```



## Bubble Sort

Bubble sort is a simple sorting algorithm that repeatedly compares adjacent elements in a list and swaps them if they're in the wrong order. The goal is to "bubble up" the largest elements by moving them to their correct positions. Each pass ensures that the largest unsorted element finds its place. However, bubble sort is not efficient for large data sets due to its slow performance.

```
FUNCTION BubbleSort(arrayToSort)
  last = arrayToSort.length - 1
  swapped = TRUE
  WHILE swapped
    swapped = FALSE
    i = 0
    WHILE i < last
      IF arrayToSort [i] > arrayToSort [i + 1] THEN
        temp = arrayToSort [i]
        arrayToSort [i] = arrayToSort [i + 1]
        arrayToSort [i + 1] = temp
        swapped = TRUE
      END IF
      i = i + 1
    END WHILE
    last = last - 1
  END WHILE
  RETURN arrayToSort
END BubbleSort
```

## Insertion Sort

Insertion sort is a straightforward sorting algorithm that operates much like how you arrange playing cards in your hand. Here's how it works: Imagine you have an array of elements (numbers, for instance). Insertion sort virtually splits the array into two parts: a sorted portion and an unsorted portion. Initially, the sorted part contains only the first element, and the rest are unsorted. The algorithm iterates through the unsorted part of the array. For each element, it compares it with the elements in the sorted part. If the current element is smaller than its predecessor in the sorted part, it keeps moving left until it finds its correct position. The greater elements are shifted one position up to make space for the swapped element.

For example, let's say we have an array: {12, 11, 13, 5, 6}. In the first pass, 12 and 11 are compared. Since 12 is greater, they swap positions. Now, 11 is stored in the sorted sub-array. The process continues, comparing and swapping elements until the entire array is sorted.

Insertion sort is simple to understand and implement. However, it's not efficient for large lists because its average and worst-case time complexity are both  $O(n^2)$ . For mostly sorted input, insertion sort performs better, but for larger or more disordered lists, other algorithms like merge sort are preferable.

```

FUNCTION InsertionSort(arrayToSort)
  position = 0
  WHILE position < arrayToSort.length
    currentValue = arrayToSort[position]
    sortedPosition = position - 1
    WHILE sortedPosition >= 0 and arrayToSort[sortedPosition] > currentValue
      arrayToSort[sortedPosition + 1] = arrayToSort[sortedPosition]
      sortedPosition = sortedPosition - 1
    END WHILE
    arrayToSort[sortedPosition + 1] = currentValue
    position = position + 1
  END WHILE
  RETURN arrayToSort
END InsertionSort

```

## Selection Sort

Selection sort is a simple and efficient sorting algorithm that works by repeatedly selecting the smallest (or largest) element from the unsorted portion of the list and moving it to the sorted portion of the list. The algorithm divides the list into two parts: a sorted region and an unsorted region. Initially, the sorted region is empty, and the unsorted region contains the entire list. In each iteration, the algorithm identifies the smallest element from the unsorted region and swaps it with the first element of the unsorted part. This process continues until the entire list is sorted.

For example, consider an array {64, 25, 12, 22, 11}. During the first pass, the smallest value (11) is swapped with the element at the first position. In subsequent passes, the next smallest elements are placed in their correct positions.

```

FUNCTION SelectionSort(arrayToSort)
  unsortedIndex = arrayToSort.length - 1
  WHILE unsortedIndex > 0
    i = 0
    max = arrayToSort[i]
    maxIndex = i
    WHILE i <= unsortedIndex
      i = i + 1
      IF arrayToSort[i] > max THEN
        max = arrayToSort[i]
        maxIndex = i
      END IF
    END WHILE
    temp = arrayToSort[maxIndex]
    arrayToSort[maxIndex] = arrayToSort[unsortedIndex]
    arrayToSort[unsortedIndex] = temp
    unsortedIndex = unsortedIndex - 1
  END WHILE
  RETURN arrayToSort
END SelectionSort

```

## Time Complexity for Common Algorithms

Algorithm	Time Complexity
Linear Search	$O(n)$
Binary Search	$O(\log(n))$
Bubble Sort	$O(n^2)$
Insertion Sort	$O(n^2)$
Selection Sort	$O(n^2)$

## Error Checking

### Types of Errors

- **Types of programming errors:**
  - **Syntax errors:** an error that is caused by program statements that do not conform to the rules of the language
  - **Run-time errors:** an error that occurs during the execution of a program that prevents the program from running correctly.
  - **Logic errors:** an error that occurs when the program executes however produces unexpected output. Logic errors are often the most difficult to debug and software requires substantial testing to ensure there are no logic errors.

### Desk Checking

- After creating an algorithm, it needs to be tested for logic errors to ensure that it will work as expected. One common way of doing this is to perform a **desk check** using a **trace table**

### Completing a Trace Table

1. Select test data that will test **all** possible paths through your program
2. Calculate what the expected output from your code should be for each piece of test data
3. Draw a table with all relevant variables and conditions
4. Walk through the algorithm using each piece of test data
5. Record the values for each variable/condition as you step through the code

### Example

Pseudocode	Trace Table			
<pre> 1 FUNCTION Main 2   INPUT(x) 3   WHILE x &lt; 18 4     x = x + 7 5   END WHILE 6   OUTPUT(x) 7 END Main </pre>	Line	X	X < 18	OUTPUT
	2	5		
	3		TRUE	
	4	12		
	3		TRUE	
	4	19		
	3		FALSE	
	6			19

## Exercises

### Question 1

Consider the following pseudocode:

```

1      FUNCTION Main
2          min = 100
3          total = 0
4          average = 0
5          FOR i = 1 TO 3 DO
6              INPUT(num)
7              total = total + num
8              IF min > num THEN
9                  min = num
10             END IF
11         END FOR
12         average = total / 3
13     END Main

```

Complete a trace table for the algorithm using the following test data for the variable *num*:

5      3      7

Line	Min	Total	Average	i	num	Min > num

**Question 2**

Refer to the following code snippet to answer the questions below.

```

BEGIN
    n = "Jeff"
    found = False

    FOR i 0 TO 4 DO
        IF names[i] == n THEN
            found = TRUE
            OUTPUT("Found")
        END IF
    END FOR

    IF found == False THEN
        OUTPUT("Not found")
    END IF
END

```

Use the following test data to complete the trace table for the algorithm in the table below.

names = ["Chris", "Ashley", "Graham", "Jeff", "Geoff"]

n	found	i	names[i] == n	Output	found == False

## Object-Oriented Programming

Object-oriented programming (OOP) programs are based around the data that is needed and the operations that need to be performed on that data, rather than the procedural logic of the program.

### Key Concepts

<b>Classes</b>	A class is a blueprint for creating objects, providing initial values for state (member variables) and implementations of behaviour (member functions or methods).
<b>Objects</b>	An object is an instance of a class. It can contain data in the form of fields (often known as attributes or properties), and code in the form of procedures (often known as methods).
<b>Attributes</b>	Attributes are characteristics of an object. They are data members inside a class or an object that represent the different features of the class.
<b>Methods</b>	In object-oriented programming (OOP), a method is a procedure associated with an object. It is the equivalent of a function in OOP.
<b>Abstraction</b>	Abstraction is a core concept of OOP that hides unnecessary details from the user and shows only essential attributes. It defines a model to create an application component.
<b>Polymorphism</b>	Polymorphism is a concept in OOP that describes situations where something, such as a method, can occur in several different forms. It allows objects of different types to be accessed through the same interface.
<b>Instantiation</b>	Instantiation in OOP refers to the creation of an object or an instance of a given class. It is the process of taking a class definition and creating an object that you can use in a program.
<b>Inheritance</b>	Inheritance is a mechanism in OOP that allows a class to inherit attributes and methods from another class, meaning it can reuse code by referencing the behaviours and data of the parent object.
<b>Encapsulation</b>	Encapsulation in OOP is the bundling of data and the methods that operate on that data within one unit, like a class. It is used to hide the values or state of a structured data object inside a class, preventing unauthorized parties' direct access to them.

To use objects in a program,, the first thing to do is define the properties that an object possesses (its **attributes**). For example, if we have an object of a person, we might want to know their first name, last name and email address. We then have to work out what things the object needs to be able to do (or actions that can be performed on that object). These are the **methods** that belong to the object. For example, a person might want to be able to update their email address and show their detail.

Once we have defined an object, we can use **inheritance** to create new types of objects based on the original base object. For example, we might want to create two new objects called Student and Teacher, which share the same base properties as a Person. That is, both Students and Teachers will have a first name, last name and email address, and be able to update that email address. A student, however, may also have a house (or faction), year and list of subjects that are being studies. A teacher might have a house (or faction), the subject they teach and a code. When printing the details of Student and Teacher, they might override the original method to show the details in a different way. This is called **polymorphism**.

Once we have defined our classes, we can then use them in a program by instantiating an object. That is, we can use the class definition of a Student create a new student object that has the name of James Peterson. We can also use the same class definition to create a second student with the name Jane O'Halloran. Each instance of a class is called an **object**.

## Sample Pseudocode

### *Defining a class*

CLASS Person

Attributes:

first\_name  
last\_name  
email

Methods:

```
FUNCTION Person(fname, lname)
    first_name = fname
    last_name = lname
    email = ""
END Person

FUNCTION update_email(new_email)
    email = new_email
END update_email

FUNCTION show_details()
    PRINT(last_name + ", " + first_name)
END show_details
```

### *Using Inheritance to Define a Student (based on a Person)*

CLASS Student : Person

Attributes:

house  
year  
subjects = []

Methods:

```
FUNCTION Student(fname, lname, student_year, student_house)
    Person(fname, lname)
    house = student_house
    year = student_year
END Student

FUNCTION add_subject(subject)
    subjects.append(subject)
END add_subject

FUNCTION show_details()
    PRINT(last_name + ", " + first_name)
    PRINT("Year " + year + " " + house)
    FOR i = 0 TO subjects.length - 1
        PRINT(subjects[i])
    END FOR
END show_details
```

### *Using Inheritance to Define a Teacher (based on a Person)*

CLASS Teacher: Person

Attributes:

house  
subject  
code

Methods:

```
FUNCTION Teacher(fname, lname, new_house, new_subject, new_code)
    Person(fname, lname)
    house = new_house
    subject = new_subject
    code = new_code
END Teacher

FUNCTION show_details()
    PRINT(last_name + ", " + first_name + " (" + code + ")")
    PRINT("Teaches " + subject)
END show_details
```

### Defining a Class for each Class at School

CLASS Class

Attributes:

subject  
teacher  
students = []

Methods:

```
FUNCTION Class(class_subject, class_teacher)
    subject = class_subject
    teacher = class_teacher
END Class

FUNCTION add_student(student)
    Students.append(student)
END add_student

FUNCTION add_multiple_students(new_students)
    FOR i = 0 TO new_students.length - 1
        students.append(new_students[i])
    END FOR
END add_multiple_students

FUNCTION print_class_list()

END print_class_list
```



### *Instantiating Objects*

```
james = new Student("James", "Smith", 10, "Kangaroo")
```

```
bella = new Student("Bella", "Jones", 10, "Koala")
```

```
peter = new Teacher("Peter", "Stirling", "Koala", "English", "PGS")
```

```
james.add_subject("English")
```

```
james.add_subject("Maths")
```

```
bella.add_subject("English")
```

```
bella.add_subject("Chemistry")
```

```
class = new Class("English", peter)
```

```
class.add_multiple_students([james, bella])
```

### Exercises

#### *Question 1*

Consider the following code:

CLASS Number

Attributes:

sign

magnitude

Methods:

FUNCTION Number(new\_sign, new\_magnitude)

sign = new\_sign

magnitude = new\_magnitude

END Number

FUNCTION Add(m)

magnitude = magnitude + m

IF magnitude < 0 THEN

sign = FALSE

magnitude = magnitude \* (-1)

ELSE

sign = True

END IF

END Add

END Number

(a) With reference to the above class, explain the purpose of attributes in OOP.

---



---



---



---



---

(b) With reference to the above class, explain the purpose of methods in OOP.

---

---

---

---

---

(c) Explain the difference between a class and an object.

---

---

---

---

---

(d) Instantiate a new number called *my\_num* with a sign of True and a magnitude of 11.

---

---

(e) Add 5 to *my\_num*

---

---

(f) Print out the sign and magnitude of *my\_num*

---

---

## Question 2

Consider the following class:

CLASS Animal

Attributes:

```
name
hunger = 5
food_list = []
```

Methods:

```
FUNCTION Animal(new_name, foods)
    name = new_name
END Animal

FUNCTION eat(food)
    result = ""
    IF food IN food_list
        result = "Not hungry"
        IF hunger > 0
            hunger = hunger - 1
            result = "That was yummy"
        END IF
    ELSE
        result = "I don't like that food"
    END IF
    RETURN result
END eat

FUNCTION is_hungry()
    RETURN hunger > 0
END is_hungry
```

END Animal

(a) Instantiate a new animal with the name *Roxy* that eats *chicken, beef* and *cheese*

---

---

---

(b) Check to see if the animal is hungry. If it is hungry, get it to eat some chicken.

---

---

---

(c) Create a new class called *Fish* that is based on *Animal*. Your new class should:

- Include the attribute *has\_fins* and set it to True
- Have a food list of *algae* and *plankton*
- Include a method called *swim* that will return the sting “<name> is swimming”

This image shows a single sheet of white paper with horizontal ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

(d) Instantiate a new fish called *Scaly* and ask it to eat some *grass*. You should then get it to swim.

---

---

---

---

# Networks and Communication

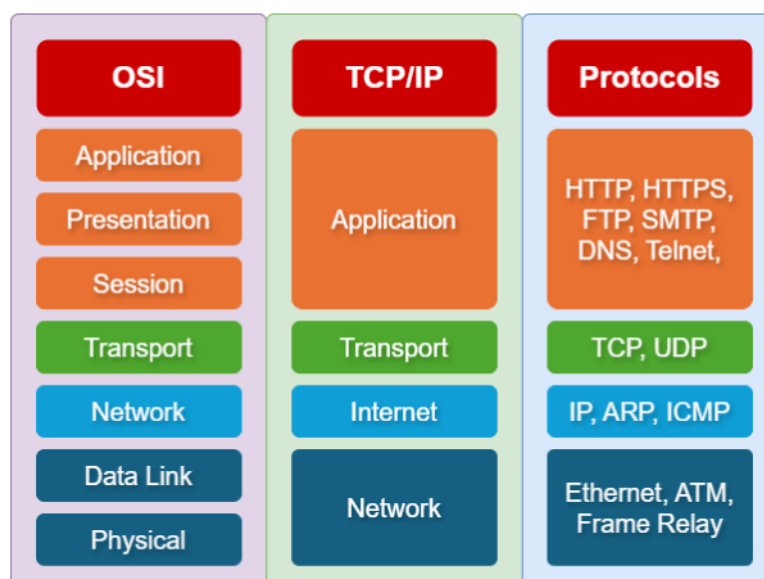
## Terminology

The following are some general networking terms that you should know:

- Protocol
- Bandwidth
- Broadcast traffic
- Transmission medium
- Data collision
- Attenuation
- Latency
- Bit rate

## Models of Networking

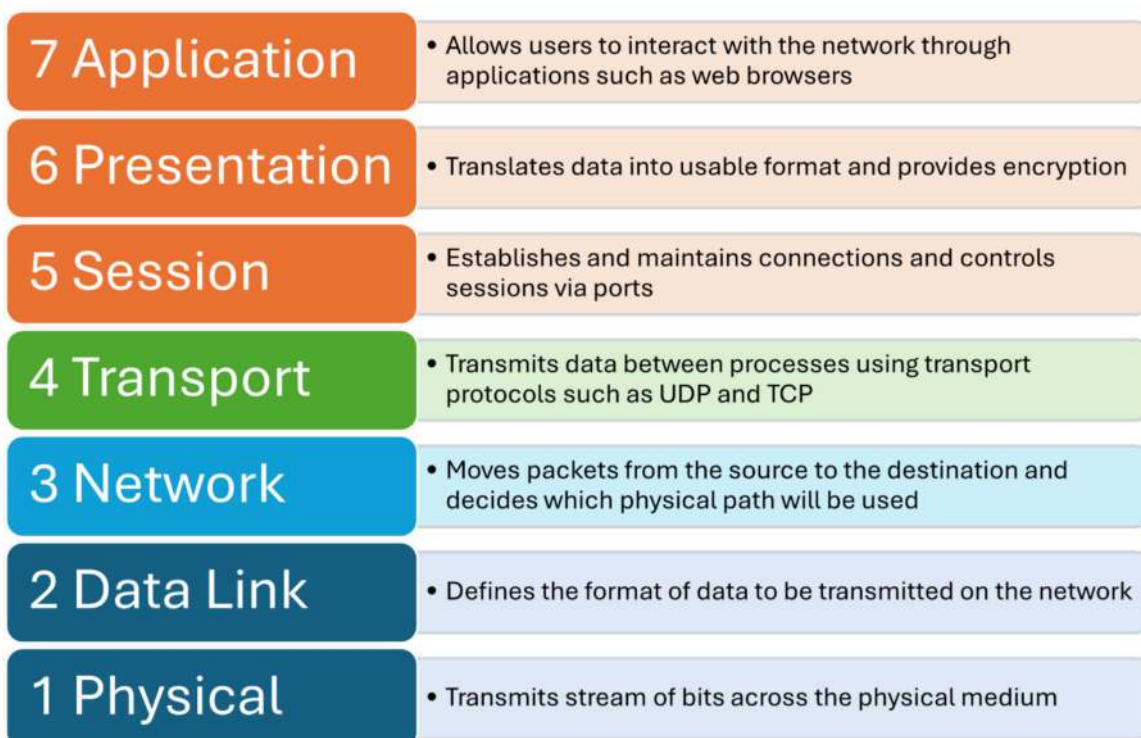
- There are two models that are used in the ATAR course:
  - Open Systems Intercommunication (OSI) Model
  - TCP/IP Model
- These models provide a framework that describes the steps required to transmit data between devices across a network and between networks
- Each layer describes a different step in the process, starting with physically transferring data between devices through sending to the correct location and finally providing a method for applications (such as a web browser) to interact with the network
- Key differences include:
  - Number of layers: OSI model has 7 layers, TCP/IP model has 4
  - Function of layers: Each OSI layers has a distinct function, whereas the TCP/IP model combines several layers of the OSI model to reduce the number layers to 4
  - Adoption: OSI model is not widely implemented and is a more theoretical concept, whereas the TCP/IP model is the foundation of the Internet and most modern networks.



## OSI Model

- purpose of OSI model
- layers of OSI model
- role of layers within the model
- MAC address layer 2 switching
- IP address layer 3 routing

- Provides a conceptual framework for understanding how different networking protocols and technologies interact within a network to allow communication between devices.
- Splits the required functionality to send a message into 7 distinct layers, each responsible for a single aspect of communications, such as addressing, routing, error detection and data formatting.
- Each layer is self-contained and operates independently of the others.
- The following diagram shows all 7 layers and their purpose:



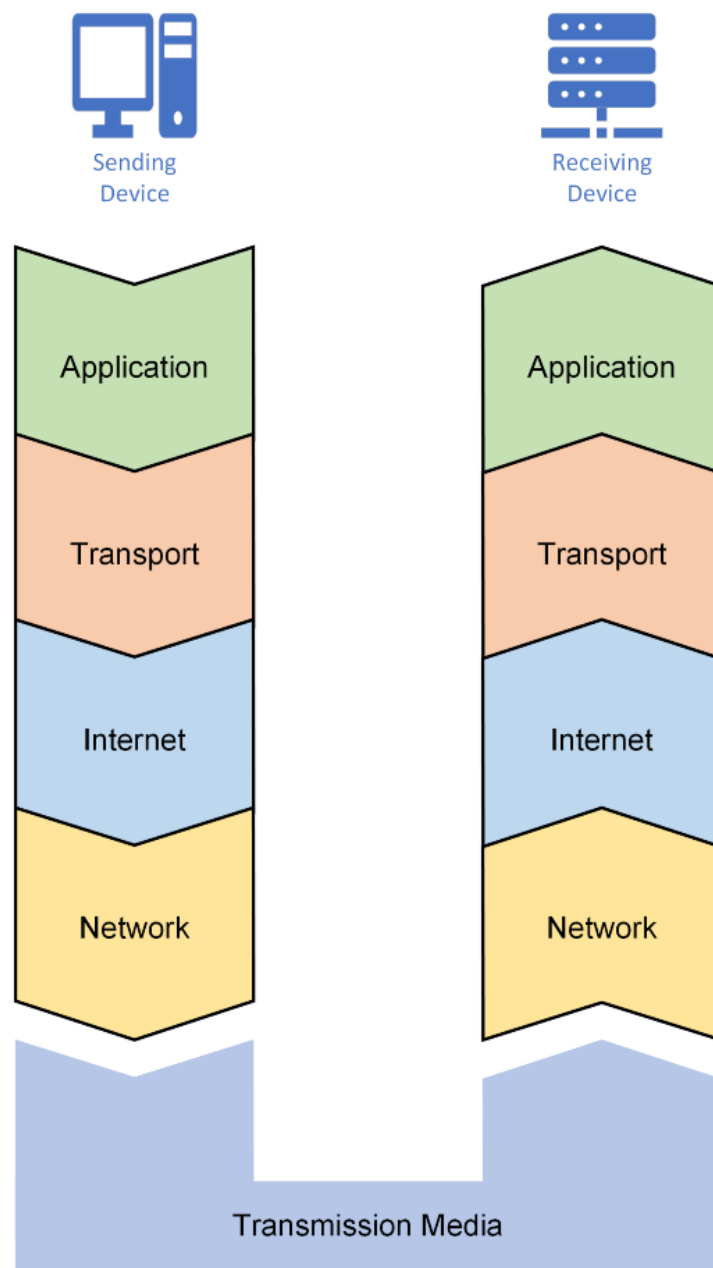
## TCP/IP Model

- purpose of the DoD model
- layers of the DoD model
  - application
  - transport
  - internet
  - network
- role of layers within the model

- Provides a practical framework to create a network and provides specific protocols that should be used to facilitate network communication.
- One of the key advantages of using the TCP/IP model is that it ensures that people using different systems are still able to communicate. If they use the correct protocols for each stage of communication, it does not matter what software or TCP/IP stack is used.
- Each layer operates independently, so that provided each layer uses the correct protocols to interact with the other layers, it does not matter how that individual layer has been implemented (or how it layer completed its job). For example, as long as the Network layer passes IP packets to the Internet layer, it does not matter what technology the Network layer uses to pass the data from one place to another.

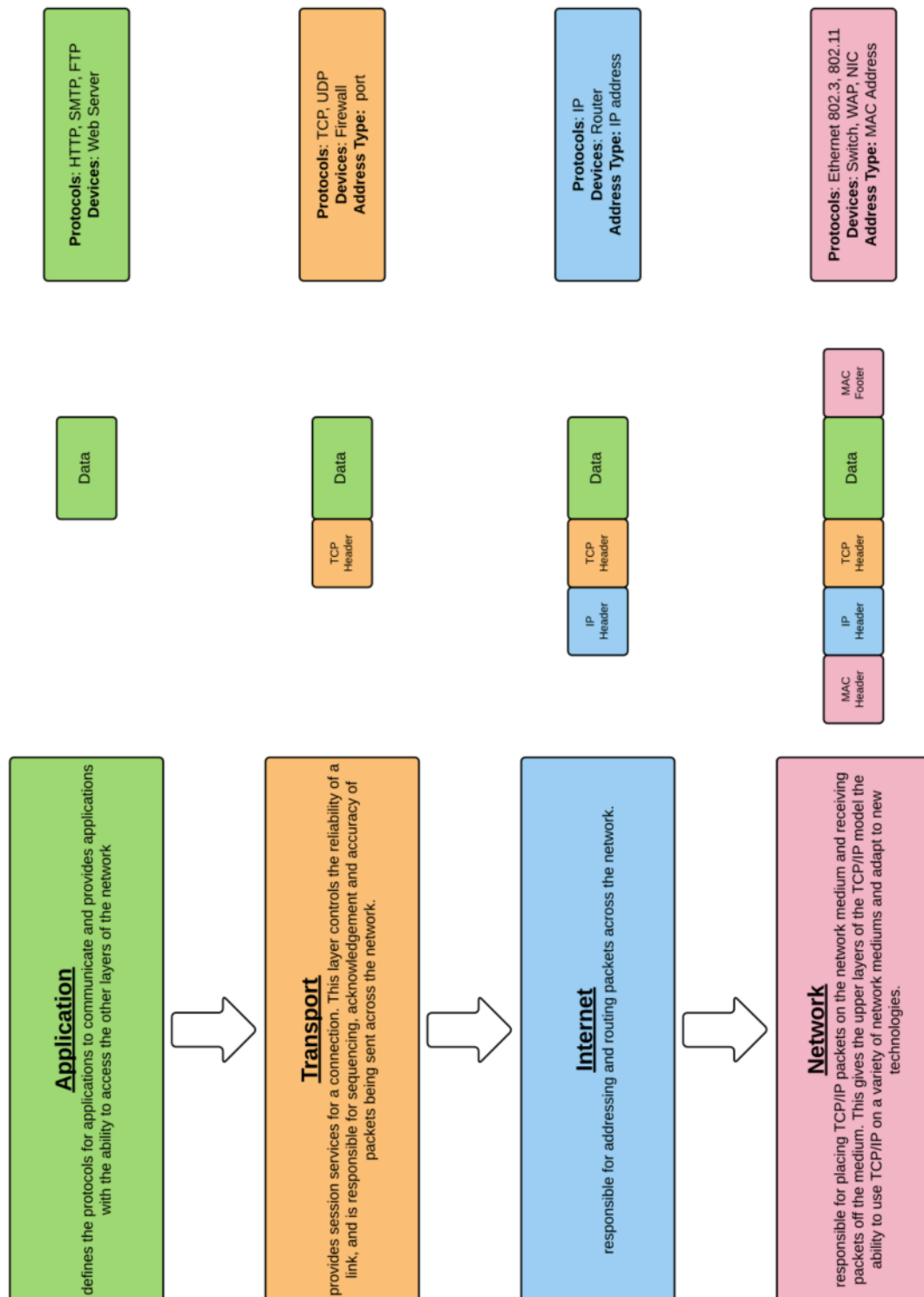
## *TCP/IP Layers*

- There are four layers to the TCP/IP model:
  - Application
  - Transport
  - Internet
  - Network
- NOTE: You will see these layers named differently in various textbooks and websites. Make sure you **use the layer names that are specified in the syllabus**.
- Each layer is responsible for a specific aspect of network communications. See the graphic below to see what each layer does
- Different protocols operate at different layers and define how the network should perform different functions.
- Tips to remembering how it all ties together:
  - Learn what the main protocols do and what layer they operate
  - Learn what devices operate on each layer
  - Learn what address types are used on each layer and with different protocols
  - Learn what address type each device uses – this will help you remember what layer each device operates on

**Flow of data through the layers:**



# TCP/IP Model of Networking



## Internet Protocol

- Internet Protocol (IP) is one of the key protocols that network communications are built on. Its main function is to facilitate the delivery of packets from a source host to a destination host across interconnected networks, enabling packets of data to be routed across multiple networks.

### IP4 vs IP6

- IPv4 (Internet Protocol version 4) and IPv6 (Internet Protocol version 6) are two different versions of the Internet Protocol, each designed to facilitate communication between devices over the internet. IPv4 was the original version that has been widely adopted, however has significant limitations that have been addressed in IPv6.
- Key features and differences are outlined in the following table:

	IPv4	IPv6
Address Length:	Uses 32-bit addresses, allowing for approximately 4.3 billion unique addresses.	Uses 128-bit addresses, providing a virtually unlimited number of unique addresses (about 340 undecillion).
Address Representation:	Expressed in dotted-decimal notation, with each 8-bit segment represented by a decimal number (e.g., 192.0.2.1).	Expressed in hexadecimal notation, with each 16-bit segment separated by colons (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
Address Configuration:	Addresses are typically configured manually or dynamically assigned using protocols like DHCP (Dynamic Host Configuration Protocol).	Supports multiple methods of address configuration, including stateless autoconfiguration (SLAAC), DHCPv6 (Dynamic Host Configuration Protocol version 6), and manual configuration.
Address Types:	Supports unicast, multicast, and broadcast addresses.	Supports unicast, multicast, and anycast addresses. IPv6 does not use broadcast addresses; instead, multicast is used for similar purposes.
Header Format:	Has a variable-length header (20 to 60 bytes), including fields such as version, header length, type of service, total length, identification, flags, fragment offset, time-to-live, protocol, header checksum, source address, and destination address.	Has a fixed-length header (40 bytes), including fields such as version, traffic class, flow label, payload length, next header, hop limit, source address, and destination address. Additional options are supported in extension headers.
Header Checksum:	Includes a header checksum field for error detection in the header.	Removes the header checksum field. Error detection is handled by higher-layer protocols.
Security:	Security features such as IPsec (Internet Protocol Security) are optional and often implemented as additional protocols.	IPsec support is mandatory, providing built-in security features for authentication, confidentiality, and integrity.
Fragmentation:	Supports packet fragmentation at routers when the packet size exceeds the Maximum Transmission Unit (MTU) of a network link.	Generally avoids fragmentation by relying on Path MTU Discovery to determine the optimal packet size for end-to-end communication. If fragmentation is necessary, it is performed by the source host.

## Public vs Private IP Addresses

- IP addresses can be divided into two categories:
  - **Public IP addresses:** globally unique addresses that are routable on the Internet and are given to devices that are directly accessible on the Internet (such as web servers and network gateways)
  - **Private IP addresses:** uses within a private network. They must be unique within that network, but different networks may use the same address range. Some common address ranges are:
    - Class A: 10.0.0.0 to 10.255.255.255
    - Class B: 172.16.0.0 to 172.31.255.255
    - Class C: 192.168.0.0 to 192.168.255.255

## Subnetting

- Subnetting is the process of dividing a network into smaller subnetworks, known as **subnets**.
- Some advantages of segmenting a large network into multiple subnets include:
  - Improved network performance by reducing network congestion
  - Improved security by preventing devices on different subnets communicating and allowing different security rules to be applied to each subnet
- **Useable IP Addresses:**
  - For each subnet, the number of **useable** IP addresses is 2 less than the total number of addresses available. That is because the first address in each subnet range is used for the network address, and the last address is used for the broadcast address.
  - The **first** useable IP address is one more than the network address
  - The **last** useable IP address is one less than the broadcast address
  - Consider the network 172.20.0.0 with the subnet mask 255.255.0.0
    - Network address: 172.20.0.0
    - Broadcast address: 172.20.255.255
    - Useable IP address range 172.20.0.1 – 172.20.255.254

## Subnet Mask

- Each IP address can be split into two portions:
  - The network address – the first part of the address
  - The host address – the second part of the address
- Although an IP address is usually written in decimal numbers, it actually refers to a 32-bit binary number, hence each part of the IP address represents one byte and is therefore a number between 0 and 255. This is important to keep in mind when working out subnets and subnet masks.
- The subnet mask indicates how many bits of the address belong to the network, and how many bits belong to the host. The network portion is indicated with a 1, and the host portion is indicated with a 0. Hence, a subnet mask of 255.255.0.0 indicates that the first 16 bits belong to the network address, and the last 16 bits belong to the host.
- Typical subnet masks:
  - Class A: 255.0.0.0
  - Class B: 255.255.0.0
  - Class C: 255.255.255.0
- The subnet mask can also be represented in terms of **CIDR** which indicates the number of bits that are being used to define the network address. For example, the address 192.168.1.100/24 indicates that the first 24 bits of the address are the network address, and the next 8 bits are the host address. Thus, the CIDR /24 is equivalent to the subnet mask of 255.255.255.0.

## Calculating Subnets

- It is possible that you will be asked to calculate the IP address range for a given number of subnets on a network. For example, given the network address 192.168.25.0 and subnet mask 255.255.255.0, split the network into 4 equal subnets.
- Steps required:
  - **Step 1:** Identify the network address for the base network
  - **Step 2:** Identify the subnet mask for the base network
  - **Step 3:** Identify the number of subnets to create
  - **Step 4:** Calculate the total number of addresses available in each subnet
  - **Step 5:** Calculate the network address for each subnet
  - **Step 6:** Calculate the broadcast address for each subnet
  - **Step 7:** Calculate the subnet mask for each subnet
  - **Step 8:** Calculate the first available host address for each subnet
  - **Step 9:** Calculate the last available host address for each subnet

**Example:**

You have been hired to help set up a network for a small graphic design company. You have been given the network address 192.168.45.0 and subnet mask of 255.255.255.0 and they have asked for the network to be divided into 4 equal subnets.

For each subnet, find the network address, broadcast address, first host address, last host address and the subnet mask.

- **Step 1:** The base network address is 192.168.45.0
- **Step 2:** The base subnet mask is 255.255.255.0
- **Step 3:** There are 4 subnets required
- **Step 4:** To find the number of available addresses, use the following table:

Number of subnets	1	2	4	8	16	32	64	128	256
Available addresses	256	128	64	32	16	8	4	2	1
Subnet mask	0	128	192	224	240	248	252	254	255
CIDR	/24	/25	/26	/27	/28	/29	/30	/31	/32

We want 4 subnets, so we have 64 available addresses.

- **Step 5:** The network address is found by starting with the first available IP address (in this case 192.168.45.0) and then adding the number of available addresses in each subnet (in this case 64). Therefore, the network address for each subnet would be:
  - Network A: 192.168.45.0
  - Network B: 192.168.45.64
  - Network C: 192.168.45.128
  - Network D: 192.168.45.192
- **Step 6:** The broadcast address is the last available address in the range for each subnet, which is the address before the next subnet starts. Therefore, the broadcast addresses would be:
  - Network A: 192.168.45.63
  - Network B: 192.168.45.127
  - Network C: 192.168.45.191
  - Network D: 192.168.45.255
- **Step 7:** To calculate the new subnet mask, we need to take the first 2 bits from the original host address space, which will give us the binary 11000000. This converts to 192 in decimal, therefore the new subnet mask for each subnet is 255.255.255.192. The CIDR representation of this is /26 (indicating the first **26** bits are being used for the network address)
- **Step 8:** The *first* available host address for each network is the one *after* the network address. So for these subnets, the first useable address is:
  - Network A: 192.168.45.1
  - Network B: 192.168.45.65
  - Network C: 192.168.45.129
  - Network D: 192.168.45.193
- **Step 9:** The *last* available host address for each network is the one *before* the broadcast address. So for these subnets, the last useable address is:
  - Network A: 192.168.45.62
  - Network B: 192.168.45.126
  - Network C: 192.168.45.190
  - Network D: 192.168.45.254

## Exercises

### *Question 1*

For each layer of the TCP/IP model, identify one protocol and use that to describe the purpose of that layer.

Layer	Protocol	Purpose
Application		
Transport		
Internet		
Network		

### *Question 2*

TCP and IP are two important protocols that form the basis of most Internet communications.

(a) Explain the purpose of TCP.

---

---

---

---

---

---

---

---

(b) Explain the purpose of IP.

---

---

---

---

---

---

### Question 3

Oscar has been hired to run an international eSports tournament featuring the latest fast-paced first-person shooter game. The event is being live streamed to millions of viewers worldwide. The game's performance is highly dependent on the real-time transmission of data, where even a millisecond of delay can affect the outcome of a match. Oscar has been instructed to ensure that the game's networking is robust, allowing for a seamless experience for both the players in the tournament and the audience watching the live stream.

(a) State the most appropriate transport layer protocol to be used for the live streaming of the event.

---

(b) Explain the key differences between the purpose of UDP and TCP.

---

---

---

---

---

---

---

---

---

(c) Outline **two** differences between the TCP and UDP packet architecture.

One: \_\_\_\_\_

---

Two: \_\_\_\_\_

---

### Question 4

You have been a range of IP addresses starting with 192.168.201.0 and a default subnet mask for a class C network of 255.255.255.0.

To improve the efficiency of the network, you have decided to split the network up into 8 equal subnets. Complete the details below for the network:

Subnet Mask: \_\_\_\_\_

CIDR: \_\_\_\_\_

Number of IP addresses per subnet: \_\_\_\_\_

Number of usable IP addresses per subnet: \_\_\_\_\_

	Network address	Broadcast address	First host address	Last Host address	Next network
Network A					
Network B					
Network C					
Network D					
Network E					
Network F					
Network G					
Network H					



### Question 5

The Templeton Police Department looks after an urban metropolitan suburb in the greater Melbourne area. They plan on building a new command centre with advanced surveillance capabilities, including a high-definition CCTV network, confidential databases for criminal records, and a communication hub for emergency dispatch.

Distinct network segments must be created for the following departments: Criminal Investigations, Emergency Response and Administrative Services. Each department's data access needs are unique, and their activities range from accessing confidential databases to streaming real-time surveillance footage.

Using the network base address 192.168.50.0/24, you are required to create a subnetting scheme to accommodate the three departmental networks within the command centre. Each subnet should be evenly sized, ensuring efficient utilisation of IP addresses while maintaining network segregation and security.

#### Subnetting Details:

- Divide the 192.168.50.0/24 network into four equal subnets.
- Assign the first subnet to the Administrative Services Department.
- Assign the second subnet to the Emergency Response Department.
- Assign the third subnet to the Criminal Investigations Department (CID).
- The fourth subnet will remain unused and should be reserved for future expansion.

	Network Address	Subnet Mask
Administrative services		
Emergency response		
Criminal investigations		

---

## Network Components

---

- role of components at different network layers:
  - transmission media
  - router
  - gateway
  - switch
  - wireless access point
  - firewall

### Device Roles

- **Transmission media:**
  - The transmission media is the method used to physically transfer data between devices. There are two main types of transmission media:
    - Wired: data travels over a physical connection between two devices
    - Wireless: data travels through the air between devices, which are not physically connected to each other
- **Router:**
  - A router is a network device that connects two or more networks together and directs packets of data between them. Routers form an integral part of the Internet as they determine the best path for each packet of data to take to reach its destination. They use the IP address on data packets to determine the destination and the best path to direct data.
- **Switch:**
  - A switch joins multiple devices together to form a single local area network (LAN). A switch stores the MAC address for each device that is connected to it, so when it receives a piece of data (called a **frame**), it can determine which port it needs to direct the frame to.
- **Firewall:**
  - A firewall is a device that restricts traffic both in and out of a network and helps to prevent unauthorised users gaining access to a network. It can operate either as a hardware device or as a software program (or both).
- **Wireless access point (WAP):**
  - A wireless access point (WAP) allows devices to connect to a LAN without the need to physical cables. The WAP connects to a router or switch using wires, and then projects a Wi-Fi signal that other devices can use to connect to the network.
- **Gateway:**
  - A gateway is a network node that serves as an access point to another network. This can involve a change of addressing or different networking technology. Routers and other computers such as servers are often used as gateway devices.

## Other Common Devices (not in current syllabus)

- **Network interface card (NIC):**
  - A Network Interface Card (NIC) allows a device to connect to a network using a specific transmission medium. For example, an Ethernet card allows a twisted pair cable to plug into the device and communicate using the Ethernet standard. A device needs a NIC for each transmission medium or network standard that it uses. That is, it needs a different NIC for ethernet, wireless and Bluetooth.
- **Modem:**
  - A modem (or **mod**ulator/**dem**odulator) is a device that allows networks to connect to each other over the phone lines by converting data from digital to analogue signals and vice versa.
- **Repeaters:**
  - Repeaters help to extend the range of transmissions by retransmitting a signal they receive to devices farther away that might otherwise not be able to receive a transmission due to signal loss.

## Transmission Media

### *Wired vs Wireless*

Wired	Wireless
<ul style="list-style-type: none"> <li>• Increased security as need to physically connect to wire to access signal</li> <li>• Usually, greater distances can be covered</li> <li>• Usually, higher bandwidth</li> </ul>	<ul style="list-style-type: none"> <li>• Cannot prevent people from intercepting signal, therefor potentially less secure</li> <li>• More susceptible to atmospheric conditions</li> <li>• Gives greater flexibility</li> <li>• Requires less infrastructure as don't need cables, switches for each device</li> </ul>

### *Wired Transmission Media*

- **Unshielded Twisted Pair (UTP):** Thin copper wire arranged in pairs that are twisted around each other to minimise interference
- **Fibre Optic:** Thin strand of glass (or plastic) that transmit digital data using light that uses total internal reflection to bounce light down the glass strand

## Exercises

### *Question 2*




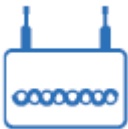





Complete the table below with information about various devices used within a network. Describe the role of the device, which layer of the TCP/IP model it operates at and what type of addressing it uses.

Device	Role	TCP/IP Layer	Address Type
Switch			
Router			
NIC			
Firewall			

## Network Diagrams

- Interpret and create network diagrams using specified CISCO conventions to represent network topologies, considering addressing, subnets, segmentation, security and performance
- A network diagram shows the logical design of a network and how the devices should be connected.
- Generally, the order of devices would be:
  - ISP/Internet/Cloud
  - Modem (if required)
  - Firewall
  - Router
  - Switch
  - WAP / Servers / End devices (such as desktop computers and laptops)
- The WA ATAR course requires the use of CISCO symbols when drawing a network diagram.
- NOTE:** For the purposes of the WA ATAR course, firewalls should be placed outside the trusted network and don't require an IP address.

### CISCO Symbols

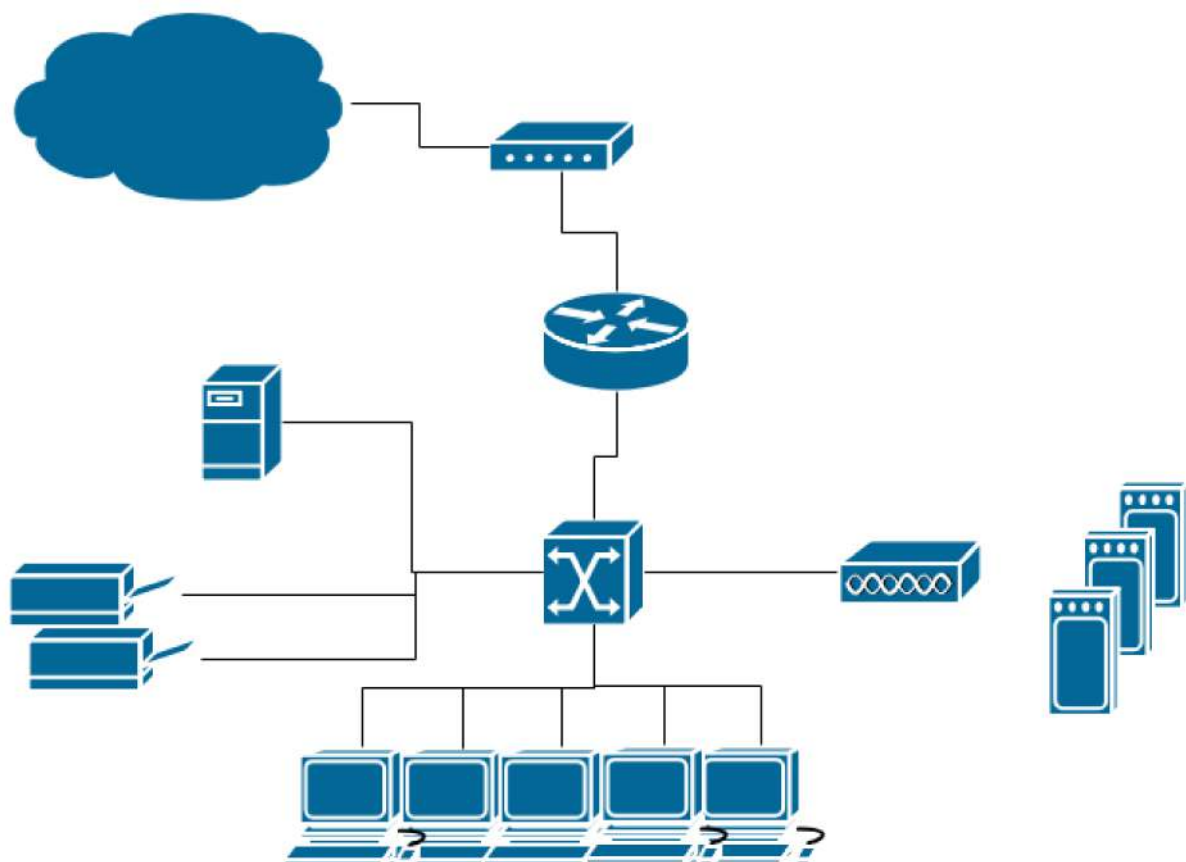
Internet/ISP		Firewall	
Router		Wireless access point (WAP)	
Modem		Database server	
Switch		File server	
End devices			

### Example

SnakeOil Cars has decided that it is time to modernise their systems and build a new computer network. Their new network will need the following features:

- Internet access via a broadband router with a built-in firewall.
- A combined File and Database server located in the electrical room
- Two network aware photocopiers, one located in the service reception area and one in the hallway.
- UTP connections for 5 desktop workstations used by the receptionist, Service Counter, Service Manager, Finance & Insurance consultant and Vehicle Sales Manager.
- Three tablet computers used by the Sales consultants. The consultants will use the tablets to show customers available options and check vehicle stock as they are dealing with customers.

Using CISCO conventions, draw a network diagram to show how SnakeOil Cars should set up their network. You do not need to show the room layout of on your diagram, only the devices and transmission media.



## Exercises

### *Question 1*

The warehouse has an office located at the front of the building where a receptionist can greet customers and coordinate the work from a desktop workstation. The database server is located in a locked storage room at the rear of the office, and there is a shared network printer in the corner. There is also a small office where the warehouse manager has a desktop workstation from where she can access the network. The entire network has access to the Internet through a shared connection to their ISP and is protected by a firewall. d) Draw a network diagram to show how the various devices are connected, clearly labelling each device. Label any necessary transmission media, network communications standards and/or network control protocols that would be used with the various parts of the network.

## Question 2

The Templeton Police Department looks after an urban metropolitan suburb in the greater Melbourne area. They plan on building a new command centre with advanced surveillance capabilities, including a high-definition CCTV network, confidential databases for criminal records, and a communication hub for emergency dispatch.

Distinct network segments must be created for the following departments: Criminal Investigations, Emergency Response and Administrative Services. Each department's data access needs are unique, and their activities range from accessing confidential databases to streaming real-time surveillance footage. You are tasked with designing a comprehensive network infrastructure for the Templeton Police Department's new command centre.

### Network Requirements:

- **Central Network Configuration:**
  - One central router is connected to the internet through a robust firewall to protect the entire network from malicious attacks.
  - Each department connects to the router using different subnets to ensure secure communication and data segregation.
- **Administrative Services Department Subnet:**
  - Two desktop computers for two officers at the reception area of the building to handle administrative tasks.
  - Include a wired printer for administrative documentation.
- **Emergency Response Department Subnet:**
  - Five desktop computers are used by operators to handle emergency dispatches.
- **Criminal Investigations Department Subnet:**
  - One wireless access point to support three notebooks, facilitating mobile access to criminal databases.
  - Includes a dedicated database server to host confidential case files.
  - Create a network diagram using Cisco networking icons that interconnect the areas of the command centre, ensuring optimal network performance and reliability. Include all necessary network devices and host devices.



### Question 3

Vikki is the network administrator at White Wilderness – an adventure company specialising in family tours to Antarctica. They have fitted a cruise ship out with a computer network to provide guests and crew with Internet access and computing facilities whilst away. The ship's network consists of the following:

- Six computers on the main deck floor forming their own LAN segment where guests can get Internet access and process their photos
- Several handheld devices that visitors can use on the vessel to access email and view video of the local wildlife
- Two computers in the reception desk for staff to process guests when they board the ship and keep track of any special guest requirements
- Three computers in the office area for the crew
- Three laptops that the ship's officers use to connect to the network wirelessly
- One printer that all crew connected to the network can use

# Cyber Security

- To keep network communications secure, it is necessary to understand what some of the threats to network security are and methods to keep a network secure.

---

## Network Threats

---

### External Network Threats

- These are threats to network security that come from outside the network, usually from people trying to gain unauthorised access to either the network or information on the network.
- **Social engineering (phishing)**
  - Use social interactions to trick a person into giving out information that they shouldn't and that may compromise network credentials
  - Phishing usually involves attacker sending an email, SMS or social media message asking to confirm personal details or click on a link
- **Denial of service (including distributed denial of service – DDOS)**
  - Purpose is to prevent access to network resources
  - Common methods include:
    - Send large number of requests to a web server
    - Send many (or large size) emails to an account
  - Servers cannot process all the requests so the system crashes or runs slowly
- **Back doors**
  - Method to access a computer and/or network that bypasses the normal security measures
  - Sometimes created by developers for testing/maintenance of systems
  - Can also be installed through malware to allow attackers to exploit the system
- **IP spoofing**
  - Attackers can hide their identity and gain access to a network by pretending they are from a trusted source
  - IP packets are modified so that the source IP address matches a trusted IP source
- **SQL Injection**
  - Attackers insert malicious SQL code into the input fields of a web application to manipulate or access the database.
  - May allow attackers to retrieve, modify, or delete data, or even gain administrative privileges, leading to unauthorised access or data breaches:
- **Man-in-the-middle**
  - An attacker intercepts, and potentially alters, messages being sent between two devices, usually between a client and a server.
  - Sensitive information, such as login credentials or personal data, can be stolen or manipulated.
- **Cross-site scripting**
  - Attackers insert malicious JavaScript code into web applications that are viewed by other users. The code is then executed when viewed by an unsuspecting user.
  - May enable attackers to steal cookies and/or session tokens, allowing them to perform actions on behalf of users or access sensitive information.
- **Types of malware**
  - Malware is software that has been specifically designed for malicious purposes, such as locking files or secretly monitoring a network.

- Some common types of malware include:
  - viruses
  - worms
  - trojan horses
  - spyware
  - adware
  - ransomware
- **Physical network threats**
  - Network infrastructure is vulnerable to physical damage or attackers physically gaining control of devices. For example:
    - Unauthorized physical access to network devices: Attackers gaining direct access to routers, switches, or servers can tamper with or steal hardware.
    - Environmental hazards: Risks like fires, floods, or power surges that can damage physical network infrastructure and disrupt operations.
- **Zero day vulnerabilities**
  - Flaws in software or hardware that are unknown to the vendor and are able to be exploited before the vendor becomes aware of them and can issue a fix.
  - Since no patches or mitigations are available at the time of the attack, these vulnerabilities can be highly dangerous and impactful.

### Internal Network Threats

- There are a number of ways that networks can be compromised from within the network environment, both accidentally and deliberately
- **Lost or stolen devices:**
  - Unauthorized individuals may gain access to sensitive data stored on the device, compromising the network.
  - Devices can be exploited to bypass security controls, leading to potential data breaches.
- **Compromised credentials:**
  - Attackers can gain access to critical systems and data using stolen or guessed passwords.
  - It allows for impersonation of legitimate users, enabling further malicious activities within the network.
- **Misuse by employees:**
  - Employees may intentionally or unintentionally access, modify, or share sensitive data without proper authorisation.
  - Insider threats can lead to data leaks, intellectual property theft, or disruption of network services.

## Security methods

- **Analysis of log files:**
  - Identifies unusual or suspicious activity by reviewing logs for anomalies and potential breaches.
  - Helps in post-incident investigation and root cause analysis to strengthen security measures.
- **Anti-malware:**
  - Detects and removes malicious software that can compromise system security and data integrity.
  - Provides real-time protection against new and emerging threats through signature updates and behavior analysis.
- **Firewall filtering:**
  - Controls incoming and outgoing network traffic based on predefined security rules to block unauthorized access.
  - Prevents unauthorized access to internal systems by filtering traffic at the network perimeter.
- **Access control lists (ACLs):**
  - Restricts network access by defining who or what can access certain resources and services.
  - Limits exposure to sensitive data by assigning permissions based on user roles and needs.
- **Intrusion Prevention Systems (IPS):**
  - Monitors network traffic in real-time to detect and block suspicious activities or known threats.
  - Automatically responds to attacks by blocking or quarantining malicious traffic before it reaches the network.
- **Virtual Private Networks (VPNs):**
  - Encrypts data transmitted over public or untrusted networks, ensuring secure remote access to the network.
  - Protects user privacy by hiding IP addresses and masking online activities from external observers.
- **User training:**
  - Educates employees on best security practices, such as recognizing phishing attacks and safe browsing habits.
  - Reduces the likelihood of human error leading to security breaches by raising awareness of potential threats.
- **ICT code of conduct:**
  - Establishes clear guidelines on acceptable use of network resources and behavior related to data security.
  - Enforces accountability by outlining consequences for violations of security policies and procedures.
- **Physical security:**
  - Protects network infrastructure and devices from unauthorized access or damage through secure access controls, such as locks and surveillance.
  - Prevents physical tampering, theft, or sabotage of critical hardware, such as servers and network equipment.

## Exercises

### *Question 1*

Vikki is responsible for network security at a local hospital and has decided to implement new password and network user policies to help ensure the security of the network. Describe three policies that could be implemented to help keep the network secure.

i \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

ii \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

iii \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

### *Question 2*

Explain each of the following network vulnerabilities:

Denial of Service (DoS) \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

IP Spoofing \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

### Question 3

(a) Explain the term 'backdoor' as it applies to network security.

---

---

---

---

---

(b) Describe a situation where using a backdoor could be considered ethically sound.

---

---

---

---

---

### Question 4

Outline how the following network threats can affect an organisation.

Man in the middle \_\_\_\_\_

---

---

---

---

Cross-site scripting \_\_\_\_\_

---

---

---

---

Zero-day vulnerabilities \_\_\_\_\_

---

---

---

---

### Question 5

Discuss the significance of employee misuse as an internal network threat and provide an example of actions that constitute misuse.

---

---

---

---

---

Example: 

---

---

---

---

# Cryptography

---

## Key Points

- **Symmetric encryption:**
  - Uses the same key to encrypt and decrypt data
  - Requires the key to be shared with both parties, which is often not possible on the Internet
  - Can be faster than asymmetric
- **Asymmetric encryption:**
  - Uses one key to encrypt the data, and another mathematically related key to decrypt the data.
  - Usually, one key is kept secure and not shared with anyone else (the **private key**) and the other key is shared with everyone else (the **public key**)
  - By encrypting a message with the sender's public key, you can be sure that the only person who will be able to read the message is the sender, as you will need to sender's private key to decrypt the message
  - By encrypting a message with your own private key, anyone who receives your message will have to use your public key to decrypt the message and can thus be sure that the message came from you and has not been tampered with.
- **Communication over the Internet:**
  - When communicating over the Internet, users can use asymmetric encryption to establish a secure connection.
  - Once a secure connection has been established it is possible to securely exchange a key for symmetric encryption
  - Symmetric encryption can then be used for the rest of the session
- **Early methods and weaknesses:**
  - **Substitution cipher** swaps out characters. Assuming 26 alphabet characters, it is easily broken using character frequency.
  - **Vigenère cipher** uses a repeated key combining plain text with the key. Easily broken if we know the length of the key and use the character frequency method similar to the substitution cipher.
  - **Mechanical encryption** such as the World War II (WW2) Enigma machine. Each mechanical method had its own weakness. The Enigma's weakness was it never encrypted a letter as itself.
  - **Data Encryption Standard (DES)** was the first digital encryption standard used a key size of 56 bits. That is small compared to today's standards and is quickly cracked with fast processing speeds available today.
  - **Advanced Encryption Standard (AES)** replaced DES as the commonly used method of encryption. It uses 128, 192 and 256 bits and is yet to be cracked.
  - AES uses symmetric encryption so is often used in conjunction with an asymmetric encryption method (such as RSA) which is comparatively slow.
- **Current best practice**
  - Secure your private key – a stolen key means your data is no longer secure. Ensure only those who need the key are able to access it.
  - Back up your key – a lost key means lost data as it will be permanently encrypted.
  - Use longer length keys to ensure brute force cracking is harder.
  - Use audit logs to check if keys have been accessed by unauthorised users.
  - Best practice is that users should encrypt any messages, critical or sensitive files they send. This extends to the encryption of storage devices in case they fall into the wrong hands.
  - Best practice is based upon the guidelines from NIST: (National Institute of Standards and Technology) <https://csrc.nist.gov/Projects/cryptographic-standards-and-guidelines>.



## Exercises

### *Question 1*

Discuss an advantage and a disadvantage of symmetric encryption compared to asymmetric encryption.

Advantage: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Disadvantage: \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

# Data Management

## Core Concepts

- organisation of a relational database:
    - entities
    - attributes
    - relationships
      - one-to-one
      - one-to-many
      - many-to-many
    - tables as the implementation of entities, consisting of fields and records
    - data types
      - integer
      - float
      - Boolean
      - text
      - date
  - primary and foreign keys to link tables
  - composite key
  - role of ACID in database transactions – atomicity, consistency, isolation, durability
- 
- **Primary key:** A primary key is a unique identifier for each record in a database table. It ensures that each record can be uniquely identified and distinguished from all other records in the table. A PK must be unique and not null
  - **Foreign key:** A foreign key is a field in a database table that establishes a link between data in two tables by referencing the primary key of the related table.
  - **Composite key:** A composite key is a unique identifier that is made up of more than one field. Sometimes there is no single field in a table that will uniquely identify each record, so we need to use a combination of two or more fields. This is a composite key.
  - **Entity:** A distinct object or thing in a database, such as a person, place, or event, that can be identified and stored as a record.
  - **Attribute:** A property or characteristic of an entity, such as a name, age, or address, that provides more details about the entity
  - **Relationship:** A connection or association between two or more entities, defining how they interact with each other within the database.

## ACID

ACID transactions are a set of properties that ensure reliable and consistent database operations. Let's break down what ACID stands for:

1. **Atomicity:** This property ensures that a transaction is treated as a single, indivisible unit of work. Either all changes within the transaction are committed, or none of them are. If any part of the transaction fails, the entire transaction is rolled back to its initial state.
2. **Consistency:** ACID transactions maintain the integrity of the database by ensuring that it transitions from one consistent state to another. In other words, the database remains valid and adheres to predefined rules (such as constraints) after each transaction.
3. **Isolation:** Transactions are isolated from each other, meaning that concurrent transactions do not interfere with each other. Isolation prevents issues like dirty reads, non-repeatable reads, and phantom reads. Different isolation levels (e.g., Read Uncommitted, Read Committed, Repeatable Read, Serializable) control the degree of isolation.
4. **Durability:** Once a transaction is committed, its changes are permanent and survive system failures (e.g., power outages, crashes). The database ensures that the committed data is stored safely and can be recovered even after a failure.

ACID properties are crucial for maintaining data integrity and reliability in database systems.

---

## Data Integrity

---

- Data integrity refers to the validity and accuracy of the data that is stored in a database.
- Factors that may influence the quality of data include:
  - Currency: how up to date the data is
  - Authenticity: how trustworthy the source of the data is
  - Relevance: how related the data is to the purpose or context of use
  - Accuracy: how correct or precise the data is
  - Outliers (cleaning): how consistent or representative the data is
- There is a difference between validity and accuracy. Validity checks to see that the data entered meets the business rules (e.g., a birthdate cannot be in the future). Accuracy means that the data is correct (e.g., the correct birthdate has been entered).
- A DBMS can only check the validity of the data – it has no way of being able to check the accuracy of data that has been entered.
- **Referential integrity:** Referential integrity refers to the relationships between tables in a database. This ensures that each foreign key refers to a valid primary key in the related table (or is null if there is no relationship between the records)
- **Domain integrity:** Domain integrity specifies the set of values that are valid for a column and determines whether null values are allowed. Domain integrity can be enforced by validity checking and by restricting the data type, format and/or range of possible values in a column.
- **Entity integrity:** Entity integrity refers to the records within a table. This ensures that every table has a primary key that is unique to each record and is not null.

## Exercise

Table: Cabin				
cabin_id	name	beds	bathrooms	pets
1	Eagle	4	1	Yes
2	Wren	4	1	No
3	Parrot	8	2	No

Table: Customer			
customer_id	first_name	last_name	email
1001	Jenny	Lane	<a href="mailto:jenny88@hmail.com">jenny88@hmail.com</a>
1027	Max	Peterson	<a href="mailto:peterston@yipee.com">peterston@yipee.com</a>
1384	Allan	Fowler	<a href="mailto:afowler@hmail.com.au">afowler@hmail.com.au</a>

Table: Booking				
booking_id	date_in	date_out	customer_id	cabin_id
56852	11/10/2020	14/10/2020	1001	1
57823	15/12/2020	26/12/2020	1384	3
69825	16/12/2020	20/12/2020	1001	1

Using examples from the tables above, describe the following data integrity terms.

(a) Referential integrity:

---



---



---



---

(b) Domain integrity:

---



---



---



---

(c) Entity Integrity:

---



---



---



---

---

## Normalisation

---

- Normalisation is the process of identifying and eliminating data anomalies and redundancies, thereby improving data integrity and efficiency for storage in a relational database. This process is designed to remove repeated data and improve database design.

### Data Anomalies

- Consider the data in the table below. This unnormalised data can cause problems when data is updated, added or deleted.

Licence	FirstName	LastName	Email	Registration	Make	ManufacturerWeb
19289385	John	Smith	<a href="mailto:jsmith@combi.net">jsmith@combi.net</a>	1COB 293	Ford	<a href="http://www.ford.com.au">www.ford.com.au</a>
19289385	John	Smith	<a href="mailto:jsmith@combi.net">jsmith@combi.net</a>	1QAZ 889	Toyota	<a href="http://www.toyota.com.au">www.toyota.com.au</a>
19289385	John	Smith	<a href="mailto:jsmith@combi.net">jsmith@combi.net</a>	1CCT 441	Mazda	<a href="http://www.mazda.com.au">www.mazda.com.au</a>
26453791	May	Hogarth	<a href="mailto:mhogarth@combi.net">mhogarth@combi.net</a>	1COB 293	Ford	<a href="http://www.ford.com.au">www.ford.com.au</a>
28852462	Peter	Jones	<a href="mailto:pjones@combi.net">pjones@combi.net</a>	1WRC 634	Toyota	<a href="http://www.toyota.com.au">www.toyota.com.au</a>

### Update anomaly

- An update anomaly occurs when you try to update data that is stored in multiple locations.
- If all records are not updated, then data becomes inconsistent and/or inaccurate
- For example, if John Smith updates his email address then **all 3** occurrences need to be updated

### Delete anomaly

- Occurs when by deleting one piece of data you delete the only instance of another piece of data
- For example, if the Peter Jones is removed from the database, the details of the car 1WRC 634 will be lost

### Insert anomaly

- Occurs when data cannot be added because only part of the data is available
- For example, if a new car is added, but no driver allocated then we would be unable to add the car as we need all the information to create a new record

## Normalisation to 3<sup>rd</sup> Normal Form

- Steps to normalisation of data:
  1. Ensure data is in the form of a relation
  2. Convert data to 1<sup>st</sup> Normal Form
  3. Convert data to 2<sup>nd</sup> Normal Form
  4. Convert data to 3<sup>rd</sup> Normal Form

### *Convert data to a Relation*

- In order for data to be in the form of a relation it must:
  1. Have no repeated attributes
  2. All cells must be atomic (that is, they must only contain a single piece of data)

### **Repeated Fields**

Licence	FirstName	LastName	Registration 1	Registration 2	Registration 3
19289385	John	Smith	1COB 293	1QAZ 889	1CCT 441
26453791	May	Hogarth	1COB 293		

The above table is **not** in the form of a relation as it has repeating fields. The Registration field is repeated multiple times.

### **Non-atomic Field**

Licence	FirstName	LastName	Registration
19289385	John	Smith	1COB 293, 1QAZ 889, 1CCT 441
26453791	May	Hogarth	1COB 293

The above table is **not** in the form of a relation as one of the fields is not atomic. The Registration field for John Smith has information about three different cars.

### **Relation**

Licence	FirstName	LastName	Email	Registration
19289385	John	Smith	<a href="mailto:jsmith@combi.net">jsmith@combi.net</a>	1COB 293
19289385	John	Smith	<a href="mailto:jsmith@combi.net">jsmith@combi.net</a>	1QAZ 889
19289385	John	Smith	<a href="mailto:jsmith@combi.net">jsmith@combi.net</a>	1CCT 441
26453791	May	Hogarth	<a href="mailto:mhogarth@combi.net">mhogarth@combi.net</a>	1COB 293

The above table **is** in the form of a relation as all fields are atomic and there are no repeating fields.

**NOTE:** This data is not normalised and would not make a good database structure, but we can now start the process of normalisation.

### 1<sup>st</sup> Normal Form

- To be in 1<sup>st</sup> Normal Form, we must:
  - ensure that all fields are atomic
  - remove all repeating attributes
- Each relation that is formed will have a primary key
- The relation formed from the non-repeating attributes will have a foreign key to the relation formed from the repeating attributes
- The primary key for the relation for the non-repeating fields will now be a composite key comprising the primary key from the non-repeating relation and the repeating relation

### 2<sup>nd</sup> Normal Form

- To be in 2<sup>nd</sup> Normal Form, we must:
  - Be in 1NF
  - Have no partial dependencies
- Partial dependencies occur when a non-key attribute is only dependent on part of the composite key, rather than on the entire composite key
- If a relation does not have a composite key (that is the primary key is made up of a single attribute) then it cannot have any partial dependencies, so is already in 2NF.

### 3<sup>rd</sup> Normal Form

- To be in 3<sup>rd</sup> Normal Form, we must:
  - Be in 2NF
  - Have no transitive dependencies
- All non-key fields in a relation must be fully functionally dependent on the primary key
- Transitive dependencies occur when a non-key field is dependent on a field other than the primary key

## Normalisation Example

### Relation

- Consider the following data. Is it in the form of a relation?

Student Num	First Name	Last Name	Course	Course Name	Result	Result Description
10010504	David	Brown	MATH1001	Mathematics 1A	A	Highly Skilled
10010504	David	Brown	MATH1002	Mathematics 1B	B	Skilled
10010504	David	Brown	COMP1001	Computing 1A	A	Highly Skilled
10020423	James	Stanton	MATH1001	Mathematics 1A	C	Competent
10020423	James	Stanton	COMP1001	Computing 1A	C	Competent
23521461	Debbie	Tainton	MATH1001	Mathematics 1A	B	Skilled
23521461	Debbie	Tainton	MATH1002	Mathematics 1B	A	Excellent
23521461	Debbie	Tainton	COMP1001	Computing 1A	A	Excellent
24352494	Alison	Brown	MATH1002	Mathematics 1B	C	Competent
24352494	Alison	Brown	COMP1001	Computing 1A	A	Excellent

- We can write this using relational notation as:  
**StudentResults**(StudentNum, FirstName, LastName, Course, CourseName, Results, ResultDescription)

### Convert to 1NF

- Firstly, we need to check that all attributes are atomic.
- Then remove all repeating attributes and place them in another relation.

Student Num	First Name	Last Name
10010504	David	Brown
10020423	James	Stanton
23521461	Debbie	Tainton
24352494	Alison	Brown

StudentNum	Course	Course Name	Result	Result Description
10010504	MATH1001	Mathematics 1A	A	Highly Skilled
10010504	MATH1002	Mathematics 1B	B	Skilled
10010504	COMP1001	Computing 1A	A	Highly Skilled
10020423	MATH1001	Mathematics 1A	C	Competent
10020423	COMP1001	Computing 1A	C	Competent
23521461	MATH1001	Mathematics 1A	B	Skilled
23521461	MATH1002	Mathematics 1B	A	Excellent
23521461	COMP1001	Computing 1A	A	Excellent
24352494	MATH1002	Mathematics 1B	C	Competent
24352494	COMP1001	Computing 1A	A	Excellent

- We can also write this using relational notation:  
**Student**(StudentNum, FirstName, LastName)  
**StudentCourse**(StudentNum FK, Course FK, CourseName, Result, ResultDescription)

### Convert to 2NF

- Check for and remove any partial dependencies.
- Partial dependencies will only occur in a relation that has a composite key, so Student is already in 2NF.

Student Num	First Name	Last Name
10010504	David	Brown
10020423	James	Stanton
23521461	Debbie	Tainton
24352494	Alison	Brown

Course	CourseName
MATH1001	Mathematics 1A
MATH1002	Mathematics 1B
COMP1001	Computing 1A

StudentNum	Course	Result	Result Description
10010504	MATH1001	A	Highly Skilled
10010504	MATH1002	B	Skilled
10010504	COMP1001	A	Highly Skilled
10020423	MATH1001	C	Competent
10020423	COMP1001	C	Competent
23521461	MATH1001	B	Skilled
23521461	MATH1002	A	Highly Skilled
23521461	COMP1001	A	Highly Skilled
24352494	MATH1002	C	Competent
24352494	COMP1001	A	Highly Skilled

- We can also write this using relational notation:  
**Student**(StudentNum, FirstName, LastName)  
**Course**(Course, CourseName)  
**StudentCourse**(StudentNum FK, Course FK, Result, ResultDescription)



*Convert to 3NF*

- Finally, we need to check there are no transitive dependencies.
- In this case, the result description is dependent on the result, not the course.

Student Num	First Name	Last Name
10010504	David	Brown
10020423	James	Stanton
23521461	Debbie	Tainton
24352494	Alison	Brown

Course	CourseName
MATH1001	Mathematics 1A
MATH1002	Mathematics 1B
COMP1001	Computing 1A

StudentNum	Course	Result
10010504	MATH1001	A
10010504	MATH1002	B
10010504	COMP1001	A
10020423	MATH1001	C
10020423	COMP1001	C
23521461	MATH1001	B
23521461	MATH1002	A
23521461	COMP1001	A
24352494	MATH1002	C
24352494	COMP1001	A

Result	Result Description
A	Highly Skilled
B	Skilled
C	Competent

- We can also write this using relational notation:  
**Student**(StudentNum, FirstName, LastName)  
**Course**(Course, CourseName)  
**StudentCourse**(StudentNum FK, Course FK, Result FK)  
**Result**(Result, ResultDescription)



## Question 2

The following excerpt shows data that is stored by a bank manager about the customer accounts that he oversees.

Customer ID	Customer Name	Account Number	Account Type	Product Name	Current Balance	Interest Rate	Date Opened	Opened By	Branch
10191	David Burgess	4485 6737 9486 1104	Credit card	Blue VISA	\$50,470.96	21.40%	Dec 23, 2016	Finn Compton	Broome
10191	David Burgess	51931146	Home Loan	Investment One	\$25,611.30	5.24%	Dec 20, 2016	Margaret Wilkins	Broome
10191	David Burgess	79573746	Savings	Everyday Savings	\$88,647.49	0.01%	Apr 20, 2018	Margaret Wilkins	Broome
10191	David Burgess	41425482	Home Loan	First Homebuyer	\$26,173.15	4.03%	Aug 26, 2017	Margaret Wilkins	Broome
10228	Alice Green	89818296	Savings	Saver Plus	\$85,739.02	0.50%	Dec 16, 2017	Adele Perry	Dunsborough
10228	Alice Green	54456999	Savings	Everyday Savings	\$25,060.73	0.01%	Jan 26, 2018	Adele Perry	Dunsborough
10228	Alice Green	73663187	Home Loan	Investment One	\$47,096.15	5.24%	Dec 16, 2016	Adele Perry	Dunsborough
10393	Stephanie Cunningham	5211 7675 1802 1926	Credit card	Platinum MasterCard	\$69,794.45	21.60%	Jul 8, 2017	Hadley Nguyen	Karratha
10393	Stephanie Cunningham	66678786	Savings	Saver Plus	\$87,059.64	0.50%	May 28, 2017	Hadley Nguyen	Karratha
10393	Stephanie Cunningham	52787122	Home Loan	Investment One	\$1,288.30	5.24%	May 12, 2018	Margaret Wilkins	Broome
10409	Mercedes Hendricks	61193735	Savings	Saver Plus	\$22,718.79	0.50%	Feb 9, 2017	Finn Compton	Broome
10589	Emily Goodwin	81269568	Home Loan	First Homebuyer	\$3,904.93	4.03%	May 7, 2017	Margaret Wilkins	Bunbury
10591	Wayne Carter	84313126	Home Loan	Investment One	\$88,073.16	5.24%	Dec 14, 2017	Margaret Wilkins	Bunbury
11011	Neville Moran	4929 4031 8905 9823	Credit card	Blue VISA	\$71,916.91	21.40%	Dec 25, 2017	Adele Perry	Dunsborough

(a) Insert anomaly

---

---

---

---

---

---

---

(b) Delete anomaly

---

---

---

---

---

---

---

(c) Update anomaly

---

---

---

---

---

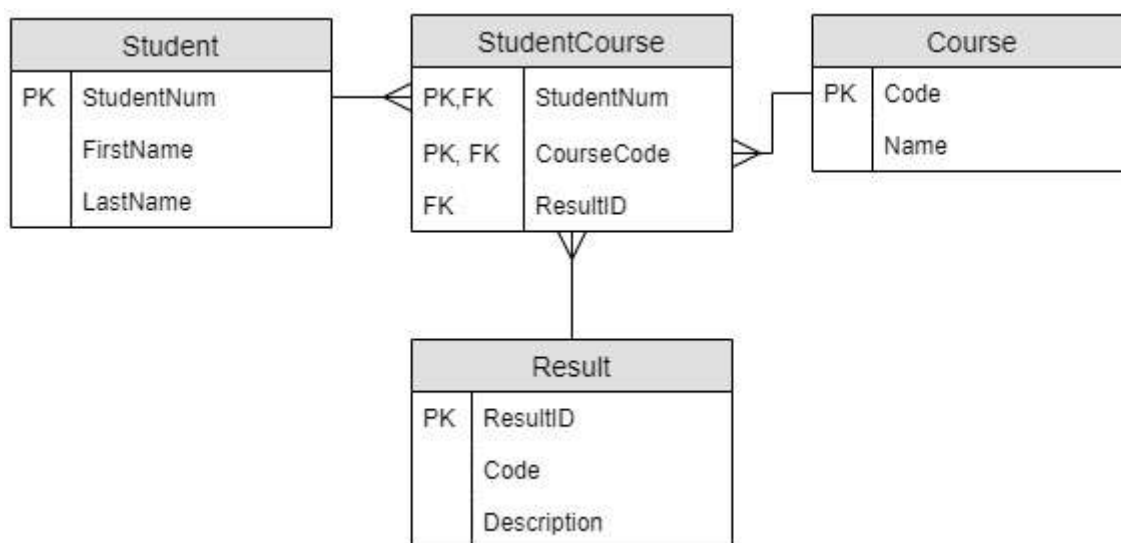
(d) Normalise the data to 3<sup>rd</sup> Normal Form (3NF)

This image shows a single sheet of white paper with horizontal blue ruling lines. The lines are evenly spaced and run across the width of the page. There are no margins, text, or other markings on the paper.

## Entity Relationship Diagrams

- An Entity Relationship Diagram (ERD) provides a graphical representation of the interrelationships between entities in a database.
- Provides an easy, graphical way to visualise how a database is structured and how each entity is related to others.
- There are many variations on how to draw ERDs, but the WACE course uses crow's foot notation. **You must use this in your exam, or you will lose marks.**
- Based on the syllabus, you can be expected to create ERDs of up to 8 tables and analyse ERDs of up to 10 tables.

### Example



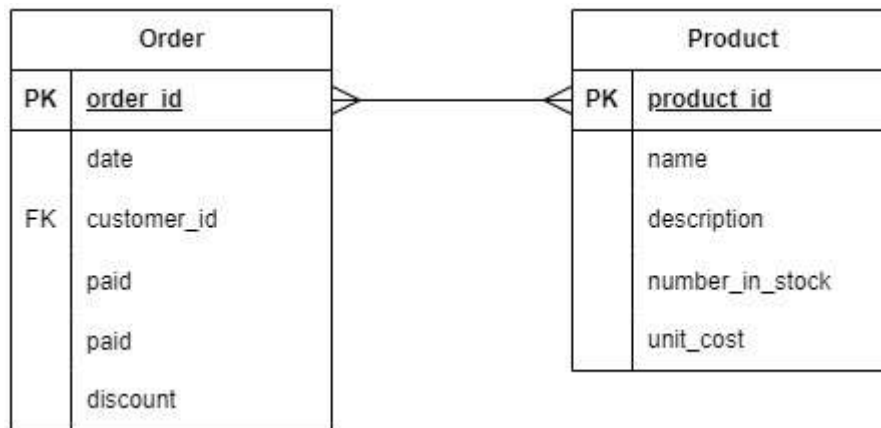
### Creating an ERD

To create an ERD:

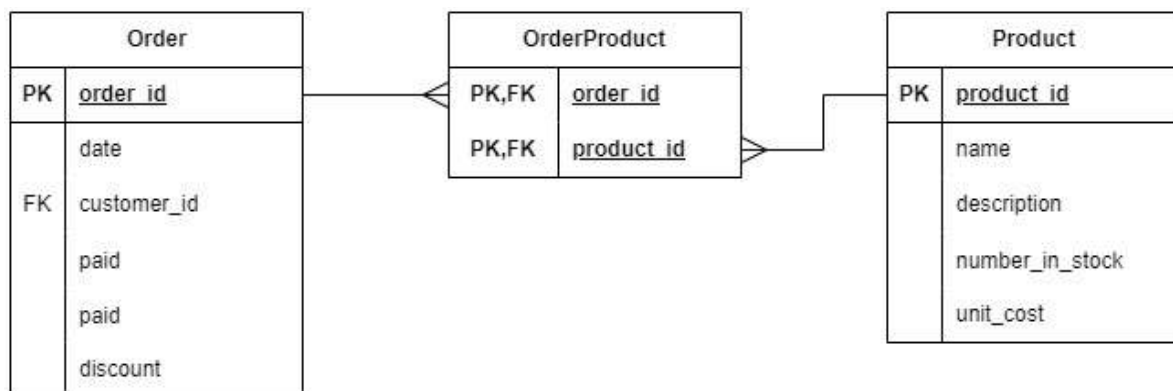
1. Identify the entities that need to have information stored about them
2. Identify the relationships between the entities
3. Identify the cardinality of the relationships
4. Create a primary key for each entity
5. Create any necessary foreign keys
6. Determine any other non-key attributes for each entity

### Many-to-many relationships

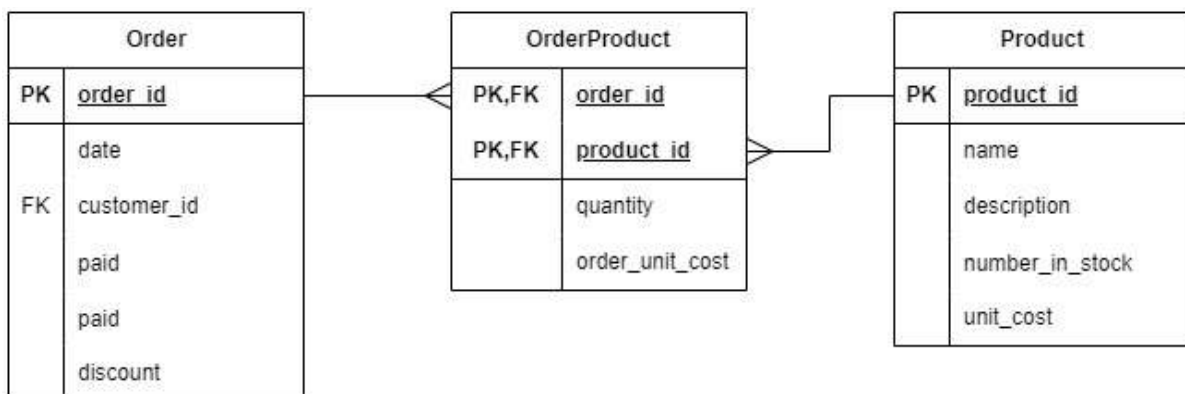
- When creating a relational database, it is impossible to directly implement a many-to-many relationship. Instead, we need to *resolve* any many-to-many relationships we have in our ERD.
- To resolve the many-to-many relationship, we need to place a linking entity between the two existing entities that will allow us to create two one-to-many (1:M) relationships.
- Consider the following M:N relationship showing an order for a number of products.



Each order can be for many products, and each product can be ordered many times. To convert this relationship into a form that can be implemented in a relational database, we need to insert an entity to join the two existing entities. This joining (or bridging) entity will have a foreign key to both the Order entity and the Product entity. We could then either use both these foreign keys to create a composite key or add another unique identifier to use as a primary key. The resulting relationships are shown in the ER diagram below.



Once we have our bridging entity, we may want to store more information, or have a more meaningful name that we could use. For example, each product in an order may have a quantity of that item being ordered, and may also have a unit cost per item for that specific order. We can store these details in the OrderProduct entity.

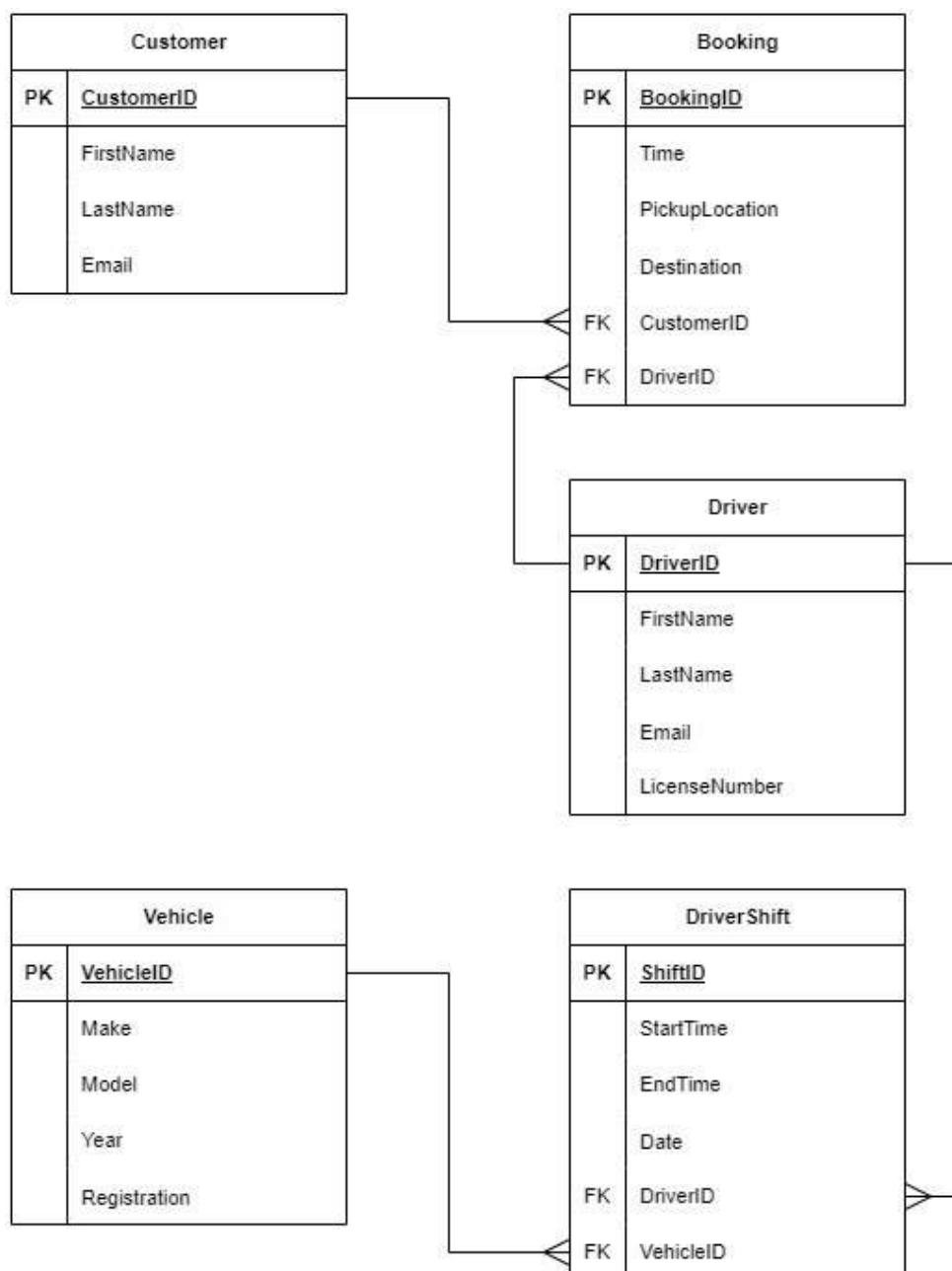


## Sample ERD

Jake is starting a new taxi service and wants to create a database to store information about all his customers, their bookings and his drivers. He has provided you with the following information:

- Customers will need to register with an app on their phone, entering their first name, last name and email address.
- Jake has a fleet of 8 cars that are shared by the drivers
- When a driver starts a shift, they are assigned one of the available cars. This may change each shift, and Jake needs to keep a record of which car each driver was using for each past shift.
- When a customer makes a booking, they enter the booking details (time and location of pickup, the destination and number of passengers). The booking system will then assign them the first driver who is free at that time.

Draw an entity relationship diagram for Jakes Taxi Service. Include all necessary attributes and resolve any many to many relationships.



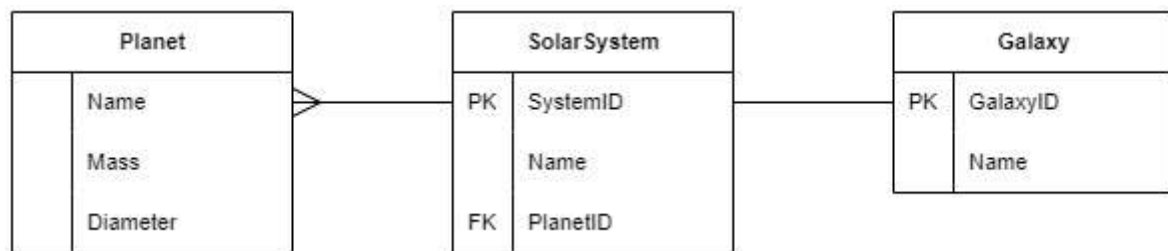
## Exercises

### Question 1

Consider the following relationships.

- A solar system is made up of many planets.
- Many solar systems make up a galaxy.

Identify and describe four errors in the Entity-Relationship Diagram (ERD) below.



i \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

ii \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

iii \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

iv \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_



### Question 2

JSV Banking has a dedicated IT department based at their head office in Bunbury and would like a database to keep track of all the projects that each employee is working on. The IT department is made up of several different sections (for example, Web Development) and each section will have multiple projects running at any one time. For most projects, more than one employee will be needed, and each employee will need to divide their time across several different projects. Each employee will also work in a specific section of IT, however, may be assigned to projects across several different sections as needed.

Draw an Entity Relationship Diagram (ERD) to model the required database. You should include all relationships, primary keys and foreign keys, attributes and resolve any many to many relationships. (15 marks)

### Question 3

Jake has started a Home Maintenance business that provides maintenance and gardening services to people throughout Perth. When a customer requests a job, he looks through his list of sub-contractors and allocates the job to the most suitable person.

When the job is completed, the customer is issued with an invoice (such as the one shown in the image below). Currently he uses an Excel spreadsheet to create each invoice, but as his business has grown, he has found it increasingly time consuming to create each invoice. To solve this problem, Jake has decided to upgrade his invoicing system to use a database to store all the necessary data.

Jake's Home Maintenance and Garden Services		INVOICE	
ABN:5684 599 214		Invoice Num	INV0018656
		Date	15/07/2020
		Customer ID	CUS1985768
Customer:	Peter Jones		
Address:	14 Bell Rd Claremont, WA, 6010		
Phone:	0458 684 286		
Email:	<a href="mailto:pjones@eenet.net.au">pjones@eenet.net.au</a>		
Contractor:	John Smith		
Qty	Description	Unit Price	Line Price
100	Pickets for fence	\$5.75	\$575.00
2	White paint	\$58.00	\$116.00
1	Wood treatment	\$75.00	\$75.00
8	Labour	\$80.00	\$640.00
		Subtotal	\$1,406.00
		GST	\$140.60
		Total	\$1,546.60
Payment must be made within 7 days of the invoice date.			

Using the information on the previous page, draw an Entity-Relationship Diagram (ERD) to show how his new database will be structured. Resolve any many-to-many relationships and include all necessary attributes.

### Question 4

The local computer gaming club wants to start a database of all the games that its members have participated in. They want to keep track of the member's details, the details of each game, and the date, time and score for each game that the member played. They have provided you with this partial Entity Relationship (ER) Diagram. Resolve the diagram, including any primary and foreign keys that may be necessary.



## Using SQL

- SQL (Structured Query Language) is a standard language designed for use with database systems and is designed for storing, manipulating and retrieving data in databases.
- You are expected to be able to use SQL to create, modify and manipulate a database.

Create table	CREATE TABLE name ( pk INTEGER PRIMARY KEY, field1 type NOT NULL, field2 type NULL, ... )
Select all data	SELECT * FROM table
Select specific fields	SELECT field1, field2, field3 FROM table
Select matching rows	SELECT field1, field2 FROM table WHERE expression
Select data from multiple tables	SELECT table1.field1, table2.field1 FROM table1, table2 WHERE table1.pk = table2.fk
Use aggregate functions	SELECT AVG(field1) FROM table
Select unique rows	SELECT DISTINCT field1 FROM table
Sort rows	SELECT field1, field2 FROM table ORDER BY field2 DESC
Group results	SELECT field1, AVG(field2) FROM table GROUP BY field1
Filter grouped results	SELECT field1, AVG(field2) FROM table GROUP BY field1 HAVING expression
Give results a new column name	SELECT field1 as 'Alias name' FROM table
Concatenate fields	SELECT field1    field2, field3 FROM table
Remove table from database	DROP TABLE IF EXISTS table
Insert record into table	INSERT INTO table (field1, field2) VALUES (value1, value2)
Delete all records from table	DELETE FROM table
Delete specific records from table	DELETE FROM table WHERE condition
Change records in a table	UPDATE table SET field1 = value WHERE expression

Comparison operators	= <> or != < > <= >=	Equal to Not equal to Less than Greater than Less than or equal to Greater than or equal to
Logic operators	ALL AND  ANY  BETWEEN  EXISTS  IN  LIKE  NOT  OR	returns TRUE if all expressions are TRUE. returns TRUE if both expressions are TRUE, and FALSE if one of the expressions is FALSE. returns TRUE if any one of a set of comparisons is TRUE. returns TRUE if a value is within a range. returns TRUE if a subquery contains any rows. returns TRUE if a value is in a list of values. returns TRUE if a value matches a pattern (use with the wildcard characters % and _) reverses the value of other operators such as NOT EXISTS, NOT IN, NOT BETWEEN, etc. returns TRUE if either expression is TRUE
Aggregate functions	AVG COUNT MAX MIN SUM	calculate the average value count the number of items in a set find the maximum value find the minimum value calculate the sum of values

---

## Development Issues

---

- When developing databases, in particular databases that may be storing personal data, developers need to be aware of both their legal and ethical obligations.
- Many ethical and legal issues may overlap. Make sure you are addressing each issue based on what the question is asking about. For example, you may need to consider the ethical implications of privacy, even though there are legal issues as well.
- Some ethical issues may include:
  - What data is being collected about individuals and should it be collected
  - Are there any privacy issues about what is being collected and who will have access to it?
  - Are the sources of data reliable
  - Ensuring that data is used appropriately
  - Ensuring that personal information is not used in such a way that may have a negative impact on an individual
- Legal obligations are covered by the Australian Privacy Act 1988, and specifically by the Australian Privacy Principles. (see <https://www.oaic.gov.au/privacy/australian-privacy-principles/> for more information)
- Some legal obligations include:
  - Must store personal data securely
  - Must allow people access to data stored about them
  - Must notify people as to what data is being stored about them
  - Personal information that is no longer needed must be disposed of in a secure, timely and reasonable manner
  - Data must only be used for purposes that the user has agreed to
- In addition to legal and ethical issues, developers also need to consider a variety of security issues. For example:
  - Making sure that all personal data is kept private (this is also a legal requirement)
  - Preventing loss of data by having regular backups of an organisation's data
  - Restricting who has access to data, through things such as authentication and user accounts
  - Ensuring that suitable people have ownership and control of the data, and only those that need access can access it.

## Exercise

### *Question 1*

Peter owns a second-hand car dealership with a yearly turnover of approximately \$3 million dollars. He collects, stores and disposes of car registration and drivers licence data as a normal part of selling cars. The registration and licence data that he collects is saved to a flash drive so he can easily exchange it with an insurance dealer he knows so that his friend can sell insurance to people who have just bought a car.

**(a)** Explain two ways that Peter may be breaking the law.

i \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

ii \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

**(b)** Explain one ethical issue with what Peter is doing.

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

### Question 2

AussieToys is an online toy store that sells Australian made toys across the world. As part of their sales process, they collect information about each of their customers so they can personalise the shopping experience for each customer.

(a) Identify the government act that regulates the use of personal information.

---

---

(b) Describe two aspects of this act that AussieToys needs to consider when collecting personal information about their customers.

i

---

---

---

---

---

ii

---

---

---

---

---



---

# Exam Technique

---

---

## Sitting the Exam

---

### Reading time

- Read scenario in Source Booklet
- Skim over questions in Extended Response section
- Go back to start of paper and skim over short answer questions

### Start paper

- Work through short answer questions
- Keep an eye on the time
- Start with questions on topics you are confident with
- Read question carefully:
  - Take note of key words – identify/state/describe/explain
  - Consider marks allocation
  - Answer question that is being asked
- Extended response questions:
  - Read source booklet carefully
  - Follow process to create ERD diagrams
  - Use correct symbols for diagrams
  - Use standard syntax and key words for pseudocode
  - Use CISCO symbols for network diagrams. It is also a good idea to label your diagram in case you get the CISCO symbols wrong. That way you will still at least get some marks.
- Make sure all answers **use content from the syllabus** as this is the examinable material. For example, if asked to identify some ethical issues when developing a database, use the ones listed in the syllabus rather than making up your own.
- NOTE: Wherever possible use keywords and terminology from the syllabus to answer questions.

---

## Preparation

---

- Identify your areas of weakness
- Memorise key concepts and terminology. Make sure you know the correct terminology to use.
- Learn the different types of question and the expectations for answering each question
- Practice reading and writing pseudocode
- Complete past papers under timed conditions. If you do not have a solid three-hour block, then complete each section in the suggested time frame

Congratulations! You have now completed your revision booklet!

Edith Cowan University would like to wish all students the best of luck with their future exams!

