Python Programming with Sequences of Data

Why Learn Python Programming with Sequences of Data?

Embarking on the journey of programming offers Year 9 students an exciting gateway to unleashing **creativity** and **problem-solving** prowess. In this course, you'll dive into practical skills like crafting programs that communicate with users, crunch numbers effortlessly, and organize data efficiently using lists. Imagine creating interactive games, designing apps, or automating tasks that simplify everyday life—all achievable with the programming fundamentals you'll master.

Understanding the logic behind loops (both for and while) will empower you to manipulate data dynamically, while mastering selection statements (if-elif-else) will grant you control over program behaviour based on conditions you define. Recognizing and fixing common syntax errors ensures your code runs smoothly, **teaching you attention to detail** crucial in programming and beyond.

Moreover, this knowledge equips you with essential problem-solving techniques and lays the groundwork for tackling more complex challenges in the future. Whether you're passionate about technology, curious about how software works, or simply eager to create something new, **programming will empower you to turn your ideas into reality.** 😊

```python
# Task 1: Arithmetic and Selection

# Ask the user to input two numbers
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))

# Calculate the sum and difference of the two numbers
sum_result = num1 + num2
difference_result = num1 - num2

# Print the sum and difference
print(f"Sum is: {sum_result}")
print(f"Difference is: {difference_result}")

# Use selection (if-elif-else) to determine and print whether the
if sum_result > 10:
    print("Sum is greater than 10.")
elif sum_result < 10:
    print("Sum is less than 10.")
else:
    print("Sum is equal to 10.")
```

```mathematica
Enter the first number: 7
Enter the second number: 3
Sum is: 10
Difference is: 4
Sum is equal to 10.
```

""Programming is the closest thing we have to a superpower. It allows you to meld the digital world to your imagination." — Jack Dorsey – Inventor of X

Key Words / Skills	Definition and Example
Display messages	Show words or sentences on screen. Example: Printing "Hello!"
Receive keyboard input	Get typed information into a program. Example: Asking for your name
Simple arithmetic expressions	Basic math operations in a program. Example: Adding two numbers
Selection (if-elif-else)	Decide actions based on conditions. Example: If sunny, play outside
Locate syntax errors	Find and fix mistakes in code. Example: Correcting missing parenthesis
Create lists	Group items together in a program. Example: List of friends' names
Access list items	Retrieve individual items from a list. Example: Picking a friend's name
Perform list operations	Manipulate items in lists. Example: Adding or removing toys
Use while loops	Repeat actions until a condition is met. Example: Rolling a dice
String operations	Modify or find characters in words. Example: Changing 'a' to 'e'

Key Questions

How do you create a list in Python?

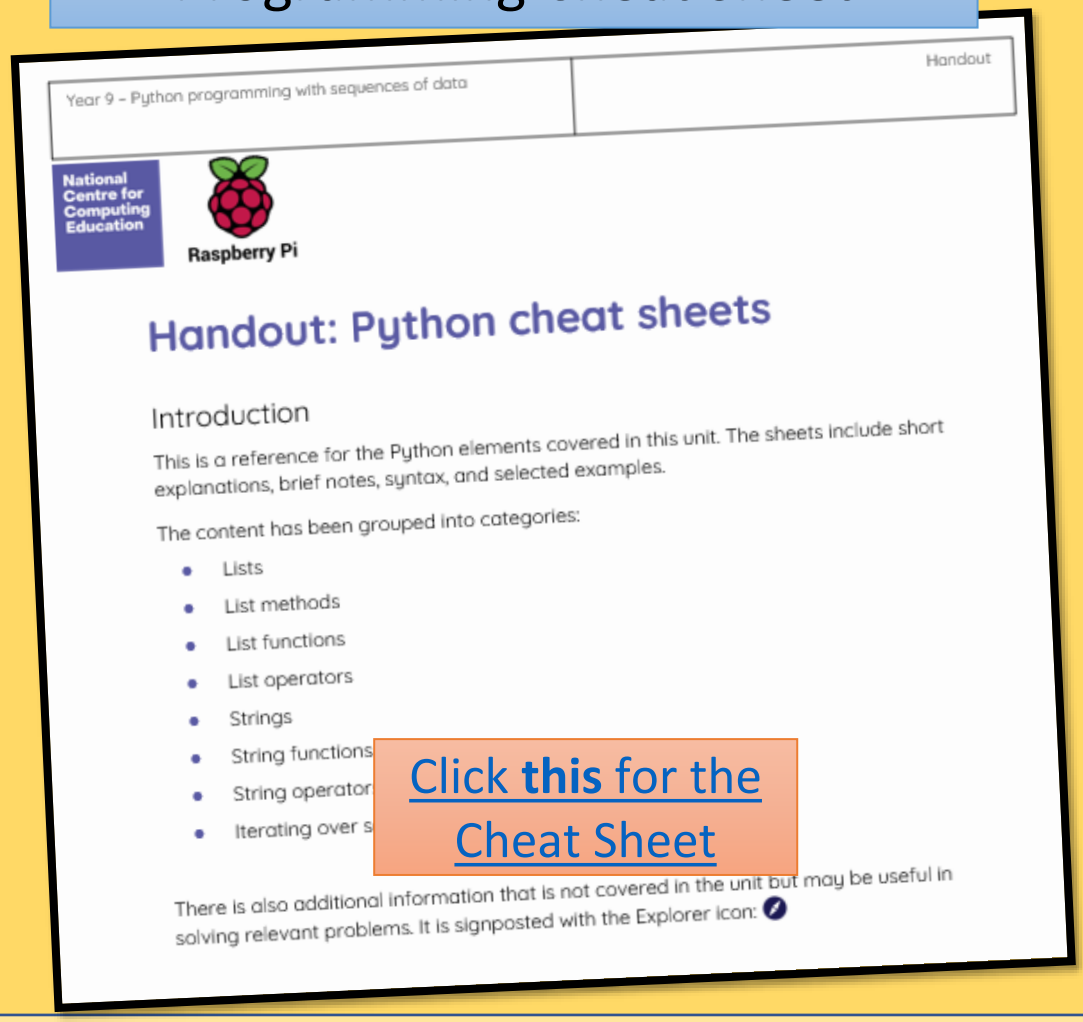
What is the purpose of an if-elif-else statement?

How would you use a for loop to iterate over a list of numbers?

What is a common syntax error you might encounter in Python, and how can you correct it?

Key Knowledge

Programming Cheat Sheet



Year 9 – Python programming with sequences of data

Handout

National Centre for Computing Education

Raspberry Pi

Handout: Python cheat sheets

Introduction

This is a reference for the Python elements covered in this unit. The sheets include short explanations, brief notes, syntax, and selected examples.

The content has been grouped into categories:

- Lists
- List methods
- List functions
- List operators
- Strings
- String functions
- String operator
- Iterating over s

There is also additional information that is not covered in the unit but may be useful in solving relevant problems. It is signposted with the Explorer icon:

Click **this** for the Cheat Sheet

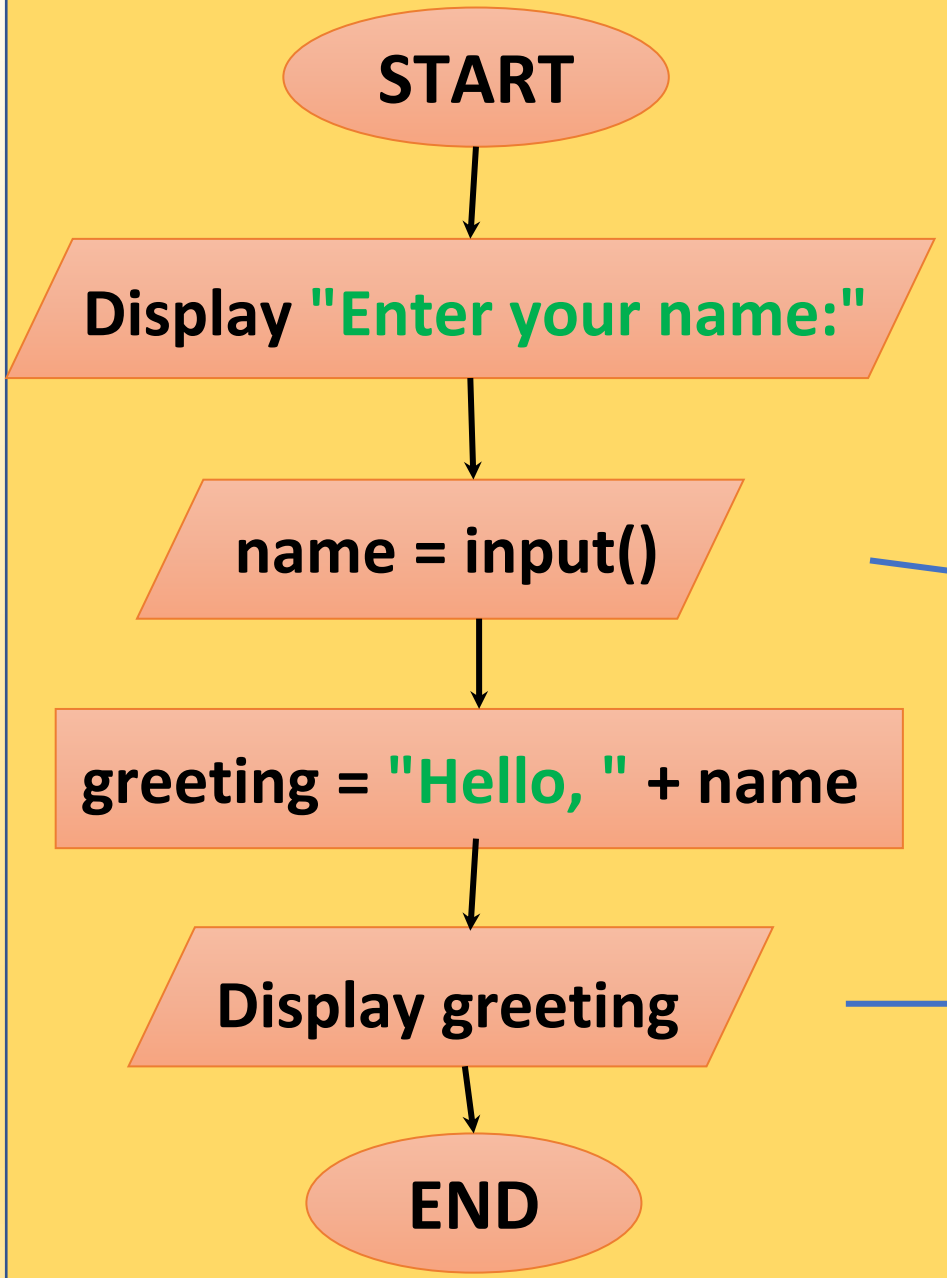
Key Knowledge

"Grasp the concept of data types and their usage in Python."

Intege	5 7 3 0 -4 15843315 -4564
Ffoat	0.3 -11.002 -4.01 1.15843315 11.1
Boolean	TRUE FALSE
String	"Hello World" "H@ppy B1rthd@y"
Character	"f" "F" " " "?" "2"

Key Knowledge

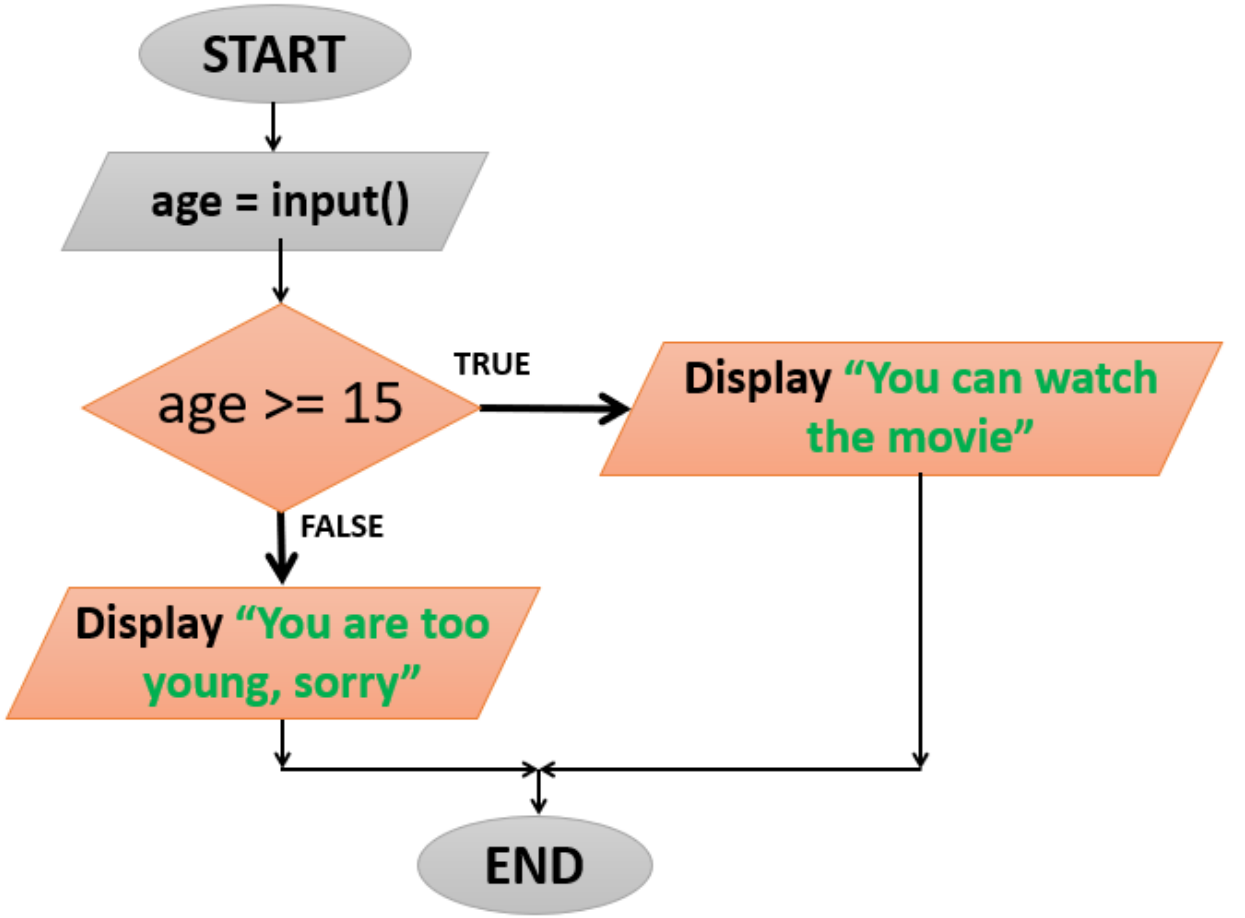
Write simple Python programs that display messages, assign values to variables, and receive keyboard input.




```
print ("Enter your name:")

name = input ()

greeting = "Hello, " + name

print (greeting)
```

Key Knowledge

Use **selection** (if-else statements) to control the flow of program execution.



Make and use **lists.**

Populate a List:
Start by using square brackets `**[ ]**` to indicate a list.
Optionally, you can initialize the list with elements inside the brackets separated by commas.

```
my_list = [1, 2, 3, 4, 5]
```

List with Different Data Types:
Python lists can hold elements of different data types (e.g., integers, floats, strings, etc.) in the same list.

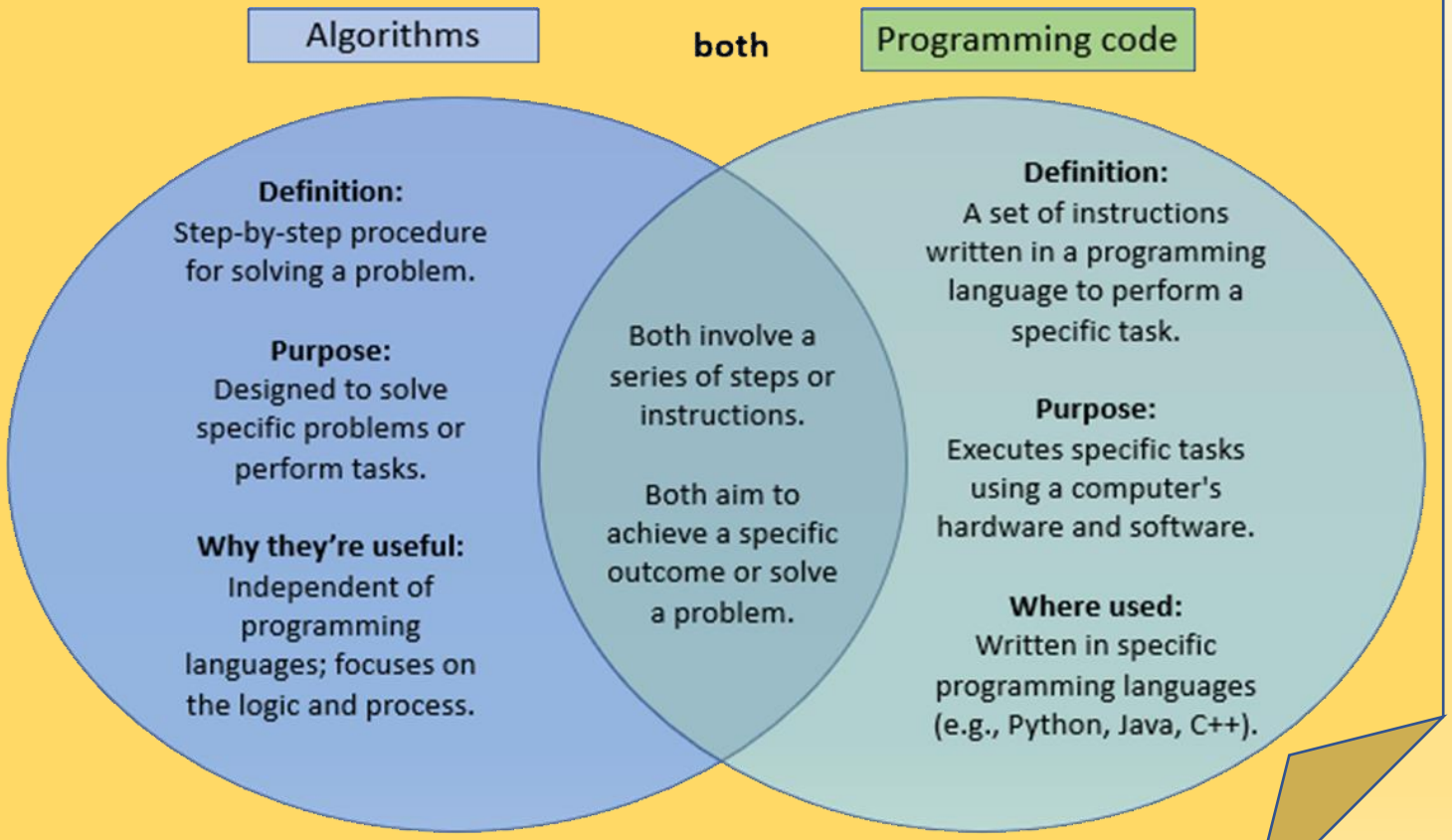
```
mixed_list = [1, 2.5, "apple", True]
```

Make an empty List:
You can also create an empty list by simply using empty square brackets `**[ ]**`.

```
empty_list = []
```

Key Knowledge

"Describe what algorithms and programs are and how they differ."



Locate and correct common syntax errors.

Incorrect Use of Equals Sign

```
# Incorrect
x = 10
if x = 10:
    print("x is 10")

# Correct
x = 10
if x == 10:
    print("x is 10")
```

Missing a Parenthesis (Bracket)

```
# Incorrect
print("Hello, world!"

# Correct
print("Hello, world!")
```

Improper Use of Tabs and Spaces

```
# Incorrect
if True:
    print("Correct")
  print("Well done!")

# Correct
if True:
    print("Correct")
    print("Well done!")
```

Variable Name Starts with a Number

```
# Incorrect
1st_place = "John"

# Correct
first_place = "John"
```

Missing Quotation Marks

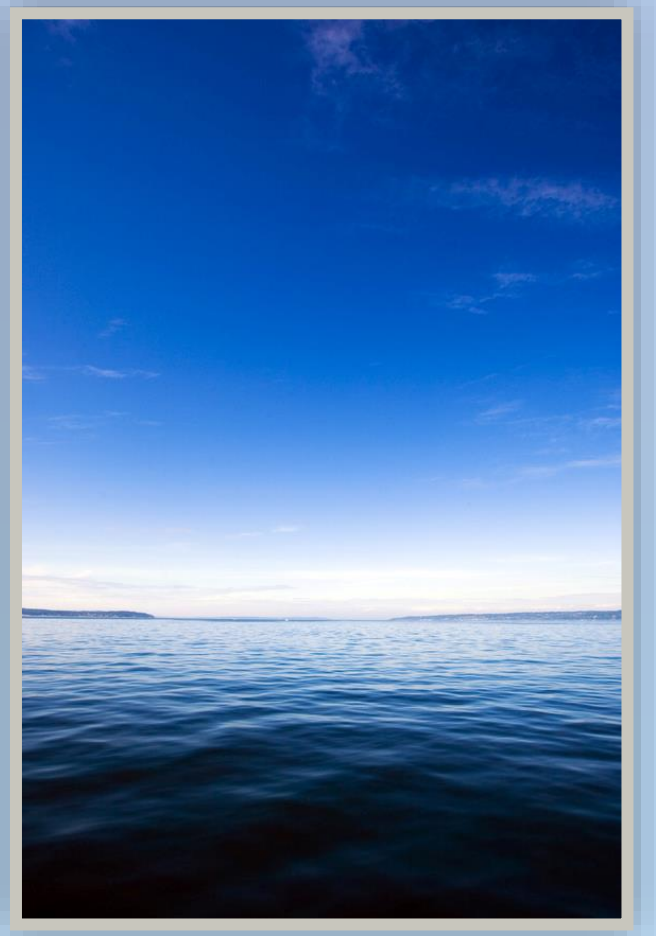
```
# Incorrect
message = Hello, world!

# Correct
message = "Hello, world!"
```

Please screenshot the results of your Programming quiz here

Please screenshot your program and paste below

Please screenshot your program running and paste below

Chillax Project



Why Learn the Chillax Project?

Imagine you're a designer, tasked with creating **the ultimate chill-out space** for your friends. The **Chillax Project** is your chance to step into those shoes and bring a cool, relaxing room to life. You'll start by understanding what your "client" wants—a place to unwind, have fun, and feel comfortable. Next, you'll brainstorm ideas and create a mind map to make sure you've got everything covered.

You'll then dive into designing, starting with **mood boards** and sketches to visualize your ideas. But it doesn't stop there! Using Blender, you'll turn your sketches into a **3D model**, making your room look almost real. Once your design is ready, you'll use Canva to create promotional photos and to put together a stylish leaflet, showing off your creation to the world.

Throughout this project, you'll learn to **think like a designer**, solve problems creatively, and use cool tools to make your ideas come to life. In the end, you'll compile all your work into an impressive **e-portfolio**, showcasing your journey from ideas to final product. The Chillax Project is not just about designing a room; **it's about turning your imagination into reality** and **sharing it with others**.

Examples of work

"Creativity is intelligence having fun!"
– Albert Einstein

Key Words / Skills	Definition and Example
Mood boards	Collections of images, colours, and ideas that help plan and inspire a project.
Visualisation diagrams	Drawings or charts that show how something will look or work before it's made.
3D visualisations	Creating lifelike images or models using special software like Blender, to see how something will appear in three dimensions.
Advertising leaflets	Flyers or papers that promote something, made by combining photos, designs, and information.
Promotional shots	Pictures that show off something in a positive way, used to advertise or promote.
E-portfolio	An online collection of your work, like pictures and projects, to show what you can do.
Target audience	The specific group of people you want to reach with your work or message.
Digital marketing	Ways to promote things online, like using websites and social media to reach people.
HTML	A language used to create web pages, making it possible to show your work online in an e-portfolio.

Key Questions

What is a mood board, and why is it important in pre-production?

How can you use Blender to create a 3D visualisation?

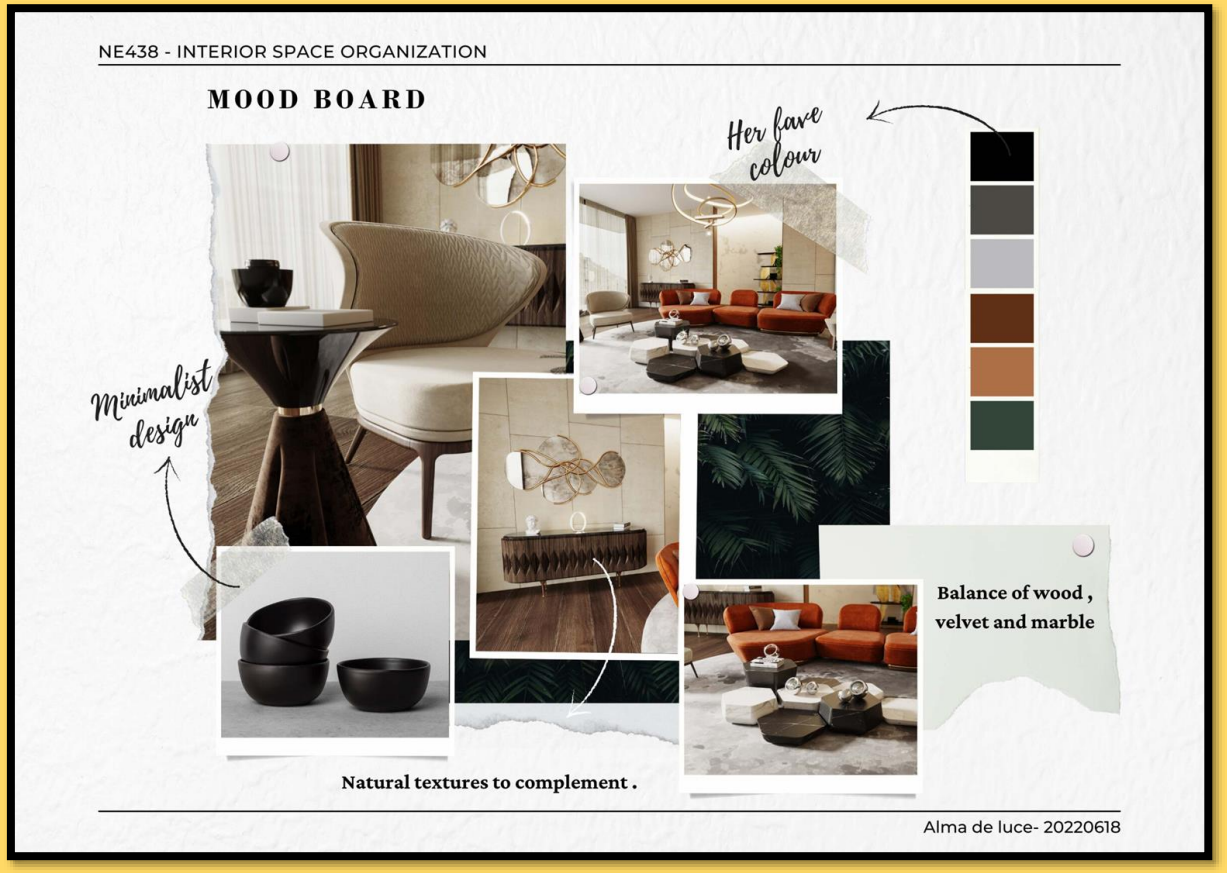
What are some key elements to include in an advertising leaflet?

How can you identify the target audience for your project?

Key Knowledge

"Create pre-production documents including"

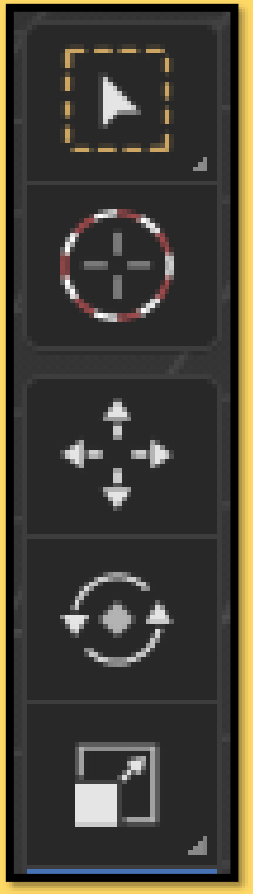
"mood boards"



"visualisation diagrams."

Create 3D visualisations using Blender

Key Knowledge



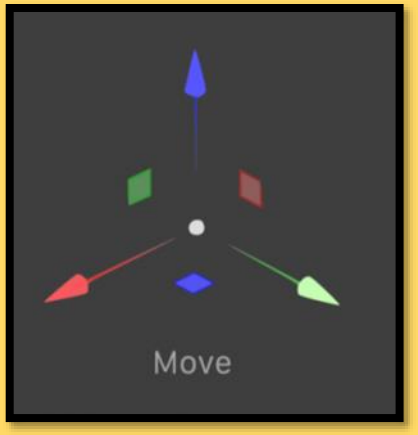
Select Box: used to select objects

Curser: put the curser where you want new objects to go

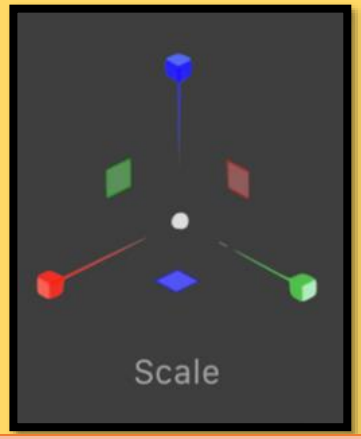
Move: select this to move the selected object

Rotate: select this to rotate the selected object

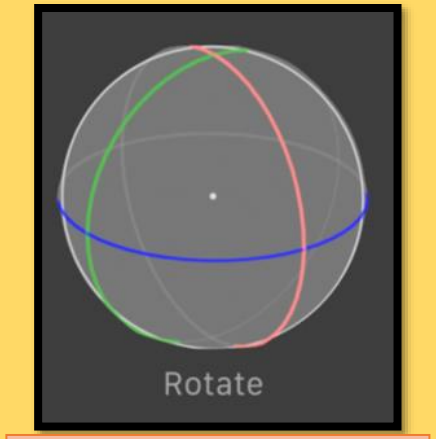
Scale: select this to scale the selected object



Pull these to move the object in the direction of the coloured arrow

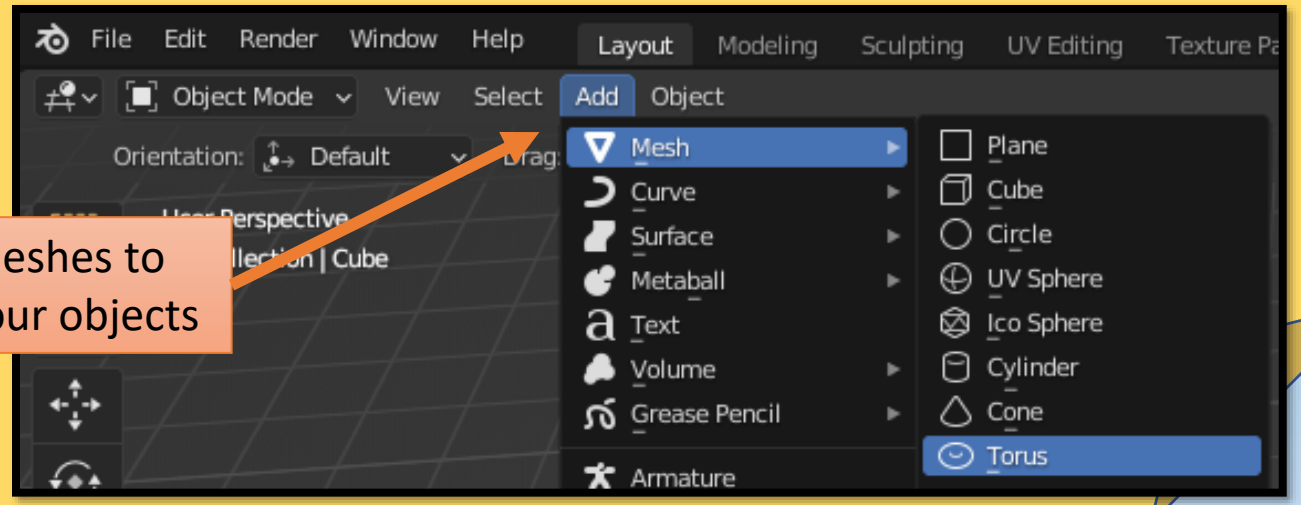


Pull these to scale the object in the direction of the coloured arrow



Pull these to rotate the object in the direction of the coloured line

Add meshes to make your objects

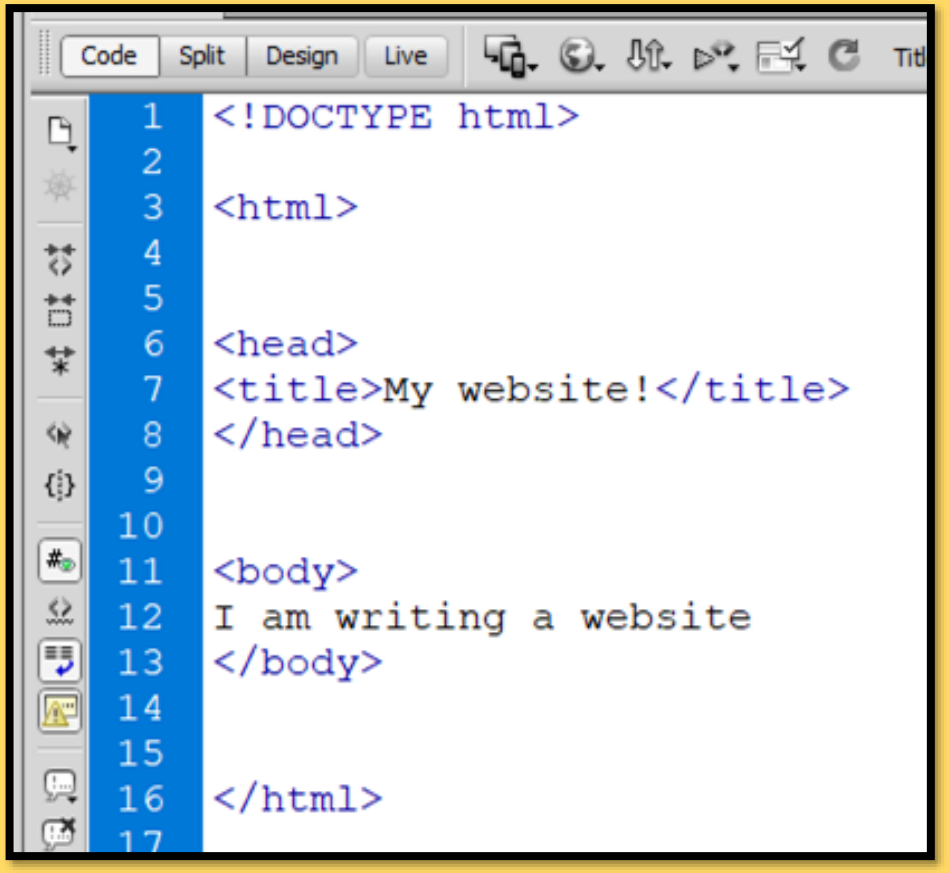


[Click here for LOOM Blender Basics: The Asset Collection, Adding and Removing Meshes and Move, Rotate and Scale]

Key Knowledge

Comprehend the basics of HTML

The HTML 'Skeleton'



<html></html>	→	START AND END DOCUMNET
	→	HEADING TAGS (H1 TO H6)
	→	BOLD TEXT TAG
	→	ITALIC TAG
	→	IMAGE TAG
	→	BASIC LINK TAG
 	→	LINE BREAK TAG

Key Knowledge

Develop advertising leaflets using Canva.

Canva is super easy
Click me for a guide

Identify the target audience and determine what appeals to them.

Your Target Audience

Please screenshot your multiple choice quiz results

Please screenshot your leaflet showing your Chillax Room

Please screenshot your e-portfolio

Please screenshot the e-portfolio's html

Mobile App Development

Year 9 Unit 3 – Mobile App Development

The market for mobile apps is huge. From cynical cash-grabbing 'gacha' games designed to squeeze as much money out of the user as possible to apps that help track fitness, in a world where there's an app for '*every possible need*', this unit aims to take you from **designer** to **project manager** to **developer** in order to create **your own mobile app**.

Using **App Lab** from code.org you will familiarise yourself with the coding environment and have an opportunity to build on the programming concepts used in previous units before undertaking your project.

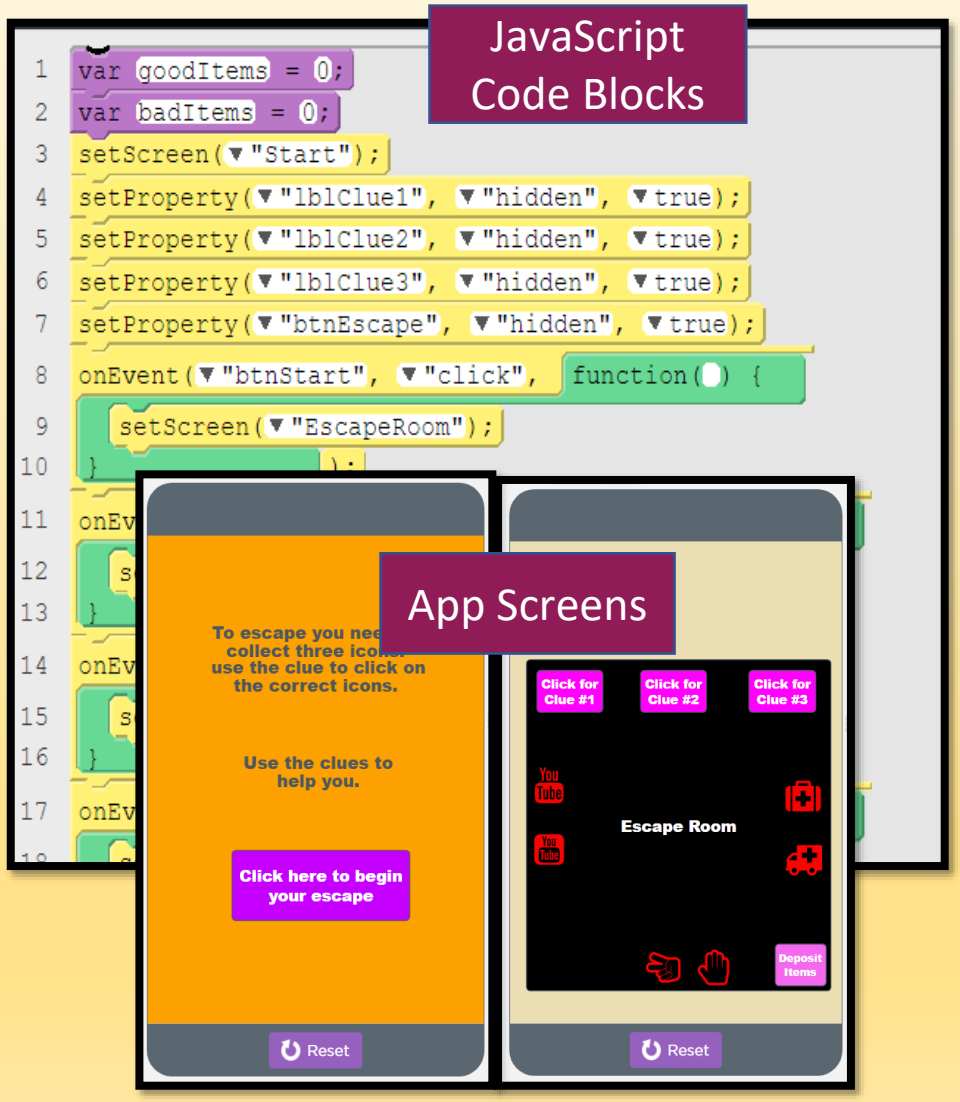
Developing an app is what we call a **complex problem**. When you look at the app as a whole it can feel overwhelming so you will also learn how to decompose the project into smaller, more manageable parts. This decomposition skill is one that the wise amongst you will adopt and use in your day to day living. It was decomposition that triggered the 2nd industrial revolution…

At the end of this unit you will know how to use the **basic programming constructs** in a block-based programming environment. You will develop your **computational thinking** skills, with the goal of creating a prototype escape room like app for your phone

Skills you will be practicing from previous units:

- Y7 Computational Thinking – using decomposition; Y7 Virtual Pet – programming (cause and effects)
- Y8 Music App – programming using selection

An example of your work



"Everyone should learn how to program a computer because it teaches you how to think."
— Steve Jobs - Apple

Year 9 Unit 3 – Mobile App Development

Mobile App Development Keywords	
Decomposition	Breaking down a big problem into smaller, easier-to-handle parts.
GUI (Graphical User Interface)	The part of an app you see and interact with, like buttons and screens.
Event-Driven Programming	A type of coding where the flow of the program is controlled by events like user actions (clicks, typing).
Variables	Containers in programming that hold information that can change, like scores in a game.
Block-Based Programming Language	A way of coding that uses blocks to represent code concepts, making it easier to learn and use, especially for beginners.
Debugging	The process of finding and fixing mistakes in your code.
User Input	Information that the user provides to the app, like typing a name or pressing a button.
Sequencing	The order in which instructions are executed in a program.
Selection	Using conditions in programming to make decisions, like if-else statements to choose different actions based on certain criteria.

Key Questions

How can you use user input in an event-driven programming environment?

What is a common coding error you might encounter in block-based programming?

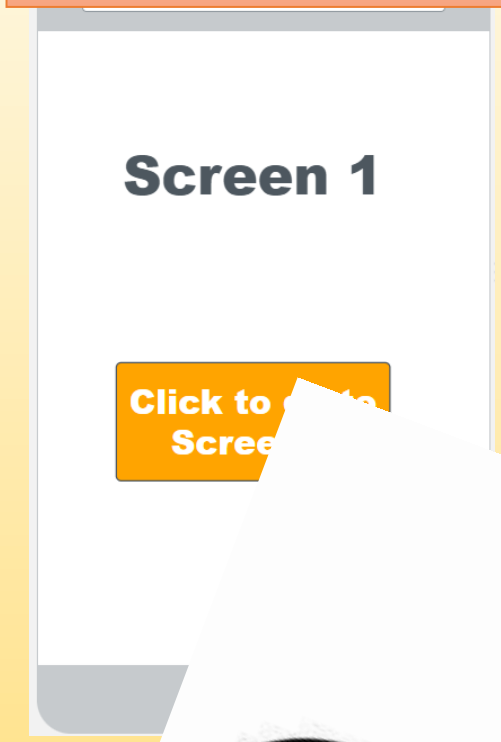
How do you create a sequence in a block-based programming language?

Why is it important to establish user needs when developing a mobile app?

Key Knowledge

Identify the error

Identify and fix common coding errors.

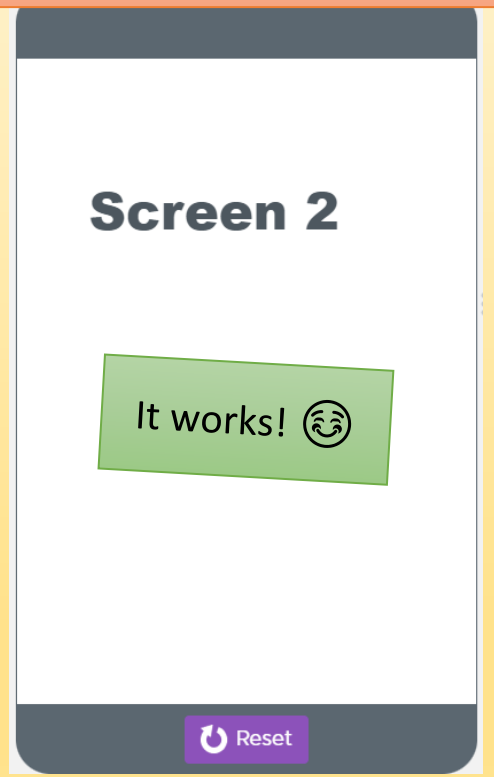


Find and fix the error

```
1  onEvent("btnGo_To_Screen_2", "click",
2    setScreen("screen2");
3  });
```

Now set to screen 2

Test the fix



Key Knowledge

Understand the principles of event-driven programming and its applications in mobile apps.

click this

Video: App Lab - Getting Started

Use block-based programming languages to create sequences including sequencing and selection.

click this

Video: App Lab - Make It Interactive

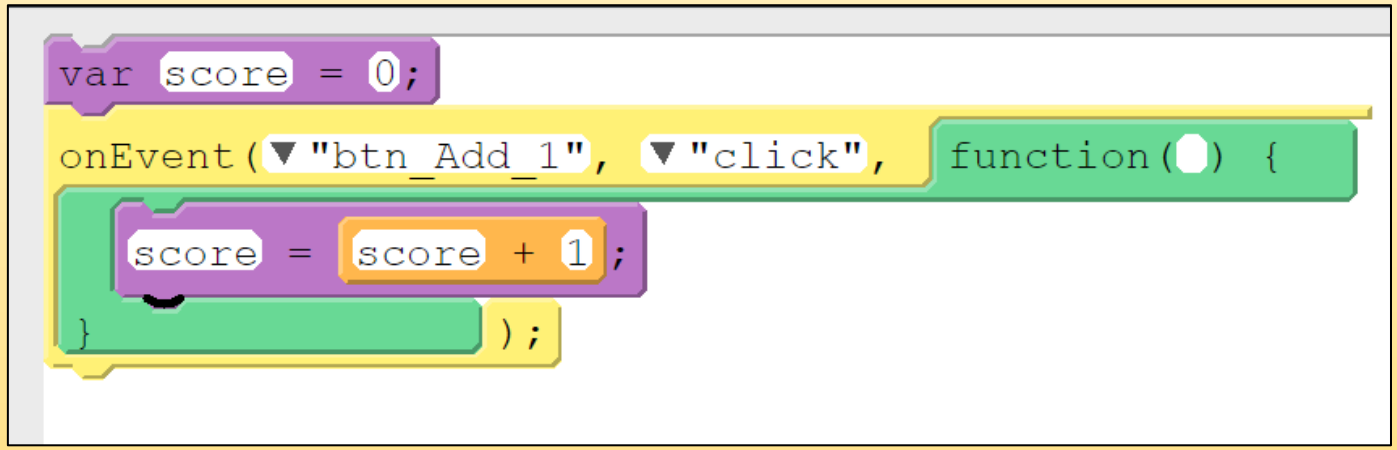
Key Knowledge

Use user input and variables in an event-driven programming environment.

Make a new variable called **score** and set its value to **0**

When the **Add 1** button is clicked

Set the value of the score variable to its current value + 1



Please screenshot your multiple choice quiz results

Please screenshot your app running

Please screenshot a section of your app's code

Representations: Going Audio Visual

Representations: Going Audio-Visual

Ever wondered how your favourite apps and games make the stunning visuals and catchy tunes you enjoy?

Just like in the previous unit, where you examined characters and numbers, the ideas that you need to understand are not really new to you. You will draw on familiar examples of composing images out of individual elements, mixing elementary colours to produce new ones, and taking samples of analogue signals, to illustrate these ideas and bring them together in a coherent narrative.

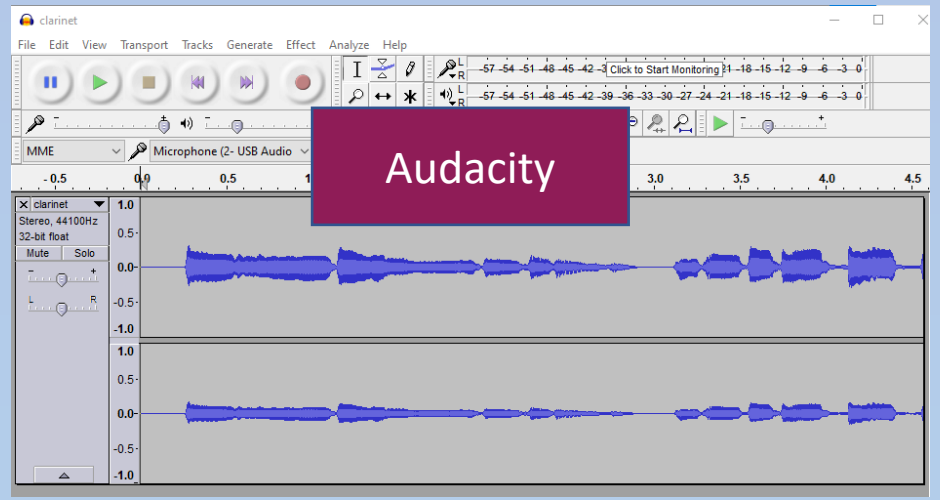
This unit has a significant practical aspect. You will use relevant (and free!) software (GIMP and Audacity, in this case) to manipulate images and sounds and get an idea of how the underlying principles of digital representations are applied in real settings.

At the end of this unit you will know how **digital media** such as **images** and **sounds** are **represented** by computers, and discover the **binary digits** that lie beneath these types of media.

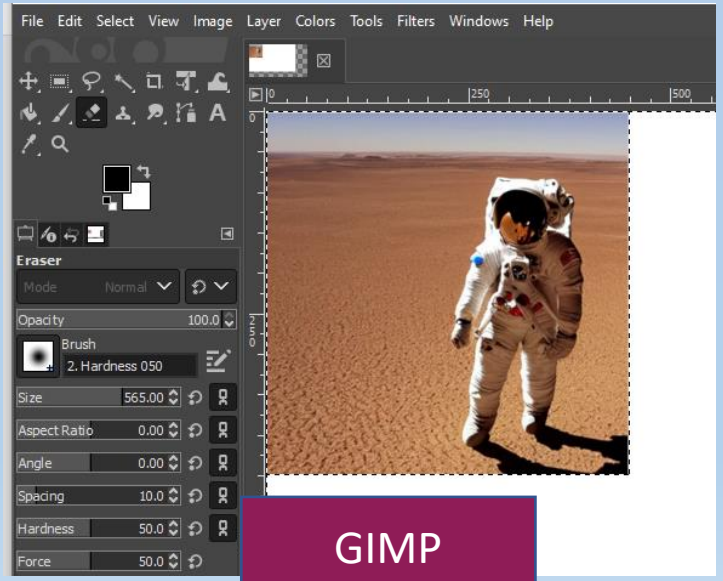
Skills you will be practicing from previous units:

- Y8 Media – Vector Graphics, Representations: From Clay to Silicon

An example of your work



Audacity



GIMP

"In the digital age, understanding data is not just about reading numbers; it's about seeing patterns and relationships through audio-visual representations."

Representations: Going Audio-Visual

Representations: Going Audio-Visual Keywords	
Bitmap	a digital image format where an image is represented by a grid of individual pixels, each with its own colour or shade.
Pixel	the smallest unit of a digital image, representing a single point of colour.
Resolution	the clarity or sharpness of an image, often measured in pixels per inch (PPI) or dots per inch (DPI).
Colour Depth	the number of bits used to represent the colour of each pixel in a digital image, determining the range and accuracy of colours that can be displayed.
Soundwave	a representation of sound as it changes over time, often depicted graphically as a waveform showing variations in amplitude (loudness) over time
Sample	a single measurement of a sound wave taken at a specific point in time.
Sample Rate	the number of samples taken per second during the digitization of analogue sound, typically measured in Hertz (Hz).
Hertz (Hz)	the frequency of samples captured to represent the original sound wave. 1Hz is one sample per second
Sample Size	Sample size refers to the amount of data recorded for each individual sample in digital audio, often measured in bits per sample
Compression	the process of reducing the size of a file or data to save storage space or transmission time, often achieved by removing redundant or unnecessary information.

What is a pixel, and how is its colour represented in binary?

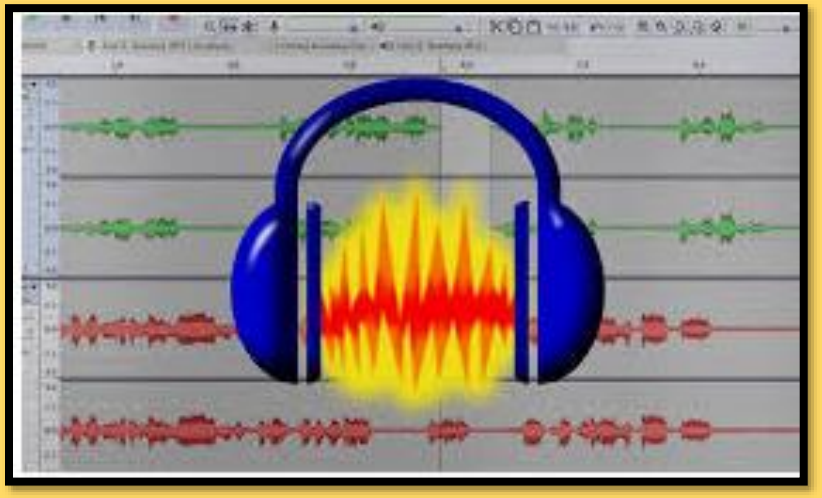
What is the difference between resolution and colour depth?

How do you perform basic image editing tasks using software like Photoshop or GIMP?

What does ‘sampling frequency’ mean in the context of sound representation?

Key Knowledge

"Perform basic sound editing tasks using appropriate software"



Click here for a help video

"Perform basic image editing tasks using appropriate software"

Click here for some help docs

Key Knowledge

Describe how digital images are composed of pixels and how their colour is represented in binary.

Click here for the sheets on the above

😂

Define key terms like '**pixels**', '**resolution**', and '**colour depth**'.

Click here for the sheets on the above

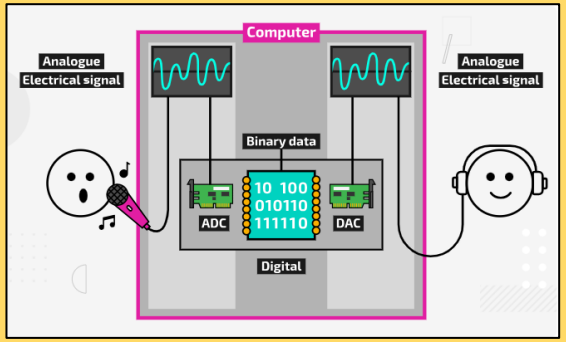
😂

Understand the principles of sound representation including **'sample'**, **'sampling frequency/rate'**, and **'sample size'**.

Click here for the sheets on the above

😂

00	00	00	00	01	01	01	01	00	00	00	00
00	00	01	01	01	01	01	01	01	01	00	00
00	01	01	01	01	01	01	01	10	01	01	00
00	01	01	10	10	01	01	10	01	01	01	00
01	01	01	10	10	01	01	01	10	01	01	01
01	01	01	01	01	01	01	01	01	01	01	01
01	01	01	01	01	01	01	01	01	01	01	01
01	01	10	10	10	10	10	10	10	10	01	01
00	01	01	10	10	11	10	11	10	01	01	00
00	01	01	01	10	11	11	11	01	01	01	00
00	00	01	01	01	01	11	01	01	01	00	00
00	00	00	00	01	01	01	01	00	00	00	00

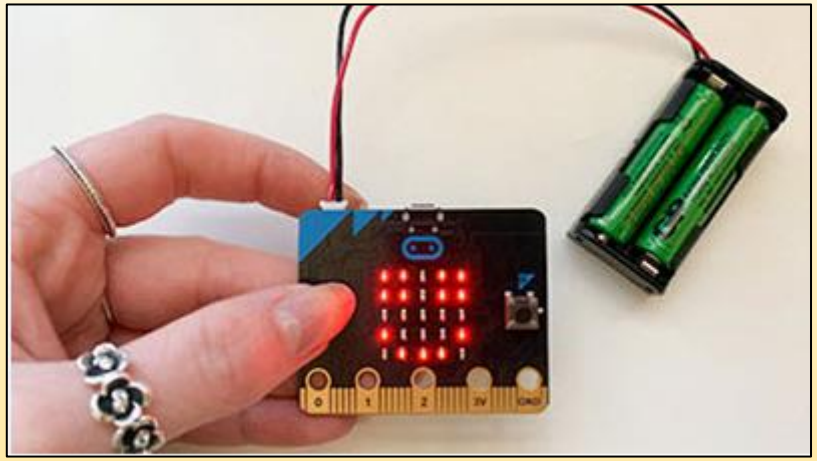


Please screenshot your multiple choice quiz results

Please screenshot your favourite image that you edited

Please screenshot the work you're most proud of from Audacity

Programming: Things



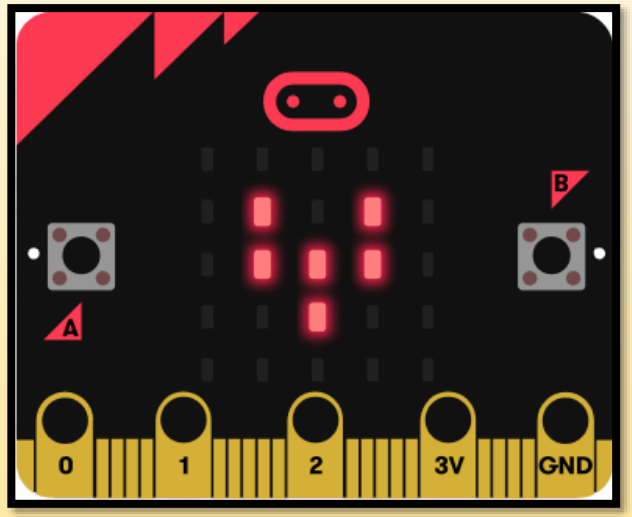
Programming: Things

Imagine **turning** your **ideas into reality** with just a **few lines of code**. With the micro:bit, you can create interactive projects that light up, move, and communicate. This unit introduces you to physical computing, where a small device can lead to big changes. From flashing LEDs to detecting motion, the micro:bit is packed with sensors and components that will **bring your code to life in exciting ways**.

Refresh your Python skills and discover the magic of combining programming with physical objects. You'll write and execute Python programs on the micro:bit, familiarising yourself with the development environment and practicalities of flashing your programs. **Coding becomes even more enjoyable when your programs can interact with the real world.**

Learn to **connect** the micro:bit to external hardware using its GPIO pins, expanding its capabilities and opening up endless project possibilities. Imagine creating a device that sends and receives messages wirelessly or controls lights and sounds based on sensor inputs. **Design and build your own physical computing project**, bringing together everything you've learnt into a meaningful creation. This unit promises a thrilling journey of creativity and innovation, inspiring you to explore the endless potential of the micro:bit.

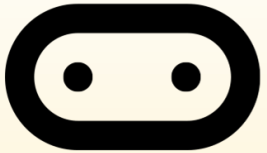
An example of your work




```
1  from microbit import *
2
3  while True:
4      display.show(Image.HEART)
5      sleep(500)
6      display.show(Image.HEART_SMALL)
7      sleep(500)
8
```

"The best way to create the future is to invent it."
Steve Jobs – Inventor of the iPhone

Programming: Things



Representations: Going Audio-Visual Keywords	
Micro:bit	A small programmable device equipped with sensors, buttons, and an LED display, used for learning and experimenting with coding and physical computing.
Python	A high-level programming language that is easy to read and write, often used for teaching coding concepts and controlling devices like the micro:bit.
Sensor	A device that detects and responds to changes in the environment, such as light, temperature, or motion. The micro:bit includes built-in sensors.
GPIO (General Purpose Input / Output)	Pins on the micro:bit that can be used to connect and control external hardware components like LEDs, buttons, and motors.
LED (Light Emitting Diode)	A small light source that glows when electricity passes through it. The micro:bit has a 5x5 LED grid for displaying patterns and information.
Flashing	The process of transferring a program from a computer to the micro:bit, allowing it to run the code and perform tasks.
Wireless Communication	The transmission of data without physical connections, using radio waves. The micro:bit can send and receive messages wirelessly.
Prototype	An initial version of a device or project built to test and demonstrate concepts before creating a final version.
Debugging	The process of finding and fixing errors or bugs in a program to ensure it runs correctly.
Physical Computing	The practice of using code to interact with and control physical devices, combining software and hardware to create interactive systems.

How can you use the micro:bit's built-in sensors and LED display to create interactive programs?

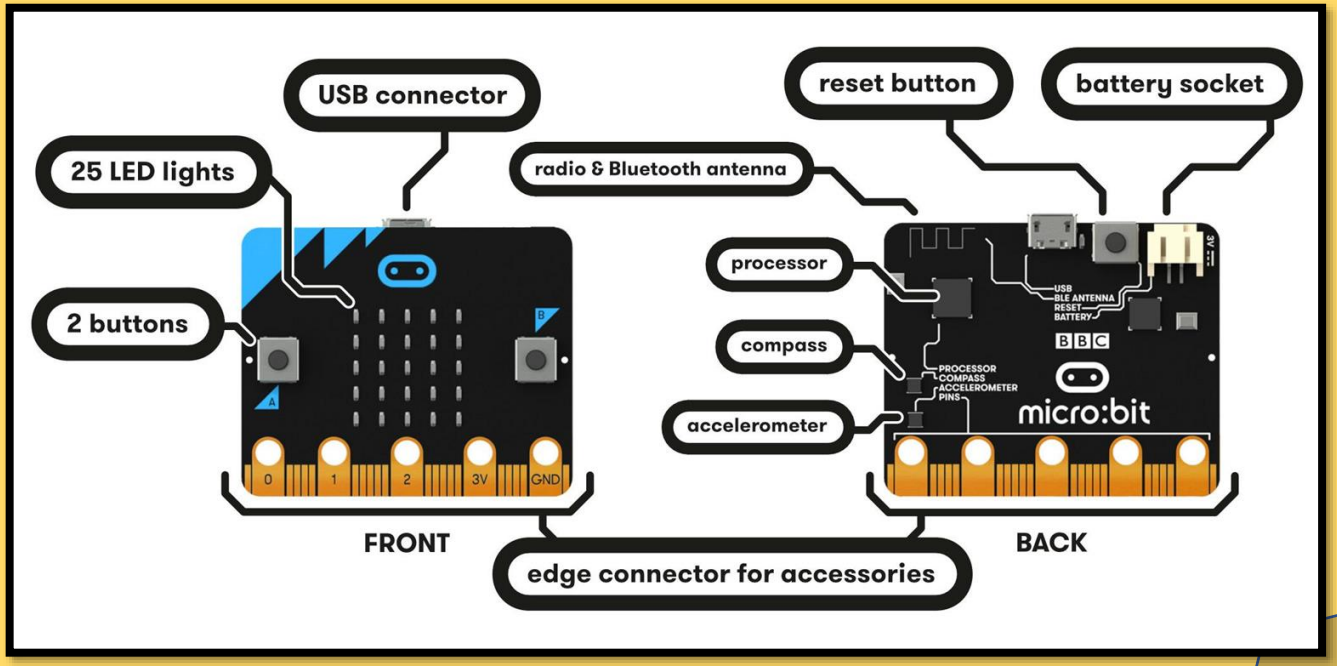
What steps are involved in designing and building a physical computing project with the micro:bit?

How can GPIO pins be used to extend the micro:bit’s capabilities by connecting external hardware components?

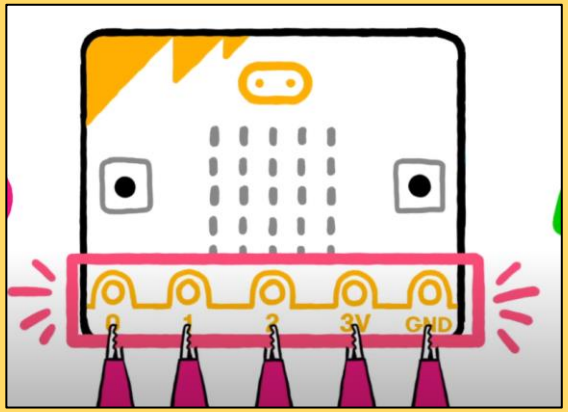
What techniques can be employed to debug and refine your code to ensure your physical computing project works as intended?

Key Knowledge

To know the Micro:bit's hardware components - including a 5x5 LED display, buttons, accelerometer, compass, temperature sensor and GPIO pins.



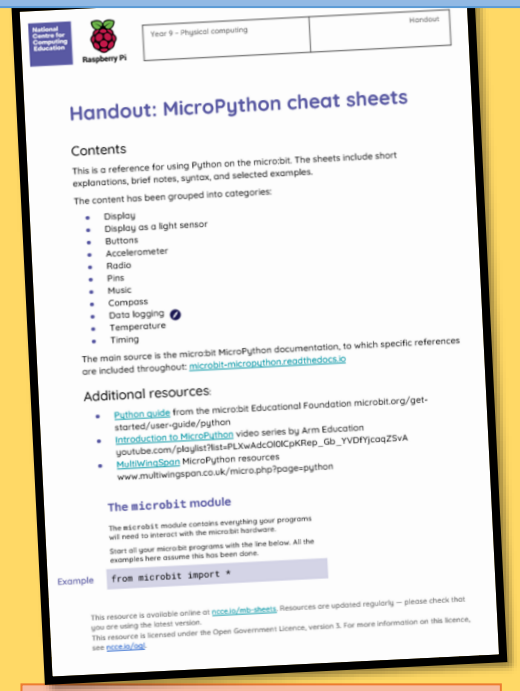
To know and be able to use the GPIO pins to connect external components like LEDs, buttons, and sensors to the Micro:bit.



Click here for a help video

Key Knowledge

To be able to use essential Python ***constructs*** include variables, loops, conditionals, and functions.



Handout: MicroPython cheat sheets

Contents

This is a reference for using Python on the micro:bit. The sheets include short explanations, brief notes, syntax, and selected examples.

The content has been grouped into categories:

- Display
- Display as a light sensor
- Buttons
- Accelerometer
- Radio
- Pins
- Music
- Compass
- Data logging
- Temperature
- Timing

The main source is the micro:bit MicroPython documentation, to which specific references are included throughout: microbit-micropython.readthedocs.io

Additional resources:

- Python guide from the micro:bit Educational Foundation microbit.org/get-started/user-guide/python
- Introduction to MicroPython video series by Arm Education youtube.com/playlist?list=PLXwAdcOlOICpKRep_Gb_YVDfYjcoqZSvA
- MultiWingSpan MicroPython resources www.multiwingspan.co.uk/micro.php?page=python

The microbit module

The microbit module contains everything your programs will need to interact with the micro:bit hardware.

Start all your micro:bit programs with the line below. All the examples here assume this has been done.

Example

```
from microbit import *
```

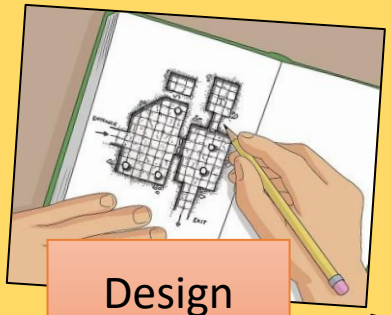
This resource is available online at ncce.io/mb-sheets. Resources are updated regularly — please check that you are using the latest version.

This resource is licensed under the Open Government Licence, version 3. For more information on this licence, see ncce.io/ogl

Click here for the programming cheat sheets 😊

To know how to design a Physical Computing Project

Brainstorm ideas



Design

Program

Build your project

Test

Refine

Please screenshot your multiple choice quiz results

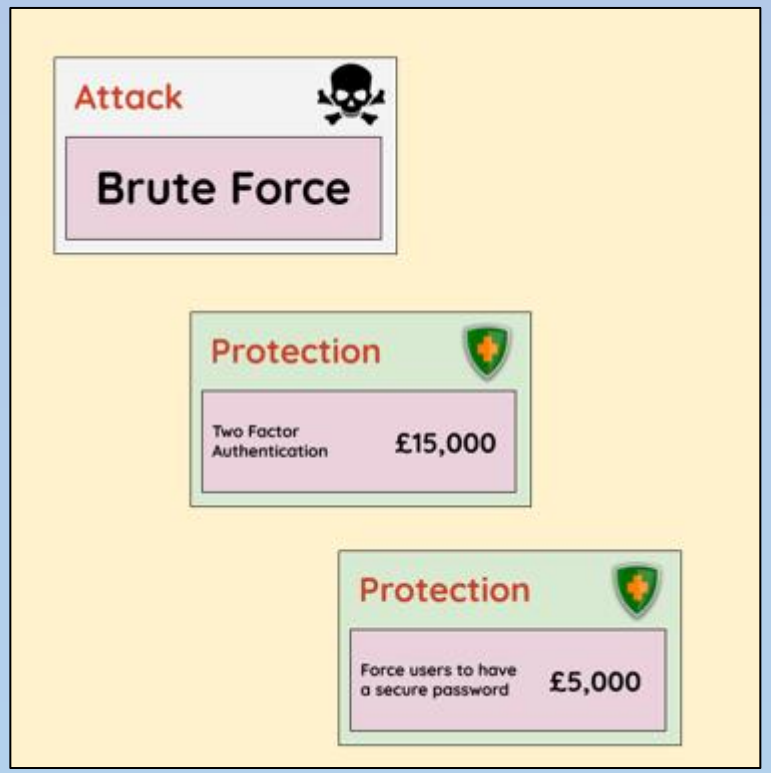
Please screenshot the code for your project

Introduction to Cybersecurity

Introduction to Cybersecurity

Imagine waking up to find all **your private information stolen by hackers**. Welcome to cybersecurity, where we learn to combat cybercriminals and protect our digital lives. In this unit, you'll discover why your data is so valuable and how hackers use sneaky tricks to steal it. You'll explore hacking, malware threats, and the devastating effects of cyberattacks.

But **don't worry**, you'll also learn how to defend yourself with firewalls, anti-malware, and strong passwords. Get ready to become a digital defender and **stay safe in our connected world**.



"Cybersecurity is a shared responsibility, and it boils down to this: In cybersecurity, the more systems we secure, the more secure we all are." - Jeh Johnson - Secretary of Homeland Security under President Barack Obama.

Key Terms	
Data	Information that companies collect about you, like your name, age, or what you like to do online.
Cybercriminals	People who use computers to steal data or cause problems on the internet.
Social Engineering	Tricks used by cybercriminals to get you to give them your personal information, like pretending to be someone you trust.
Hacking	Breaking into computers or networks without permission to steal data or cause damage.
Malware	Harmful software, like viruses, that can damage your computer or steal your information.
DDoS Attack	When many computers are used to flood a website with traffic, making it slow or unable to work.
Firewall	A security system that helps protect your computer from unauthorized access or attacks.
Anti-Malware	Software that helps detect and remove malware from your computer.
Password Authentication	A security process where you use a password to prove your identity and access your accounts.
Data Protection Act	A law that helps keep your personal information safe and ensures it is used correctly.

Key questions

What is the difference between data and information?

What are some strategies to minimize data security risks due to human error?

How can you protect a network from common security threats?

What are hacking and DDoS attacks, and how do they differ from each other?

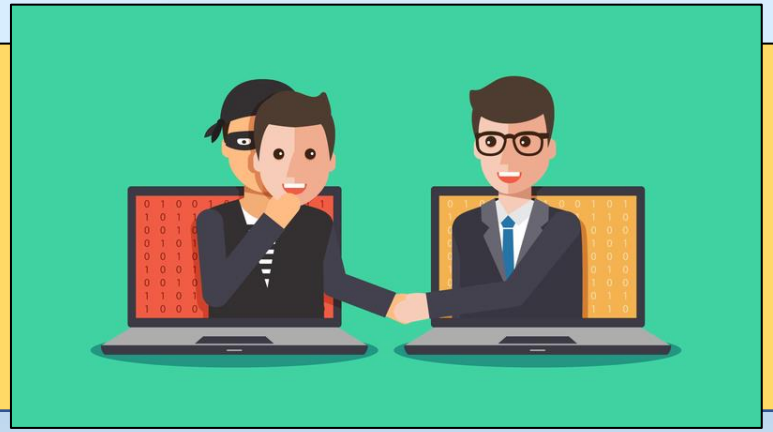
Key Knowledge

Understanding what data is, how it differs from information, and why it is valuable to companies and cybercriminals.

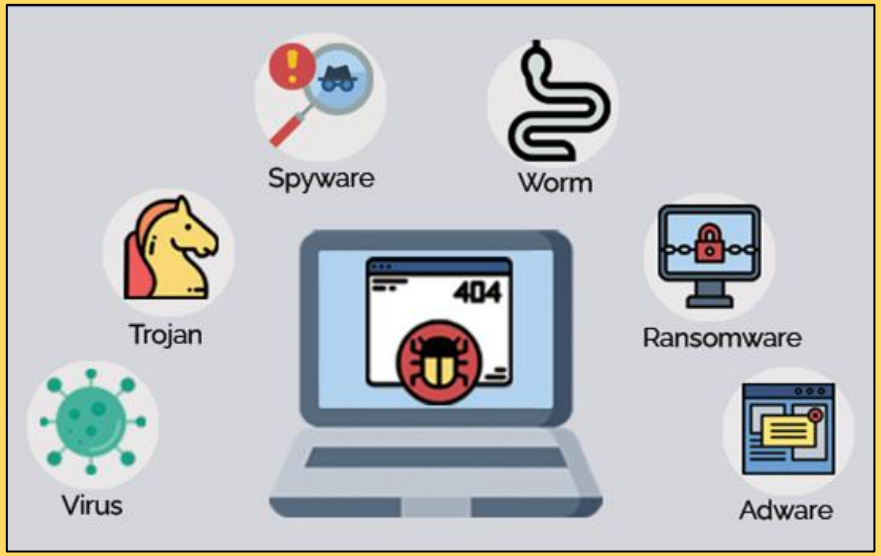
This information can then be used to generate money

Key Knowledge

Learning about the tactics used by cybercriminals to trick people into giving away their personal information.



Identifying various types of malware, such as viruses, trojans, and ransomware, and understanding the damage they can cause.



Understanding laws like the Data Protection Act and Computer Misuse Act that help protect data and prevent cybercrime.

Computer Misuse Act 1990

Section 1 **Unauthorised access to computer material.**

Adam watches a friend entering their username and password. Adam remembers their login details and without their permission, later logs in and reads all their messages.

Section 2 **Unauthorised access with intent to commit or facilitate commission of further offences.**

Raj's teacher leaves their tablet on their desk. Without their permission, Raj accesses their online shopping account and buys items with the attached credit card.

Section 3 **Unauthorised acts with intent to impair, or with recklessness as to impairing, operation of a computer.**

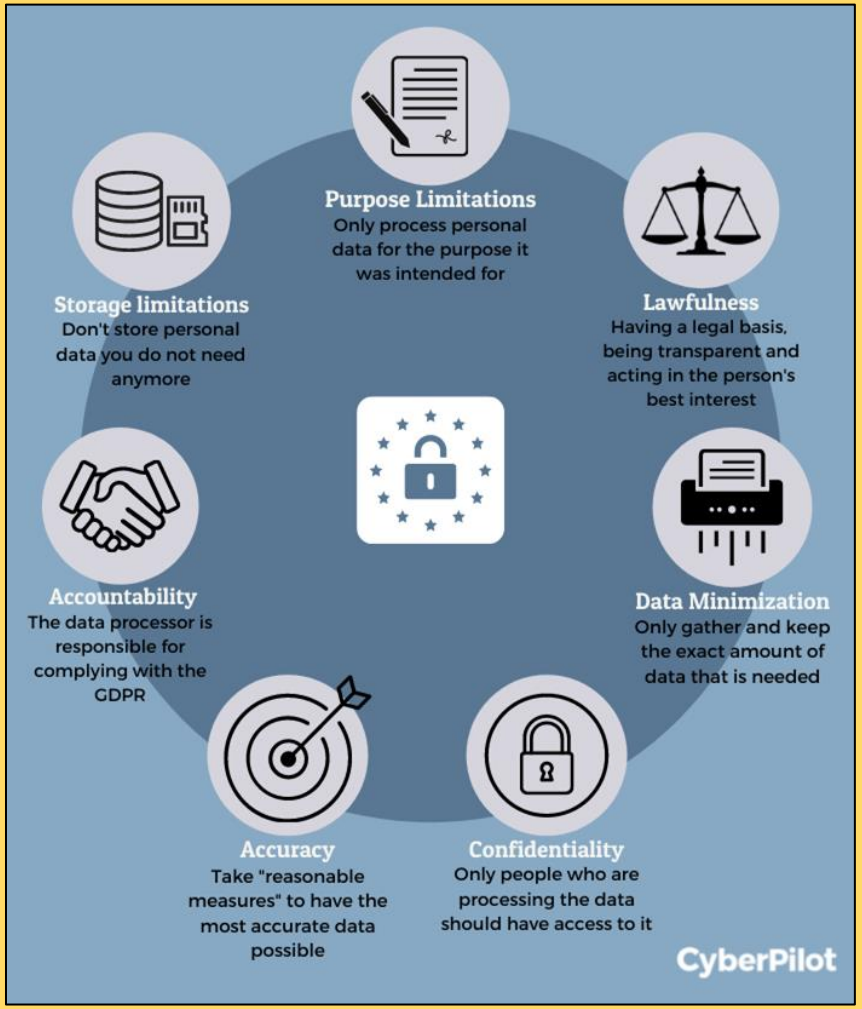
Sarah is playing an online game with a friend who scores higher than her. Sarah uses a 'Booter' tool knowing it will knock them offline, so she can win the game.

Section 3ZA **Unauthorised acts causing, or creating risk of, serious damage.**

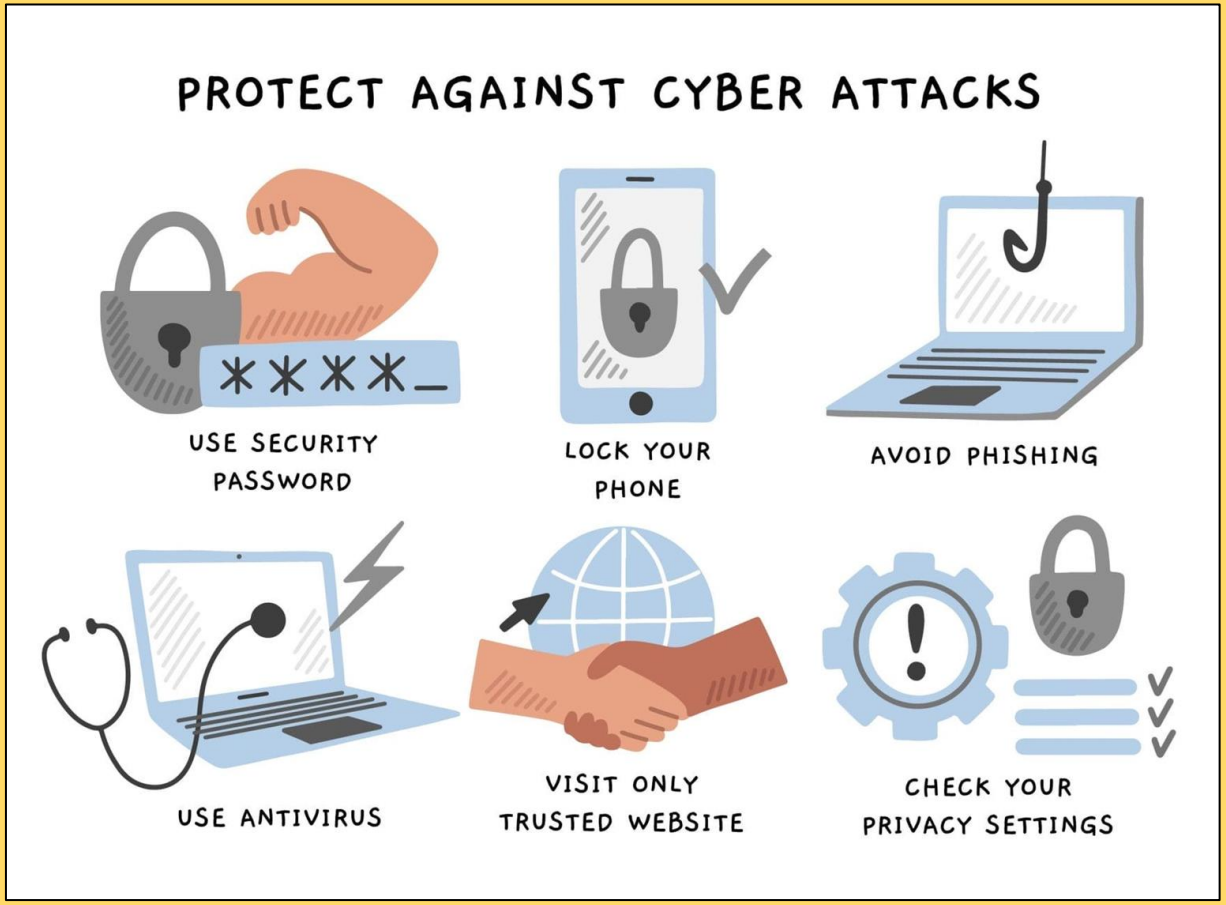
Kim hacks a phone company. This hack stops some people phoning the Police when they are in danger. They didn't mean for this to happen but they were reckless.

Section 3A **Making, supplying or obtaining articles for use in offence under Sections 1, 3 or 3ZA.**

Robin downloads software so they can bypass login credentials and hack into a friend's laptop, however they've not had a chance to use it yet.



Learning about different methods to protect computers and networks, including firewalls, anti-malware software, and strong passwords.



Please screenshot your multiple choice quiz results