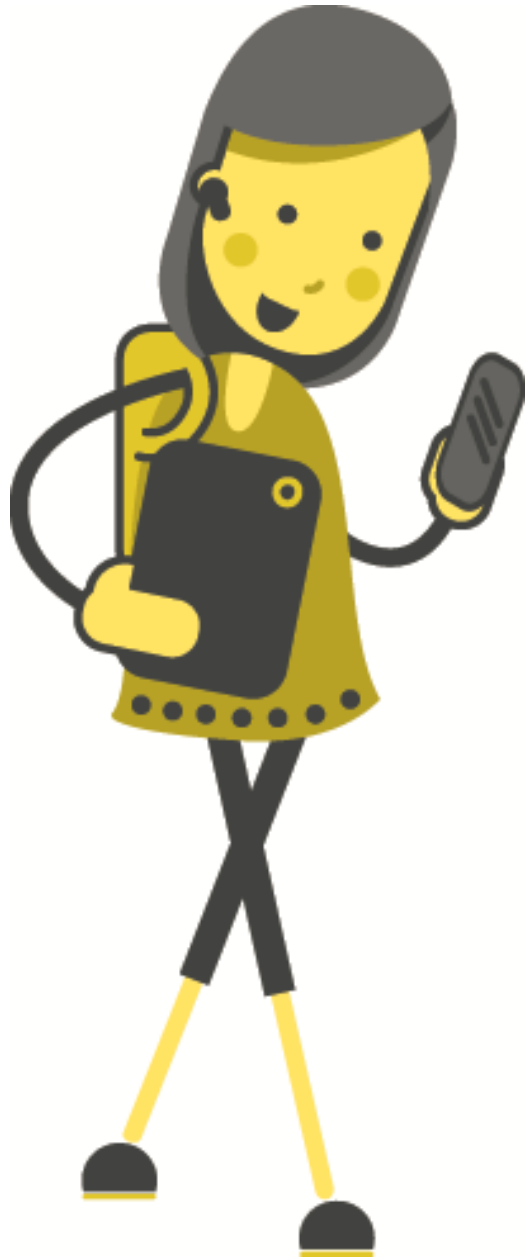


technocamps



Algorithms



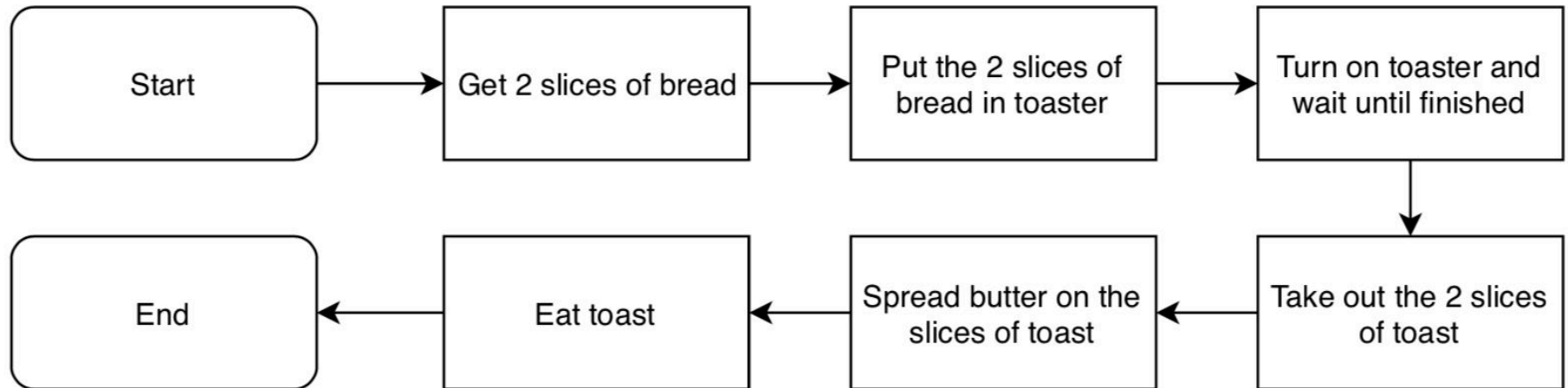


Activity: What is an Algorithm?

Algorithms

An Algorithm is a set of **simple instructions** that are done in a **certain order** to **solve a problem**.

We use algorithms all the time in everyday life. An example is making and eating toast.



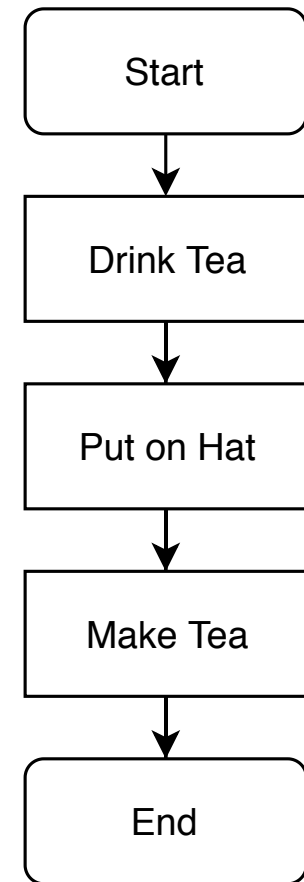


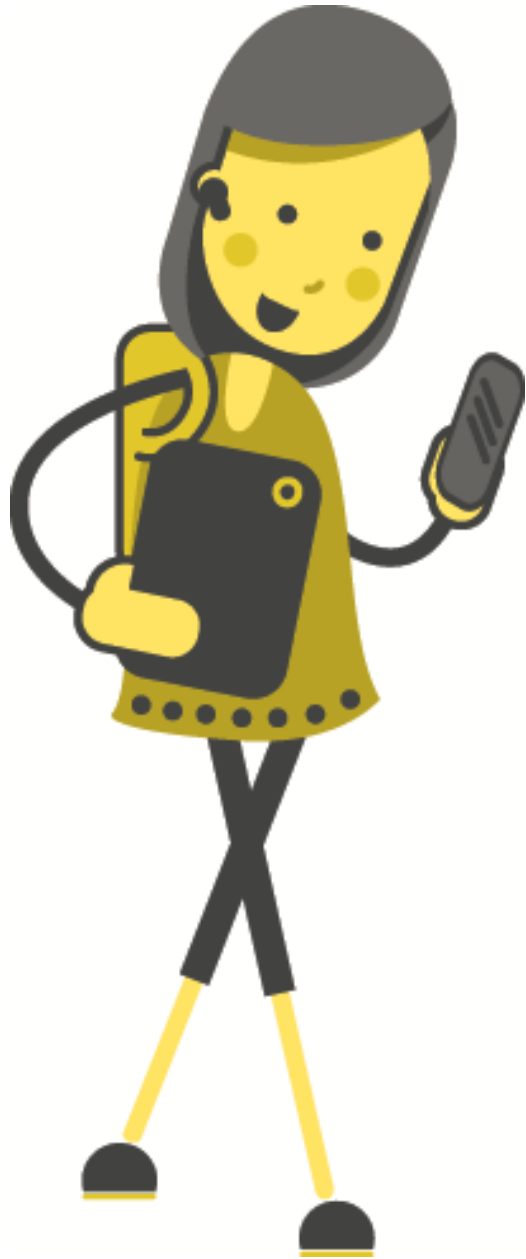
Activity: Making a Cup of Tea

Algorithms

It is important to remember when writing an algorithm to keep instructions:

- Simple.
- In the correct order.
- Unambiguous
- Relevant to solving the problem at hand.





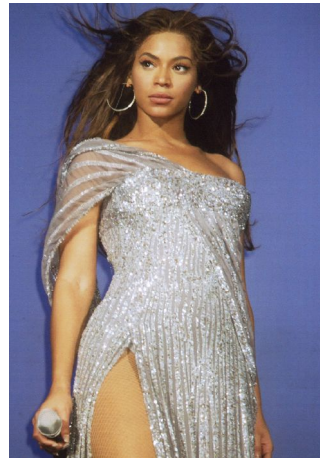
Activity: Define Algorithms

Activity: Guess Who

Cristiano Ronaldo



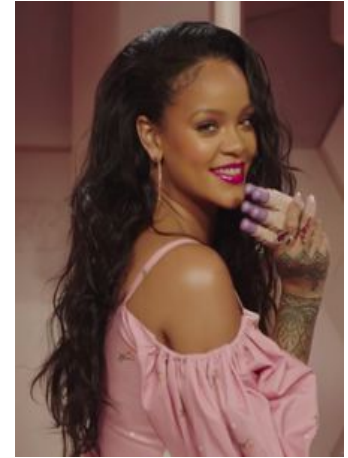
Beyoncé



Lionel Messi



Rihanna



Serena Williams



Drake

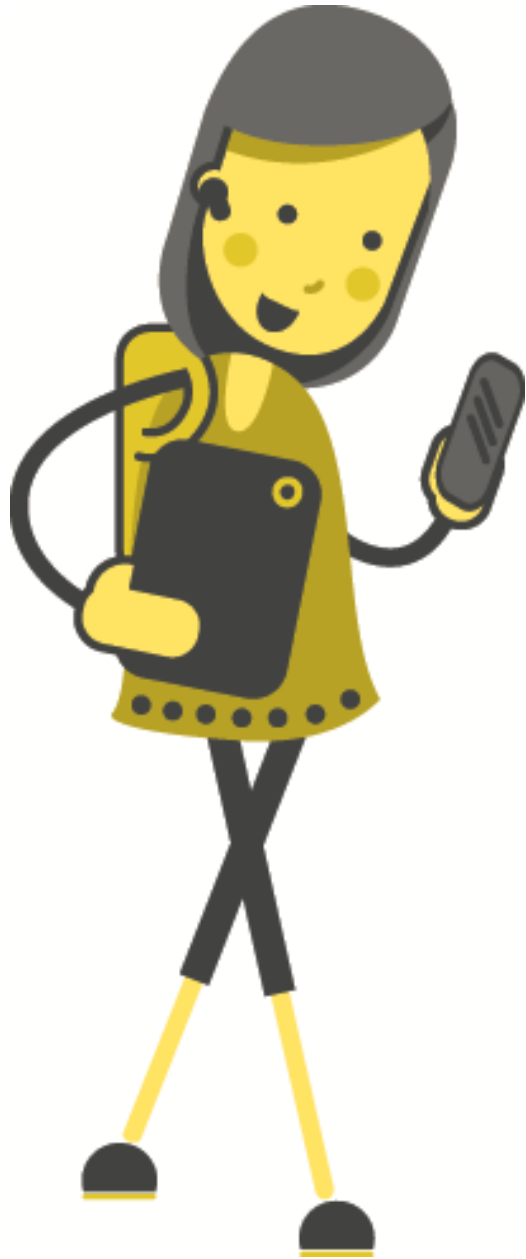


Kylie Jenner



Robert Downey Jr.





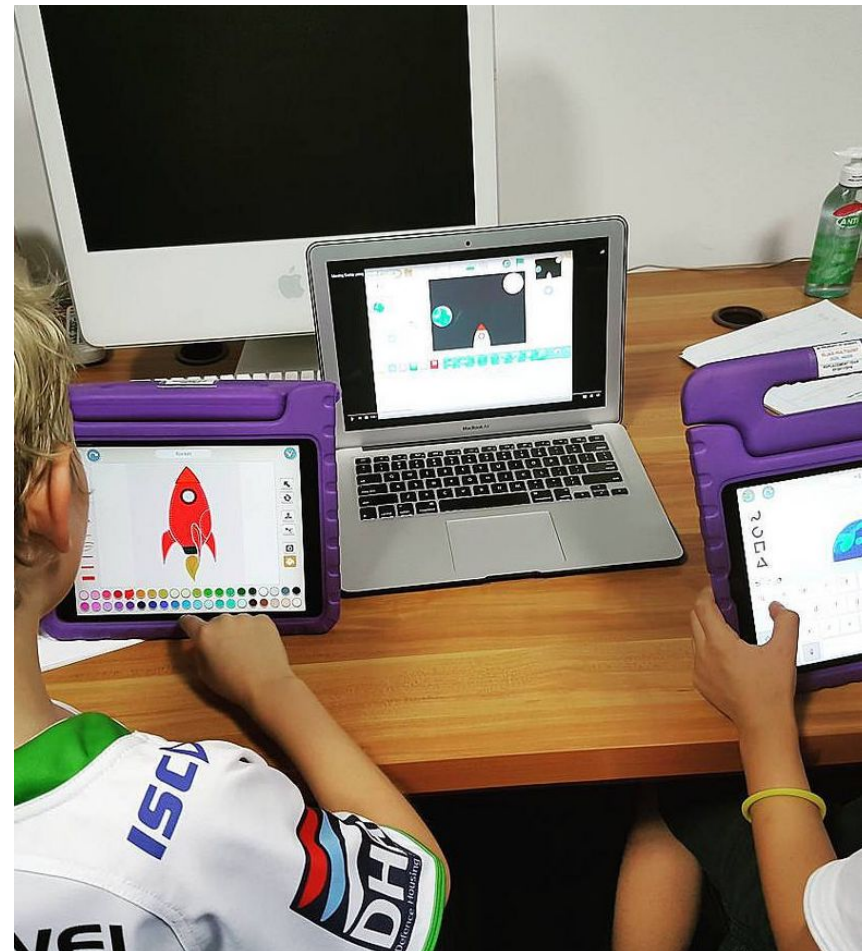
Activity: What is Decomposition?

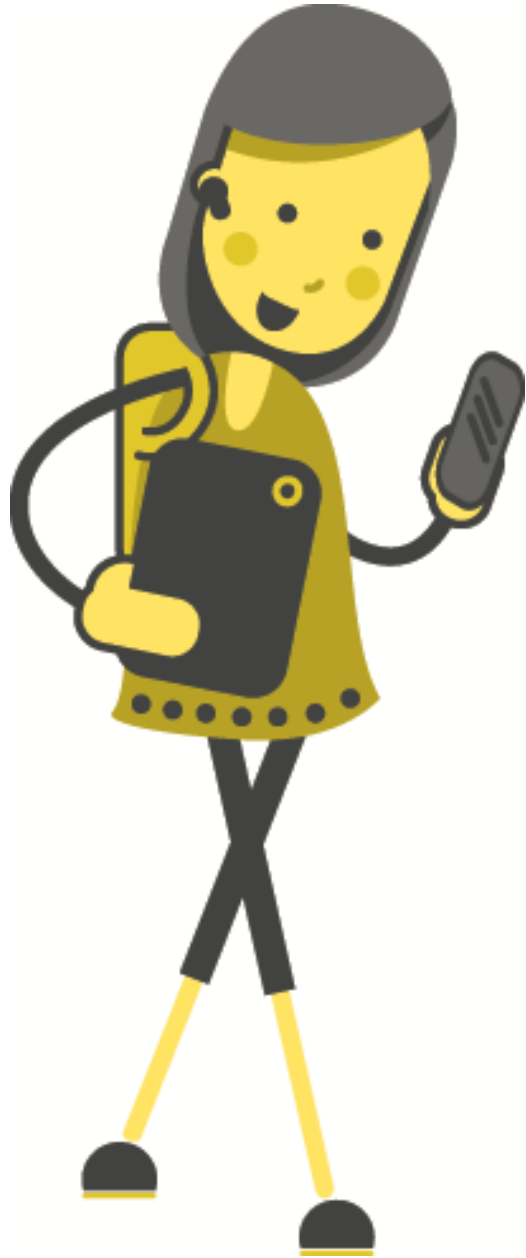
Decomposition

Decomposition is the process of breaking a complex problem into smaller component parts.

Real world examples of using decomposition:

- Creating a video game.
- Complex maths problems.
- Cooking.
- Cleaning your room!





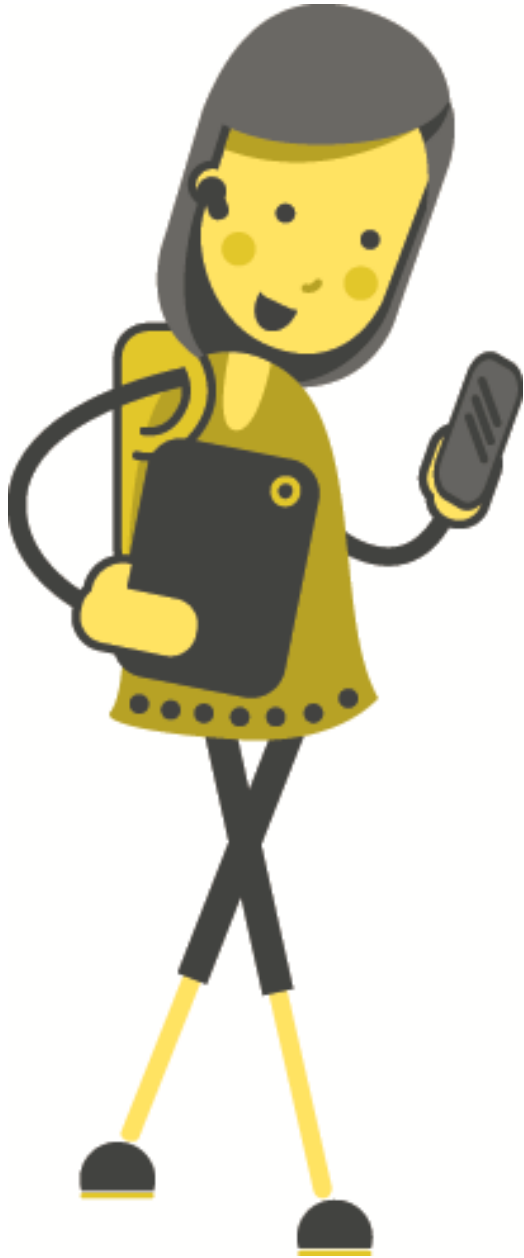
Activity: LEGO Building

LEGO Building Reflection

The activity was a great way of showing the importance of giving clear, simple and detailed instructions.

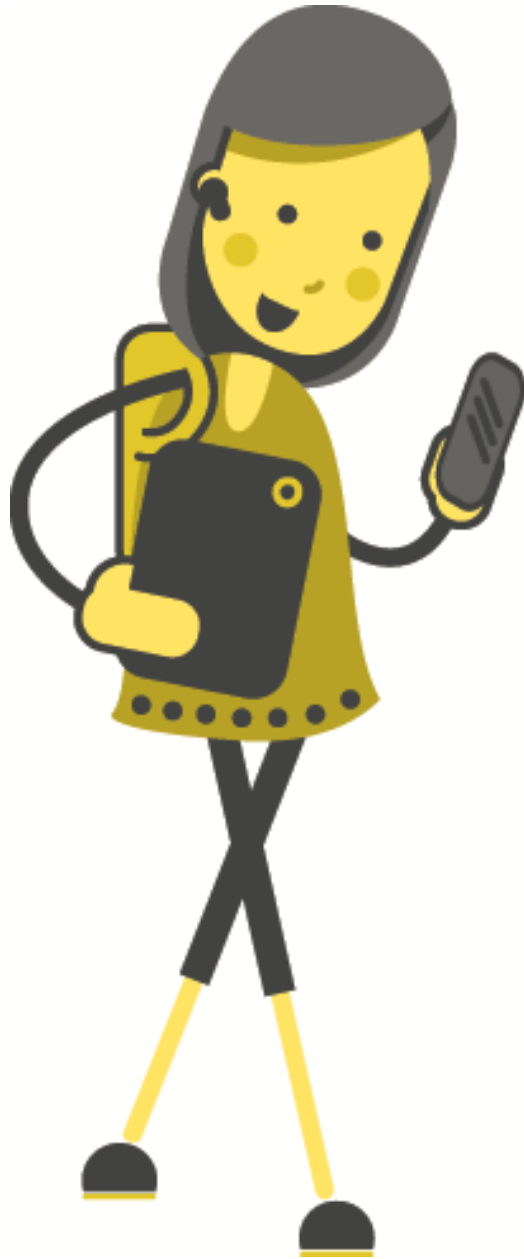
It highlights the importance of decomposition when faced with a complex task. Most complex tasks can be broken down into smaller problems.





Activity: Get Artistic





Activity: Get Artistic Reflection

Abstraction

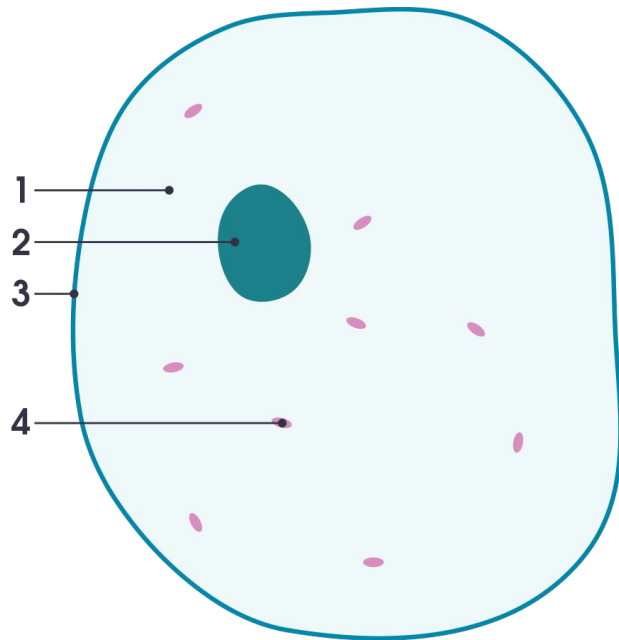
Abstraction – is the process of removing unnecessary detail and simplifying. Abstraction is used to remove unnecessary detail from a real-world situation and to model the simplified result in an algorithm or program.

Real world examples of abstraction in action:

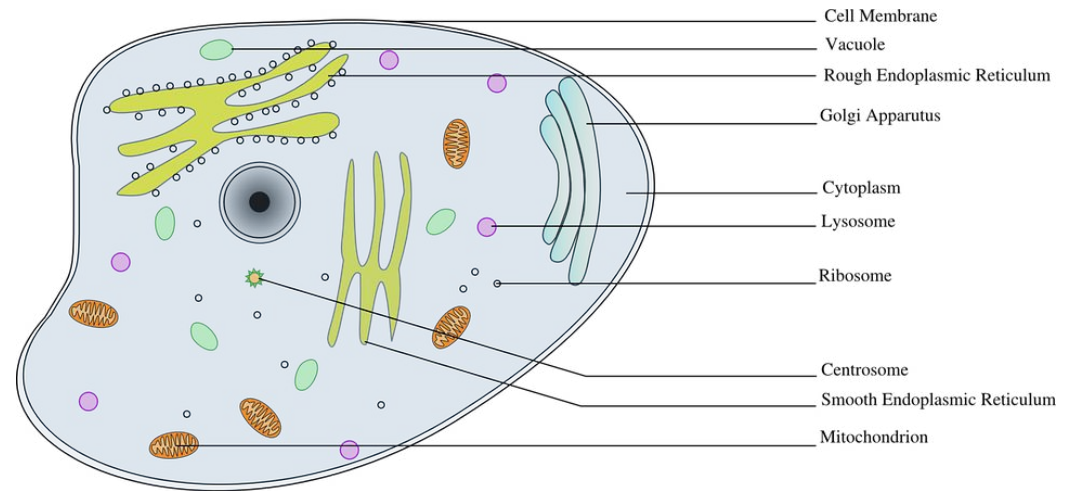
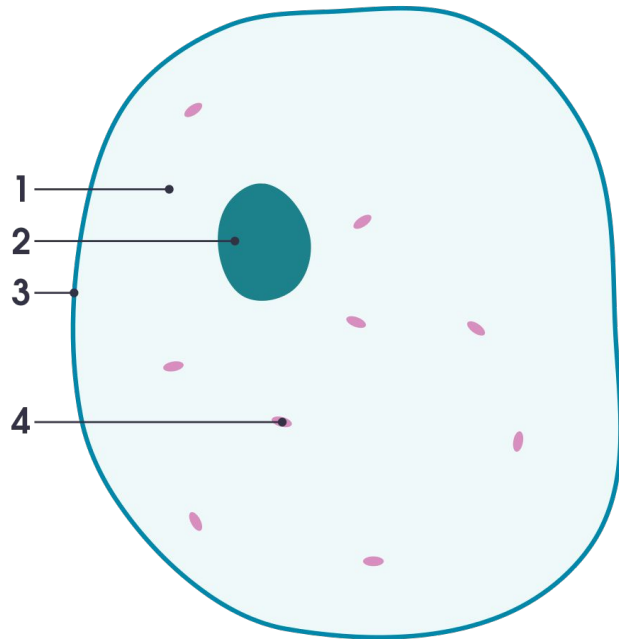
- In driving.
- In programming.
- In teaching.



Teachers using Abstraction



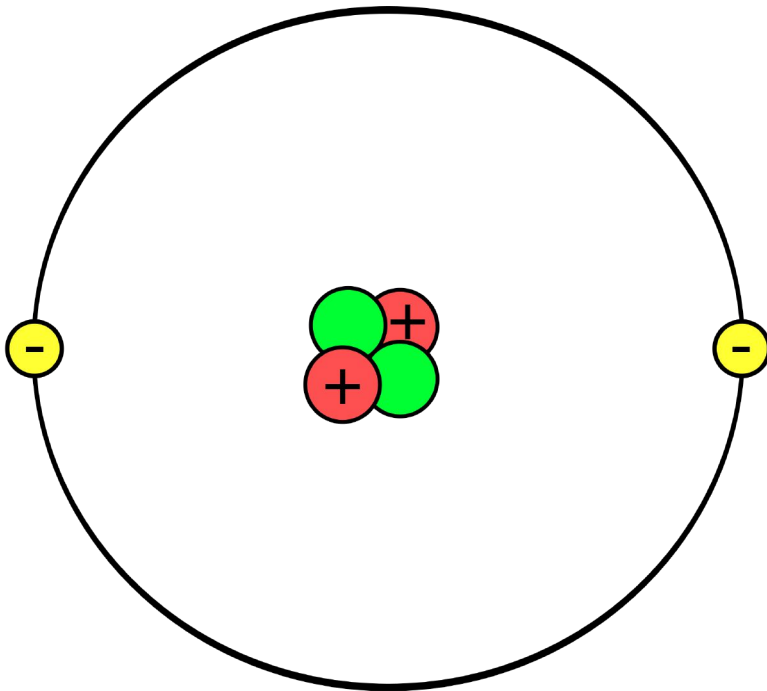
Teachers using Abstraction



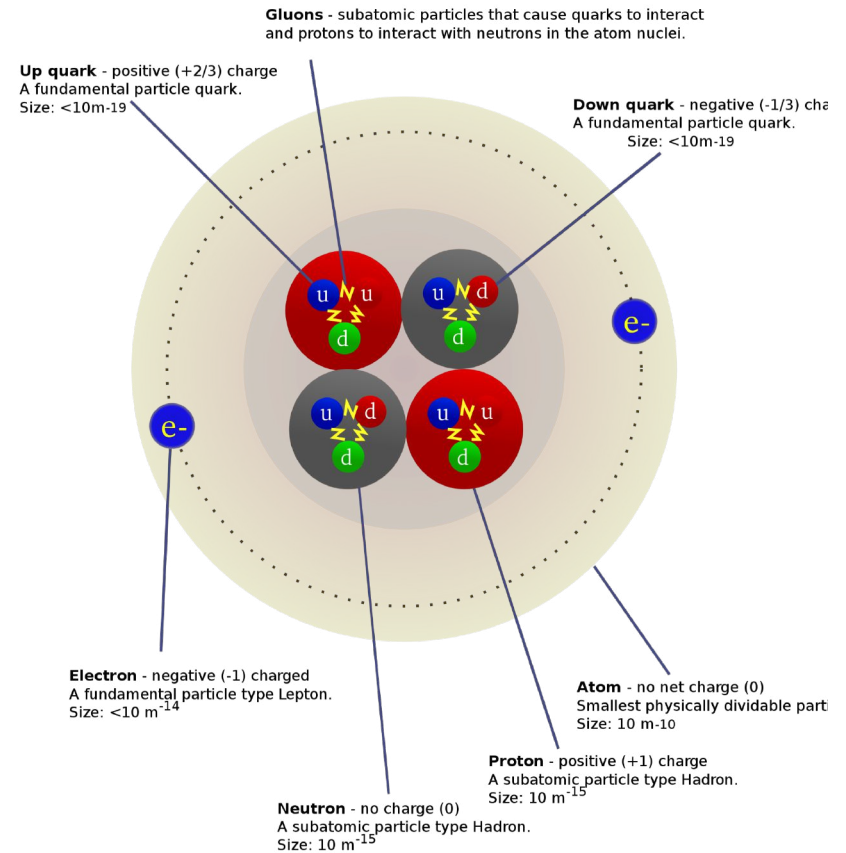
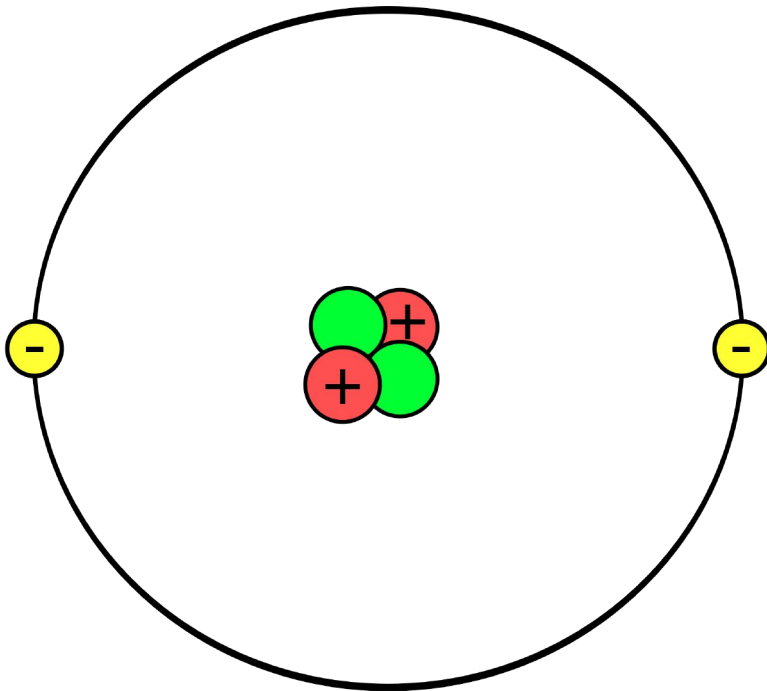
Cross Section of an Animal Cell

1. Cytoplasm, 2. Nucleus, 3. Cell Membrane, 4. Mitochondrion

Teachers using Abstraction

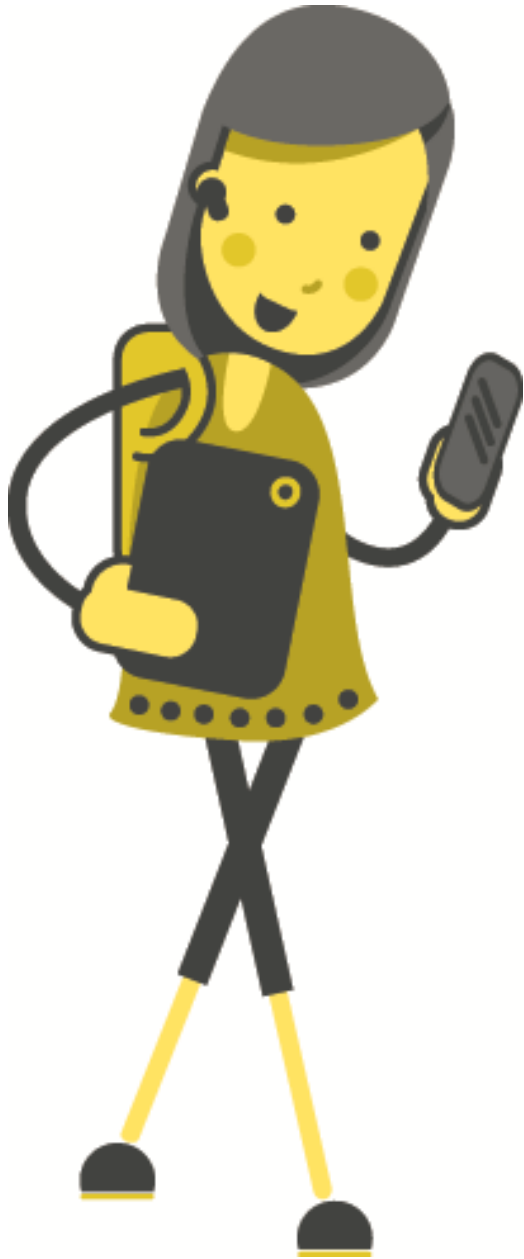


Teachers using Abstraction




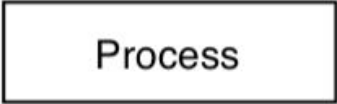
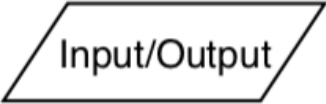

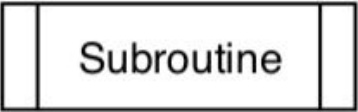



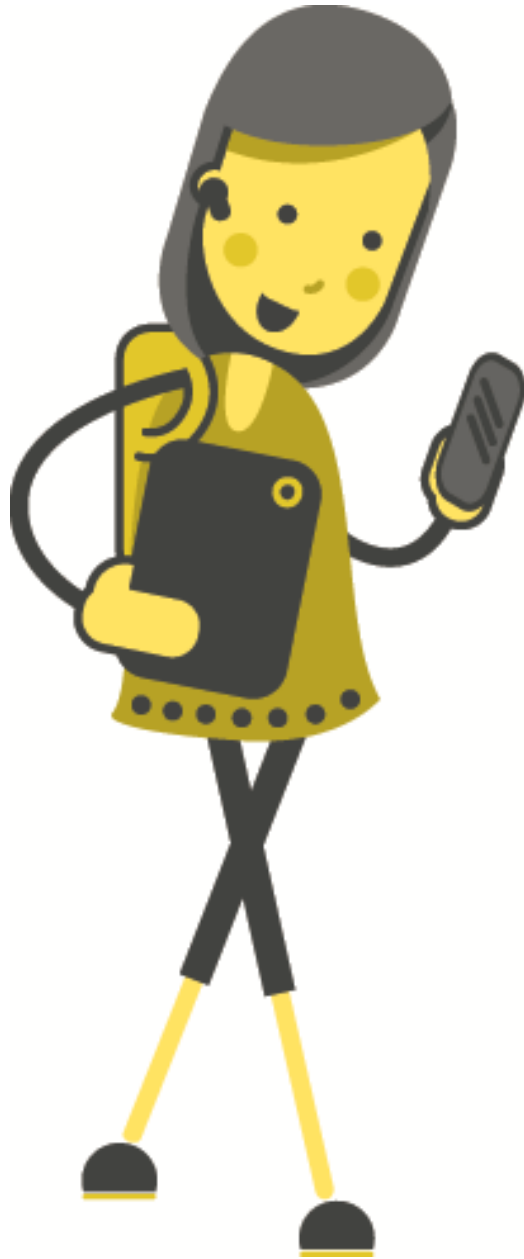
Activity:
What is
Abstraction?



Flowcharts

Flowchart Conventions

Name	Symbol	Usage
Start or Stop/End		Signifies the start or end of a sequence.
Process		An instruction.
Input/Output		Data received or sent by a computer.
Decision		A condition which is either true or false.
Subroutines		Calls a subroutine
Direction of Flow		Connects symbols.

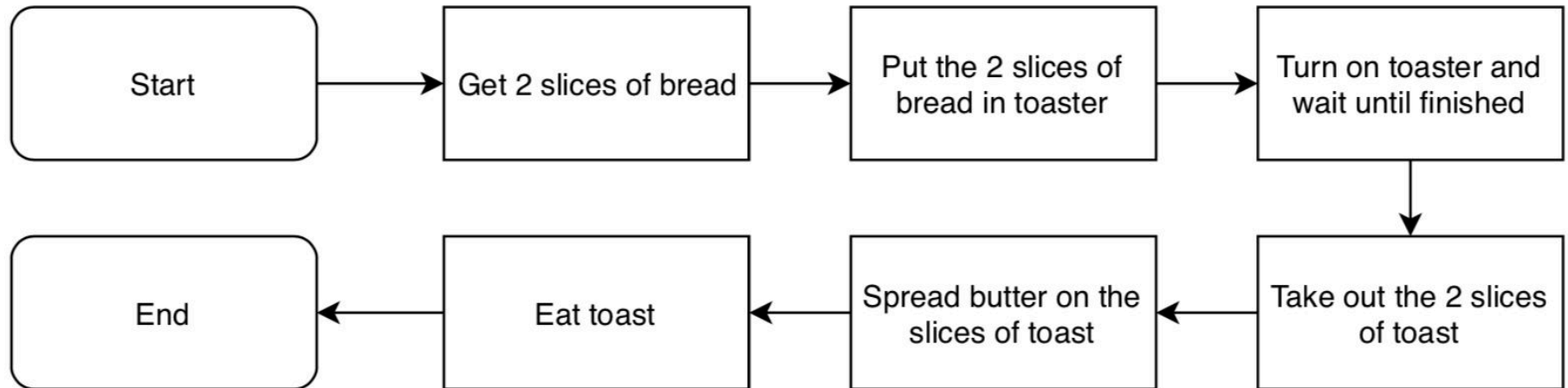


Sequence, Selection, Iteration and Subroutines

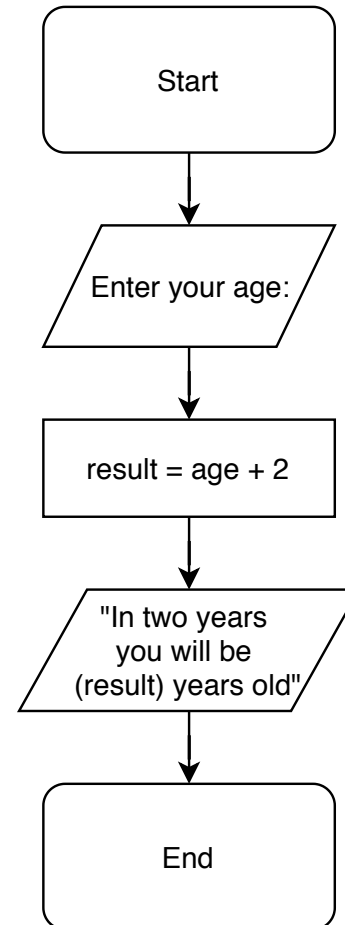
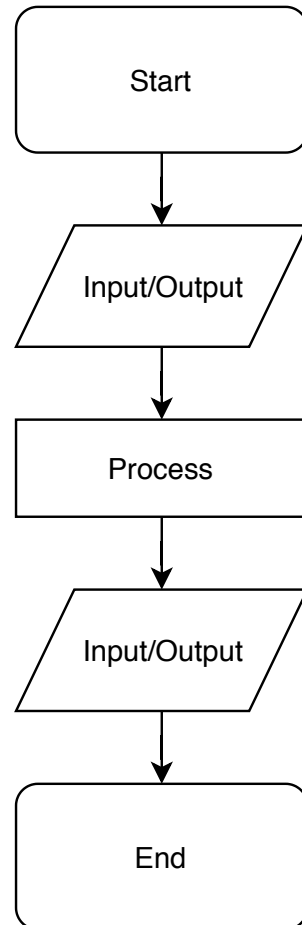
Sequence

Sequence: An action, or event, leads to the next ordered action in a predetermined order.

Recall the making and eating toast example:

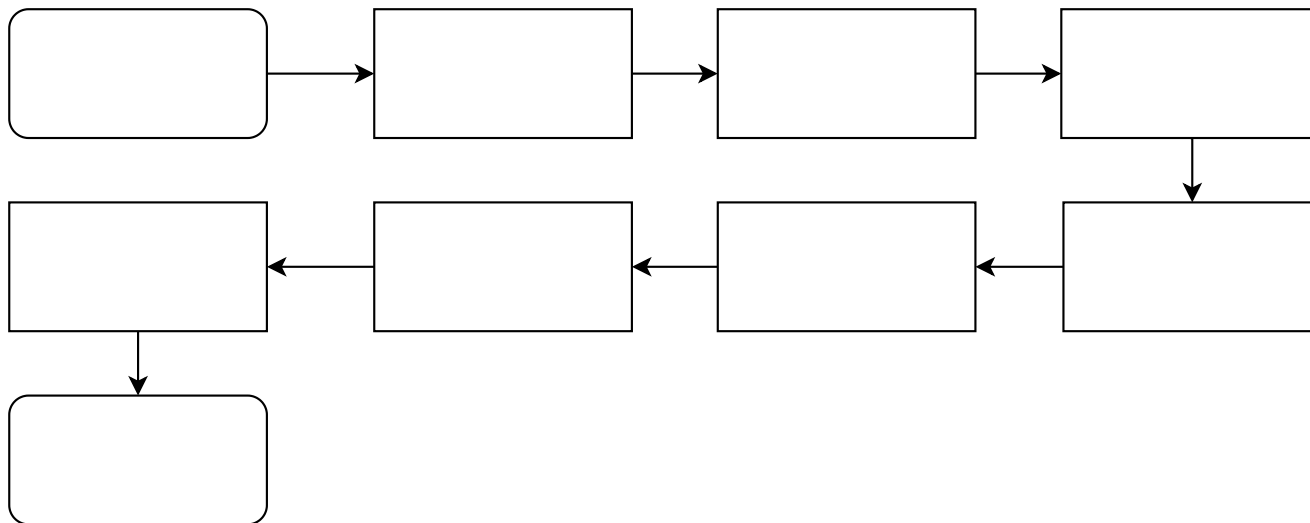


Sequences

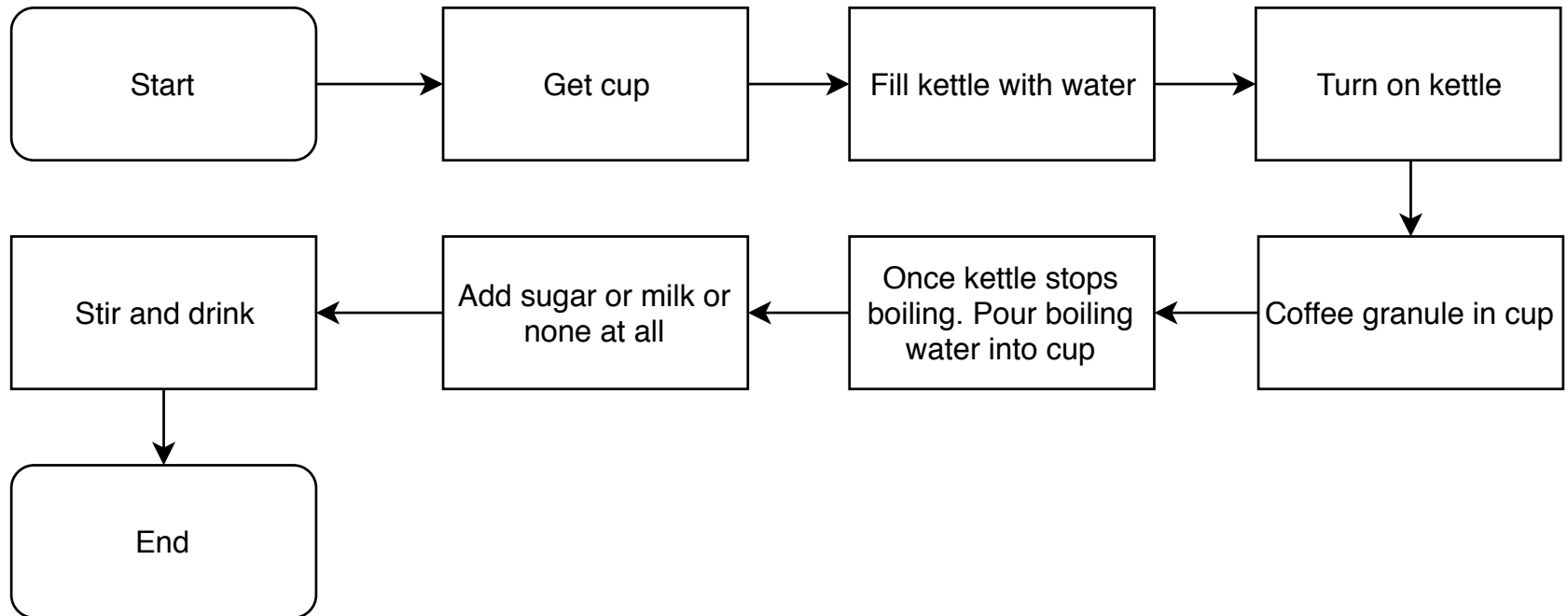


Activity: Making a Cup of Coffee

In your work books try to put all of the instructions for making a cup of coffee into the correct sequence.



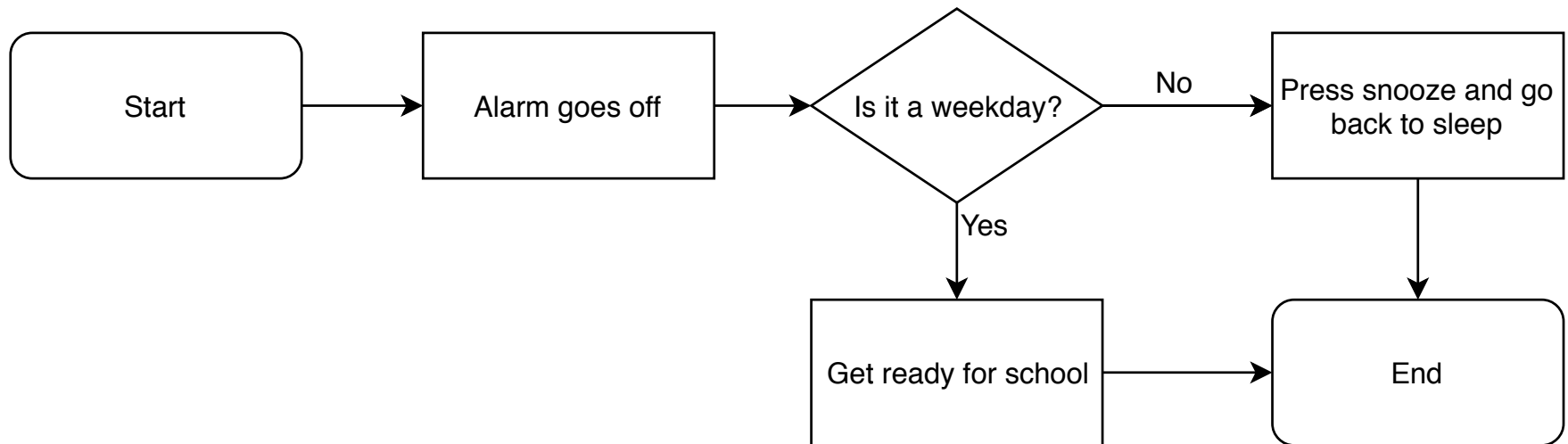
Making a Cup of Coffee Solution



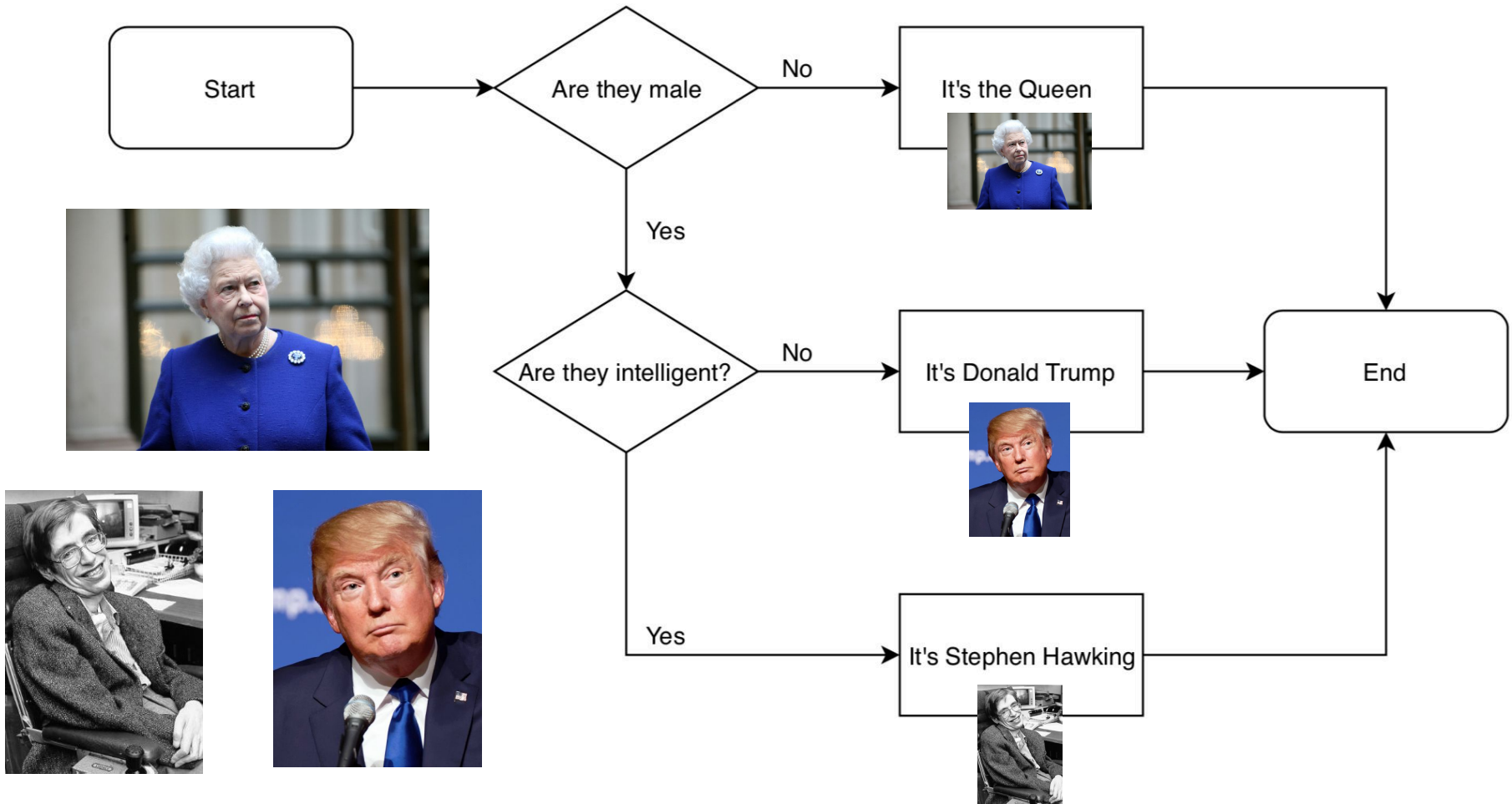
Selection

In selection, a question is asked, and depending on the answer, the program takes one of the two courses of action.

Example: Waking up in the morning



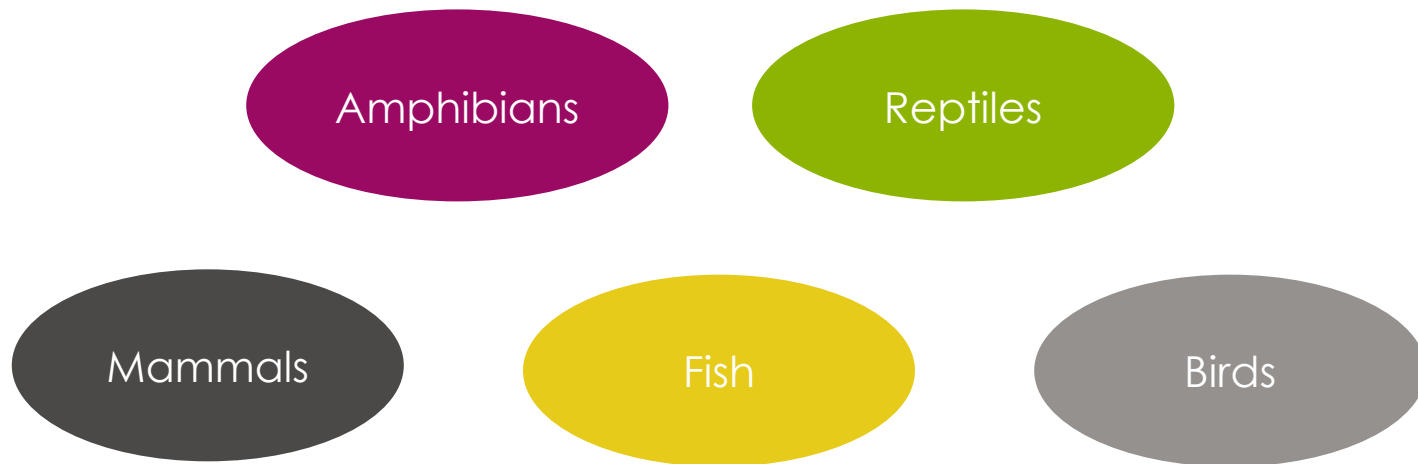
Selection Flowchart



Activity: Vertebrates Flowchart

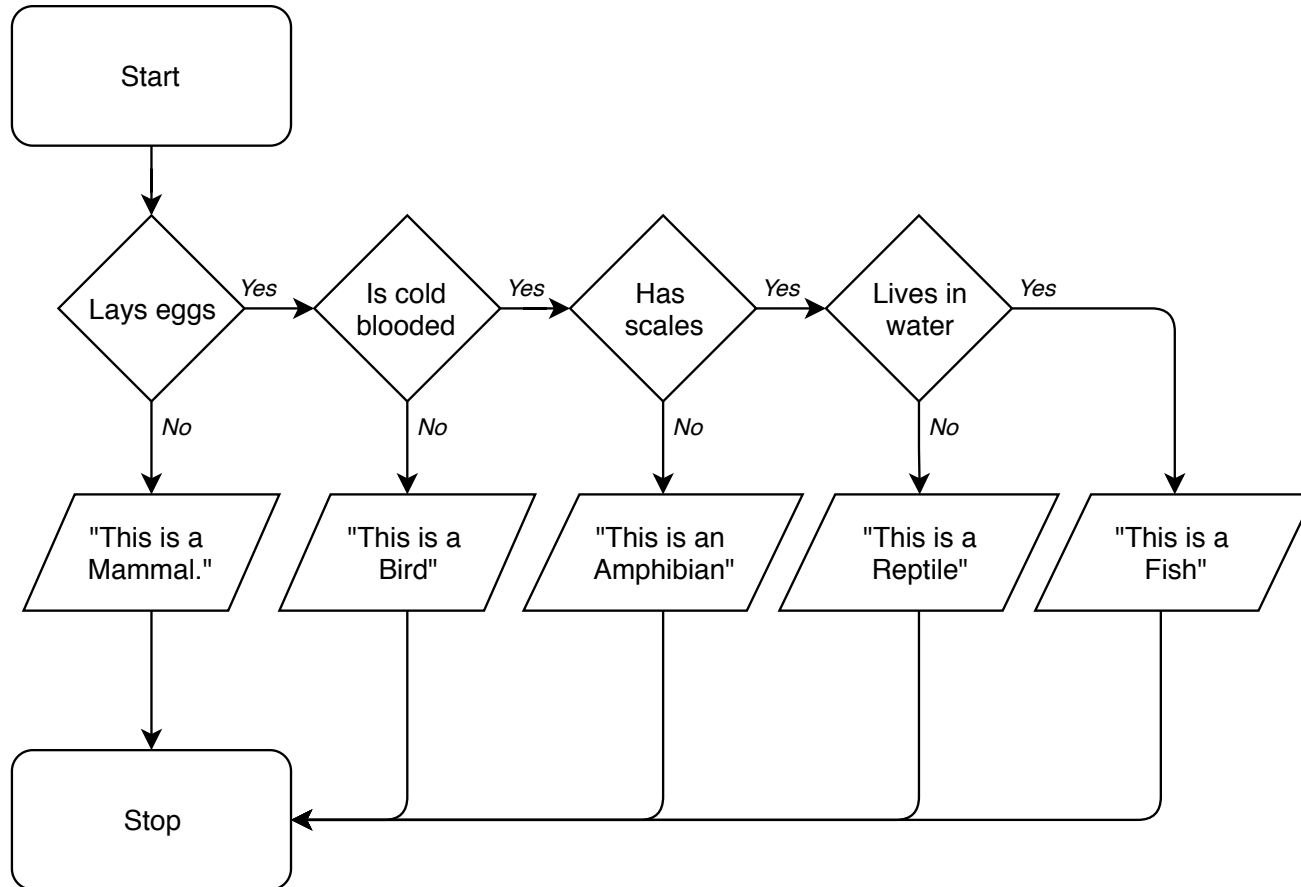
Write a flowchart with appropriate questions to be able to differentiate between each of the 5 classes of Vertebrates.

Think about what differentiates them and how you would structure your flowchart.



(You can research vertebrate classification on your computers)

Vertebrates Solution

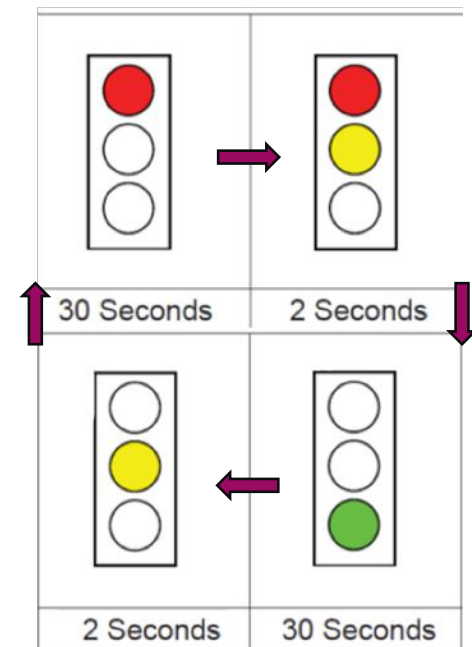


Iteration

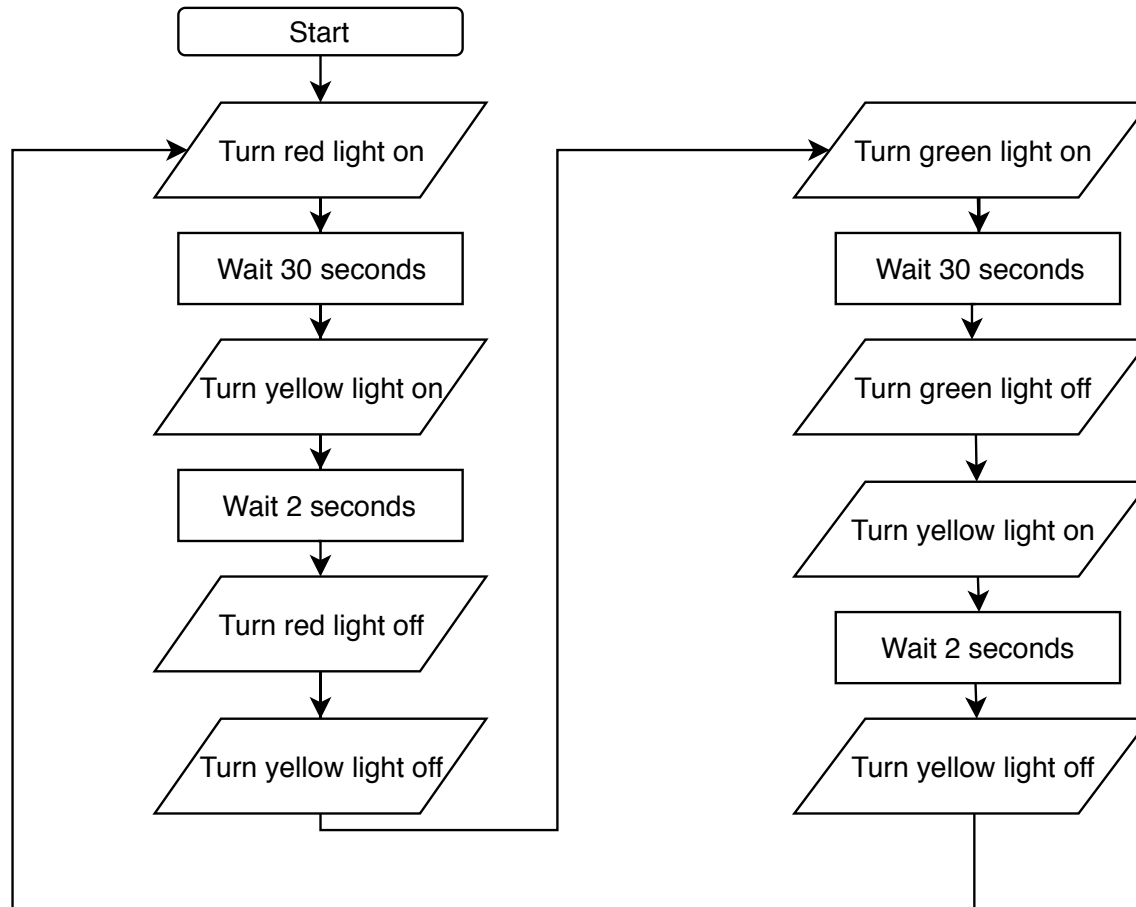
An iteration is a single pass through a set of instructions. Most programs contain a set of instructions that are executed over and over again. The computer iterates through the loop. Some processes include steps or a series of steps that are iterated.

Example: Simple Traffic Lights

- What instructions would you use for this process?
- What needs to be iterated?
- How could you show that in a flowchart?
- Does this process ever end?



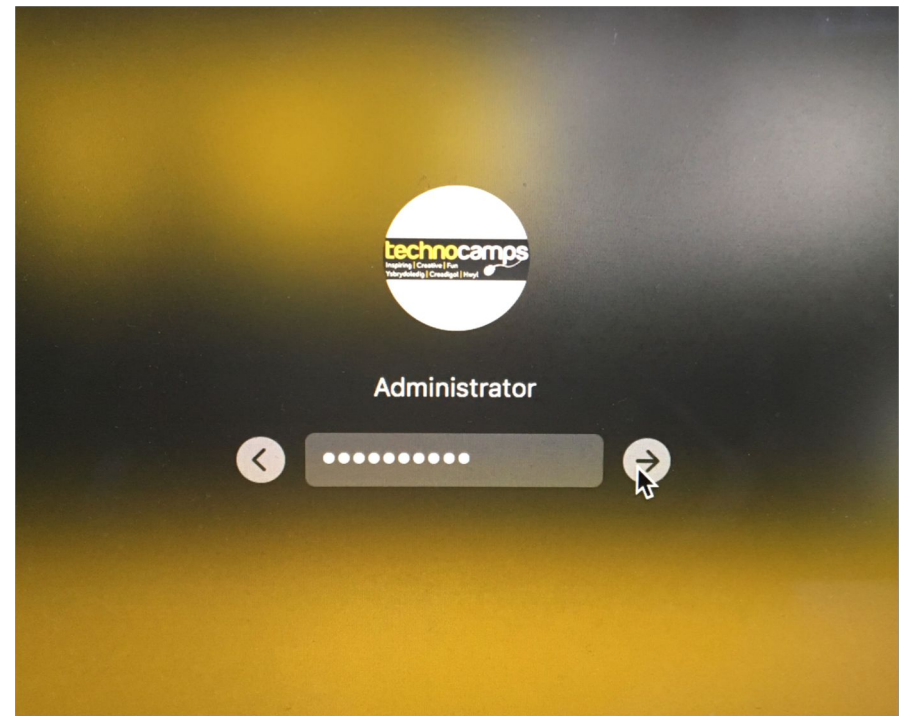
Simple Traffic Lights



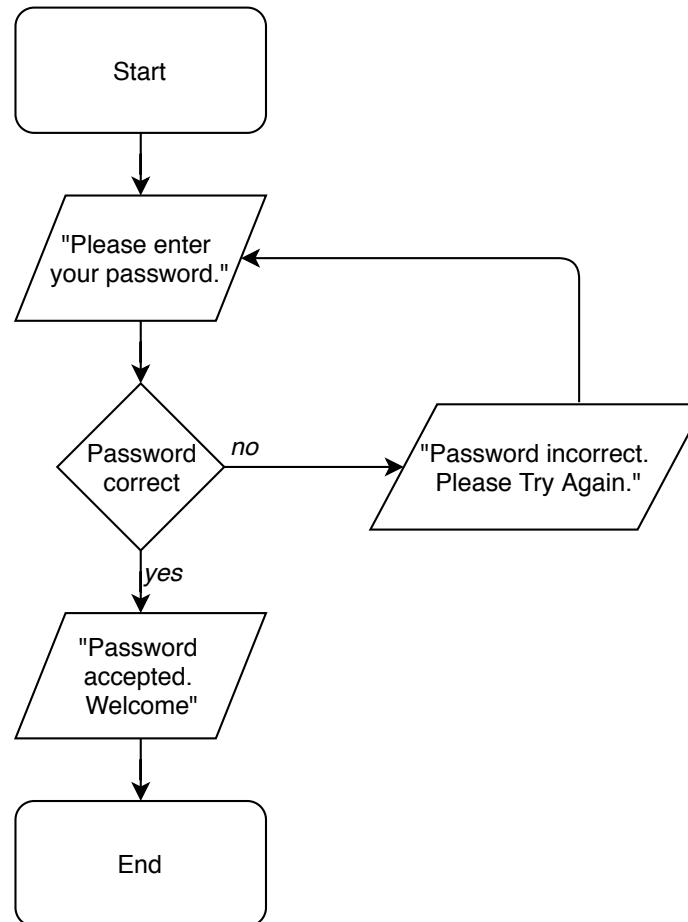
Activity: Login System Flowchart

Create a flowchart for a program which asks a user for a password:

- If the password is incorrect it should output the message **“Password incorrect. Please Try Again.”** and take the user back to the log in screen.
- Otherwise it should say **“Password accepted. Welcome!”** and end the program.

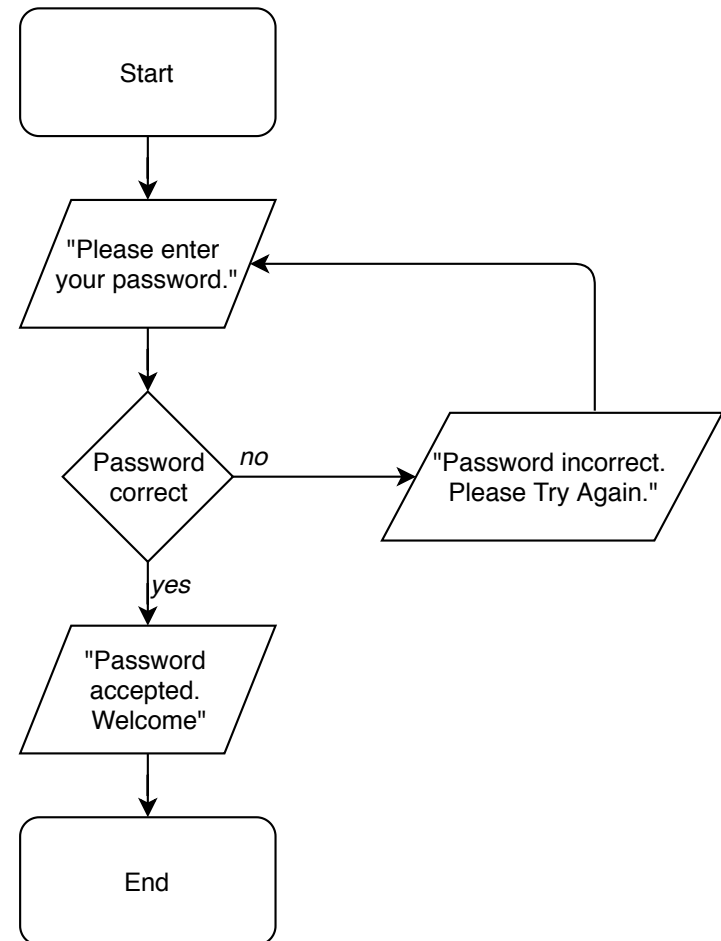


Login System Solution



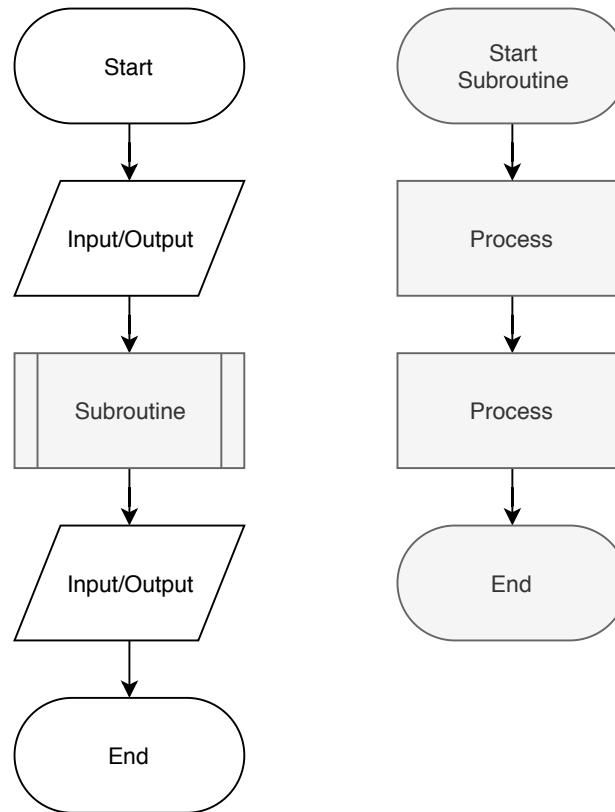
Activity: Login System in Python

Using the flowchart for the login system, write a python program that satisfies the brief.



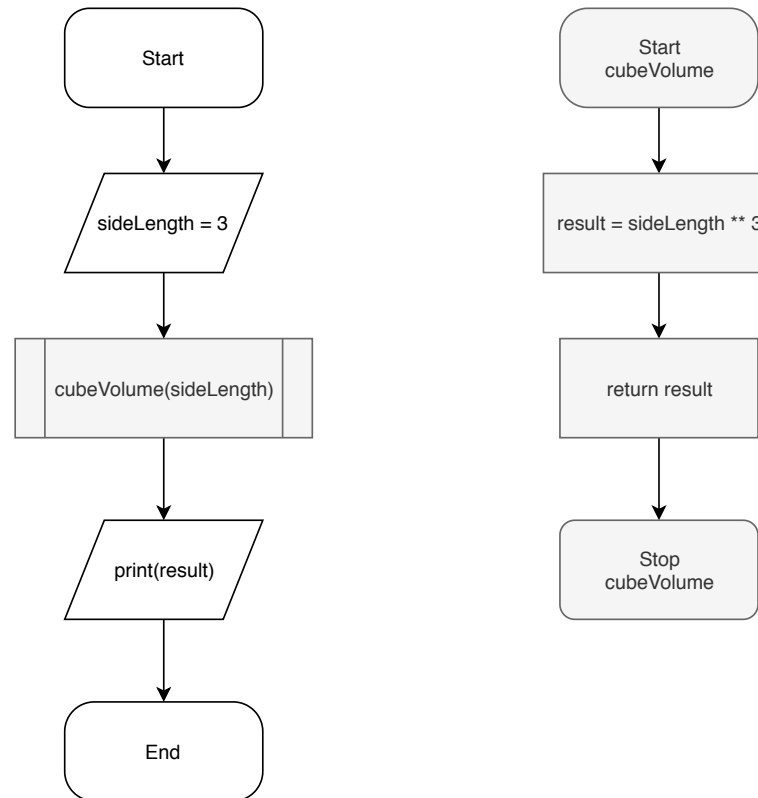
Subroutines

Subroutines are a sequence of instructions that perform a specific task.



Subroutines

For example we could have a subroutine that calculates the volume of a cube when given the length of one side:



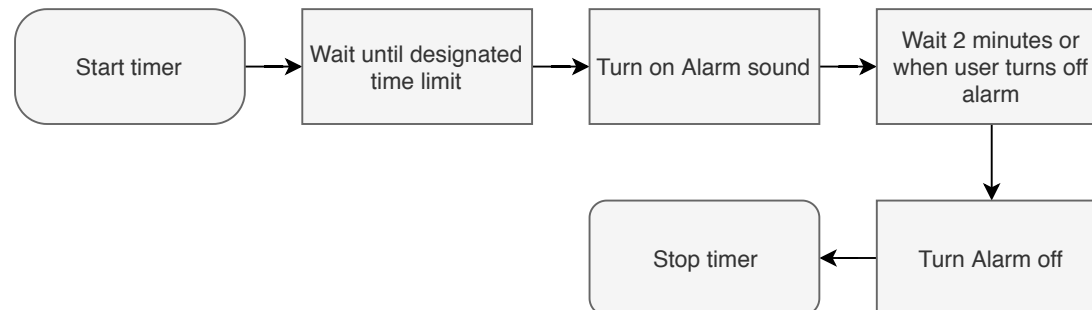
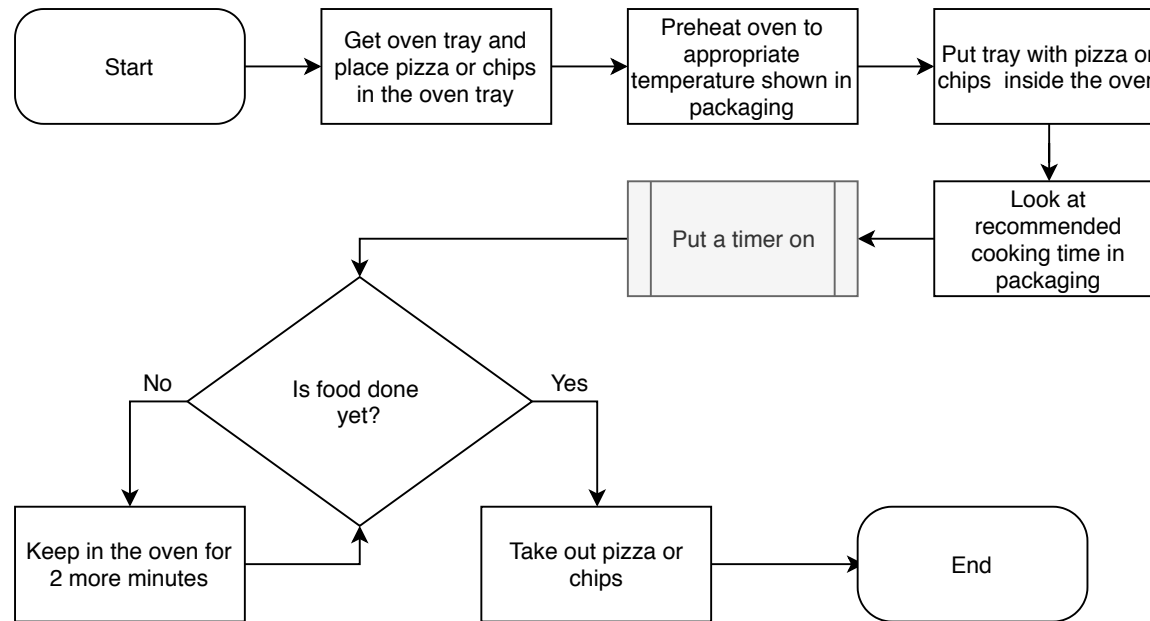
Activity: Pizza Flowchart

Create a flowchart on cooking pizza or chips in the oven. Try to make use of all of the flowchart components we have talked about:


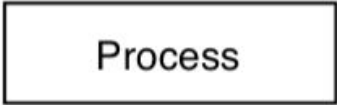
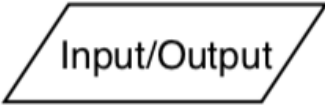

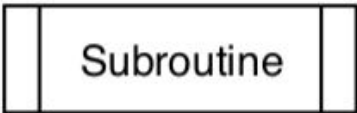

- Sequences
- Selection
- Iteration
- Subroutines

Extension Task: In addition to your flowchart try using the subroutine symbol, maybe a timer to tell you when to take the pizza/chips out.

Pizza Flowchart Solution



Flowchart Shape Recap

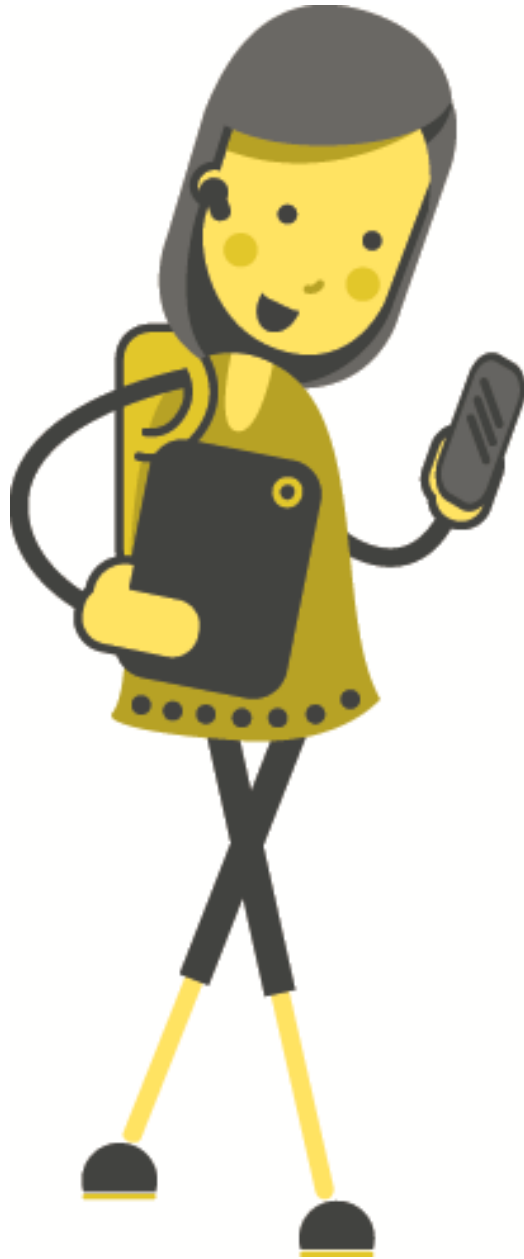
Name	Symbol	Usage
Start or Stop/End		Signifies the start or end of a sequence.
Process		An instruction.
Input/Output		Data received or sent by a computer.
Decision		A condition which is either true or false.
Subroutines		Calls a subroutine
Direction of Flow		Connects symbols.

Flowchart Shape Recap

Name	Symbol	Usage
Start or Stop/End		
Process		
Input/Output		
Decision		
Subroutines		
Direction of Flow		

Python Syntax

Action	Python code
Assign a variable	<pre>myVariable = 42 myOtherVariable = "hello"</pre>
Print something	<pre>print("This will be printed")</pre>
Getting input	<pre>age = input("How old are you?")</pre>
if / else	<pre>if age > 17: print("You are an adult") else: print("You are not an adult yet")</pre>
for loop	<pre>for i in range(0,10): print(i)</pre>
while loop	<pre>while(True): print("hello again")</pre>



Activity: Login System in Python

Simple Interest vs. Compound Interest

Most bank accounts will pay you **interest** on money you have deposited with them.

Imagine that you have £1000 in a bank account. Would you rather I gave you:

- 100% interest after 10 years.
- 10% interest every year for 10 years.



Calculating Compound Interest

Year	Base Amount	Interest (Base * Interest Rate)	Total
1	£1000	£1000*0.1 = £100	£1000+£100 = £1100
2	£1100	£1100*0.1 = £110	£1100+£110 = £1210
3			
4			
...
10			???

GCSE Compound Interest

GCSE MATHEMATICS - NUMERACY Specimen Assessment Materials 100

9. Carys decides to invest £380 in a savings account for 6 years. The account pays a rate of 2.54% AER.

Will Carys have sufficient money in her savings account to be able to buy a motor scooter costing £460 in 6 years' time? You must show all your working and give a reason for your answer.



[4]

First, Abstract Away The Unimportant Details

GCSE MATHEMATICS - NUMERACY Specimen Assessment Materials 100

9. Carys decides to invest £380 in a savings account for 6 years. The account pays a rate of 2.54% AER.

Will Carys have sufficient money in her savings account to be able to buy a motor scooter costing £460 in 6 years' time? You must show all your working and give a reason for your answer.



[4]

First, Abstract Away The Unimportant Details

GCSE MATHEMATICS - NUMERACY Specimen Assessment Materials 100

9. Carys decides to invest £380 in a savings account for 6 years. The account pays a rate of 2.54% AER.

Will Carys have sufficient money in her savings account to be able to buy a motor scooter costing £460 in 6 years' time? You must show all your working and give a reason for your answer.



[4]

startingAmount = £380

First, Abstract Away The Unimportant Details

GCSE MATHEMATICS - NUMERACY Specimen Assessment Materials 100

9. Carys decides to invest £380 in a savings account for 6 years. The account pays a rate of 2.54% AER.

Will Carys have sufficient money in her savings account to be able to buy a motor scooter costing £460 in 6 years' time? You must show all your working and give a reason for your answer.



[4]

startingAmount = £380

time = 6 years

First, Abstract Away The Unimportant Details

GCSE MATHEMATICS - NUMERACY Specimen Assessment Materials 100

9. Carys decides to invest £380 in a savings account for 6 years. The account pays a rate of 2.54% AER.

Will Carys have sufficient money in her savings account to be able to buy a motor scooter costing £460 in 6 years' time? You must show all your working and give a reason for your answer.



[4]

startingAmount = £380

time = 6 years

interestRate = 2.54% (Divide this by 100 to get the decimal 0.0254)

First, Abstract Away The Unimportant Details

GCSE MATHEMATICS - NUMERACY Specimen Assessment Materials 100

9. Carys decides to invest £380 in a savings account for 6 years. The account pays a rate of 2.54% AER.

Will Carys have sufficient money in her savings account to be able to buy a motor scooter costing £460 in 6 years' time? You must show all your working and give a reason for your answer.



[4]

startingAmount = £380

time = 6 years

interestRate = 2.54% (Divide this by 100 to get the decimal 0.0254)

costOfBike = £460

First, Abstract Away The Unimportant Details

£380

2.54%

6 years.

£460 in 6 years' time?

startingAmount = £380

time = 6 years

interestRate = 2.54% (Divide this by 100 to get the decimal 0.0254)

costOfBike = £460

The First Year

We have the initial £380. The interest gained which would be:

$$£380 \times 0.0254 = £9.652$$

Therefore after the **first** year we have: $£380 + £9.652 = £389.652$

The Second Year

After the first year we have: £389.652

Now we do the same for the second year, however the amount we have to start with has increased:

$$£389.652 + (£389.652 \times 0.0254) = £399.54916$$

The Final Year

We **repeat** this process until we have done it 6 times and we reach the final value of £441.72:

Starting amount: £380

1st Year: £389.652

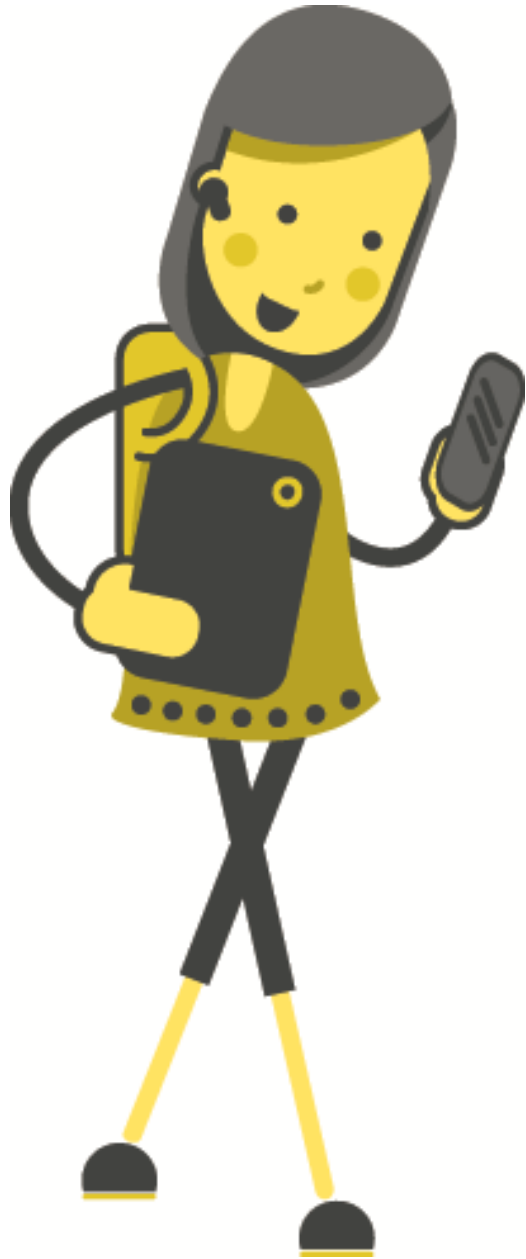
2nd Year: £399.549

3rd Year: £409.697

4th Year: £420.104

5th Year: £430.774

6th Year: £441.716



Activity: Compound Interest

Activity: Compound Interest in Python

Create a Python program which will allow a user to input an **initial value**, a **number of years** for saving, an **annual interest rate** (you'll need to decide whether to ask for a decimal or percentage value) and the **cost of an item to compare to** at the end.

The program will then need to calculate and output the final balance and compare it to the cost of the item and output if the balance is enough to purchase the item.

Is There An Easier Way?

Can we implement this without using a loop?

Is There An Easier Way?

Can we implement this without using a loop?

First we have an initial amount A , and in the first year we are adding to it this amount multiplied by the interest rate R . therefore after the first year the total T is given by:

Is There An Easier Way?

Can we implement this without using a loop?

First we have an initial amount A , and in the first year we are adding to it this amount multiplied by the interest rate R . therefore after the first year the total T is given by:

$$T = A + AR$$

which we rewrite as

$$T = A(1 + R)$$

Is There An Easier Way?

Can we implement this without using a loop?

First we have an initial amount A , and in the first year we are adding to it this amount multiplied by the interest rate R . therefore after the first year the total T is given by:

$$T = A + AR$$

which we rewrite as

$$T = A(1 + R)$$

Why?

Equation for Compound Interest

After the second year we just repeat the process of multiplying our amount by the ratio 1.0254

So every year we just multiply by another 1.0254 or another $(1 + R)$.

Equation for Compound Interest

So every year we just multiply by another 1.0254 or another $(1 + R)$.

Written out another way it looks like this:

First Year: $T = A(1 + R)$

Second Year: $T = A(1 + R)(1 + R)$

Third Year: $T = A(1 + R)(1 + R)(1 + R)$

Fourth Year: $T = A(1 + R)(1 + R)(1 + R)(1 + R)$

Fifth Year: $T = A(1 + R)(1 + R)(1 + R)(1 + R)(1 + R)$

Sixth Year: $T = A(1 + R)(1 + R)(1 + R)(1 + R)(1 + R)(1 + R)$

Equation for Compound Interest

So every year we just multiply by another 1.0254 or another $(1 + R)$.

First year: $T = A(1 + R)$

Second Year: $T = A(1 + R)(1 + R)$

Third Year: $T = A(1 + R)(1 + R)(1 + R)$

Fourth Year: $T = A(1 + R)(1 + R)(1 + R)(1 + R)$

Fifth Year: $T = A(1 + R)(1 + R)(1 + R)(1 + R)(1 + R)$

Sixth Year: $T = A(1 + R)(1 + R)(1 + R)(1 + R)(1 + R)(1 + R)$

Simplifying:

$$T = A(1 + R)(1 + R)(1 + R)(1 + R)(1 + R)(1 + R) = A(1 + R)^6$$

So with n amount of years:

$$T = A(1 + R)^n$$



Sorting Algorithms

Bubble Sort

Sometimes we have a list of numbers that we would like to be sorted. Algorithms that can do this are called **sorting algorithms** and an example of a sorting algorithm is Bubble sort.

In Bubble sort, the larger numbers 'bubble' up to the top of the list.



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

6 5 3 1 8 7 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 6 3 1 8 7 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 6 3 1 8 7 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 6 1 8 7 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 6 1 8 7 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 1 6 8 7 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 1 6 8 7 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 1 6 8 7 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 1 6 7 8 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 1 6 7 8 2 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 1 6 7 2 8 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 1 6 7 2 8 4



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. Repeat until all numbers in correct order.

5 3 1 6 7 2 4 8



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed.

Repeat until all numbers in correct order.

5 3 1 6 7 2 4 8



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed.

Repeat until all numbers in correct order.

5 3 1 6 7 2 4 8

Repeat process until all numbers in correct order.



Bubble Sort

How to Bubble Sort:

From left to right, compare two numbers, swap if needed. **Repeat until all numbers in correct order.**

1 2 3 4 5 6 7 8

Repeat process until all numbers in correct order.



Activity: Bubble Sort

Using the following numbers '7 4 1 5 8 3 6 2' perform a bubble sort to sort the numbers in a numerical order from smallest to largest. Afterwards write down the steps you performed to complete this task in your workbooks.

Again the list of numbers to sort are:

7 4 1 5 8 3 6 2

Merge Sort

Merge Sort is a "divide and conquer" sorting algorithm. We repeatedly **divide** the list of numbers into smaller and smaller lists until each list contains only one item. We then **merge** these smaller lists together, making sure that the newly merged list is sorted. We keep on merging smaller lists into bigger lists until the whole original list is sorted.

Merge sort is much faster at sorting than Bubble sort. It is important to remember that Bubble sort is OK for small lists but big lists should be sorted with a faster sorting algorithm, like Merge sort.

Merge Sort

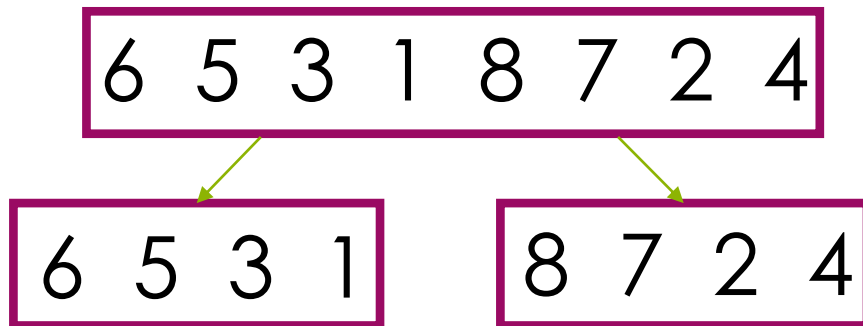
Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. Then we merge the lists together making sure the items are in the correct order.

6 5 3 1 8 7 2 4



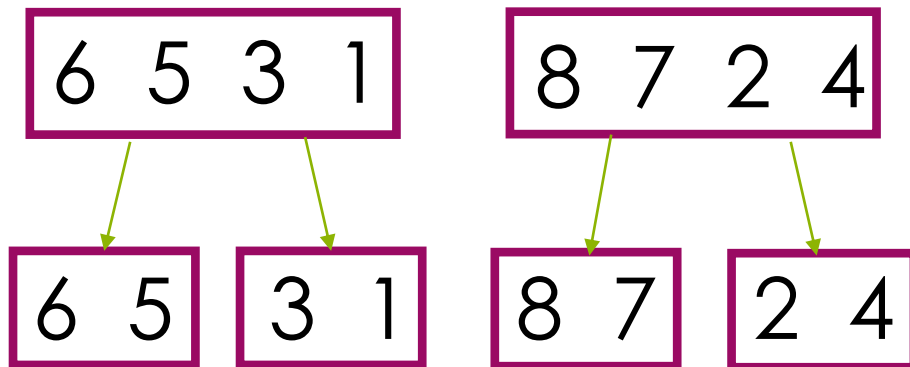
Merge Sort

Unlike Bubble Sort, we repeatedly **split the lists into smaller and smaller lists until there is only one item in each list**. Then we merge the lists together making sure the items are in the correct order.



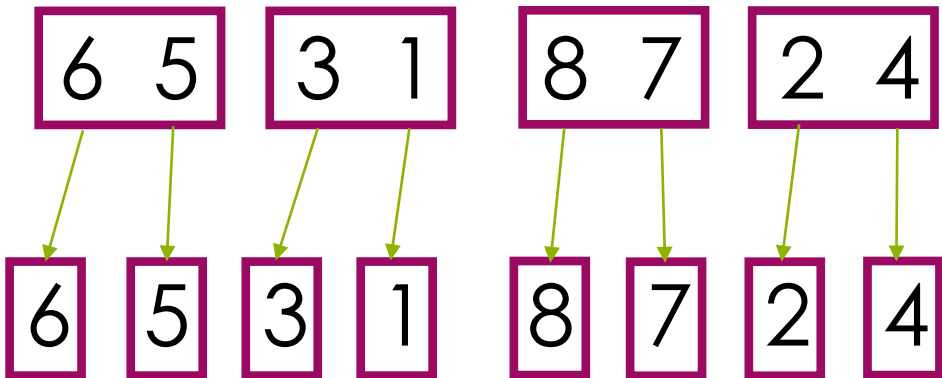
Merge Sort

Unlike Bubble Sort, we repeatedly **split the lists into smaller and smaller lists until there is only one item in each list**. Then we merge the lists together making sure the items are in the correct order.



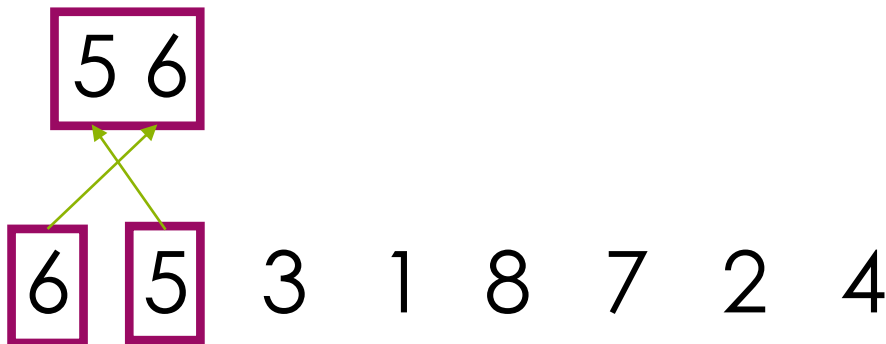
Merge Sort

Unlike Bubble Sort, we repeatedly **split the lists into smaller and smaller lists until there is only one item in each list**. Then we merge the lists together making sure the items are in the correct order.



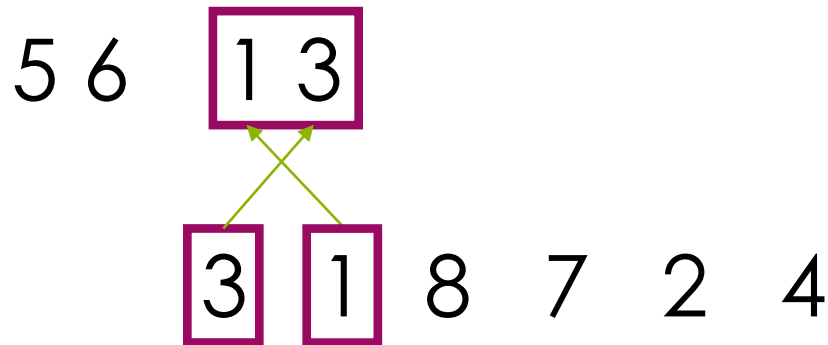
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



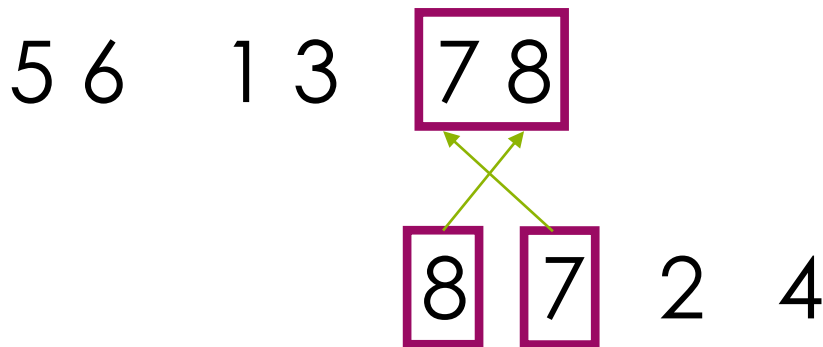
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



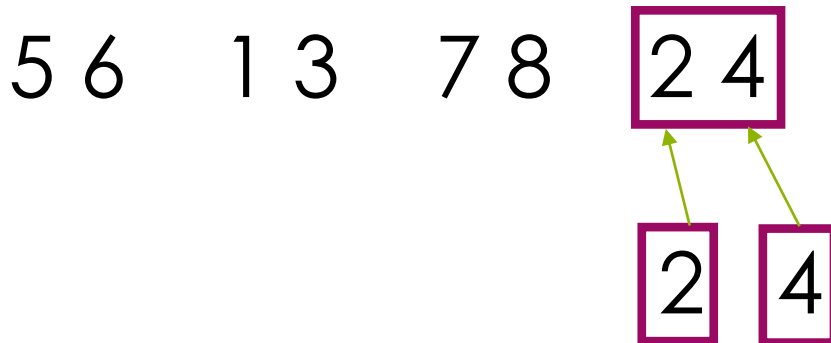
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. Then we merge the lists together making sure the items are in the correct order.



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



Merge Sort

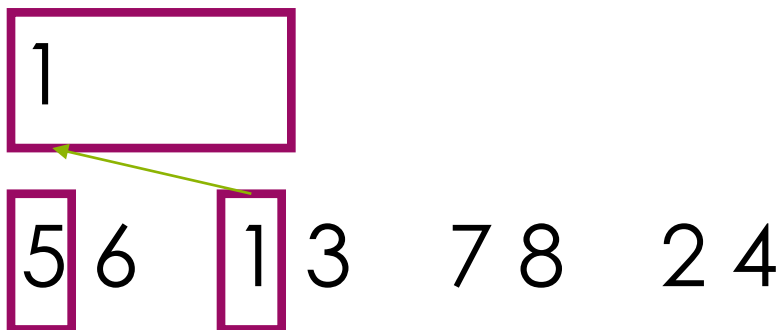
Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**

5 6 1 3 7 8 2 4



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**

1

5 6

3

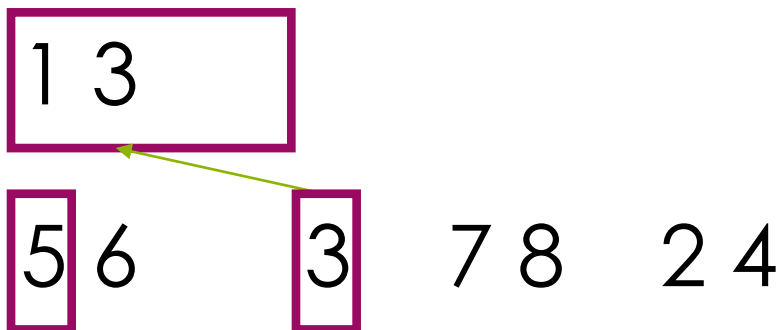
7 8

2 4



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**

1 3

5 6

7 8 2 4



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**

1 3 5

5 6

7 8 2 4



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**

1 3 5

6

7 8 2 4



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**

1 3 5 6

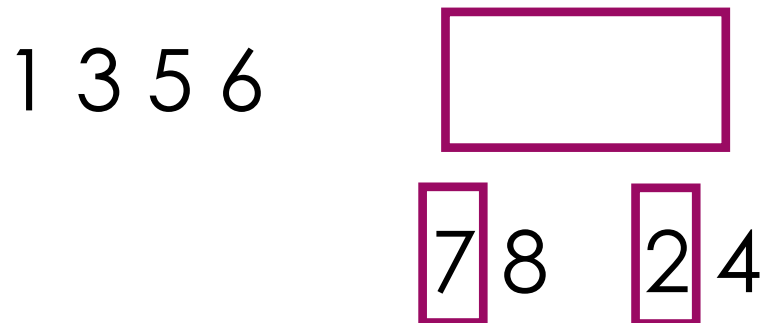
6

7 8 2 4



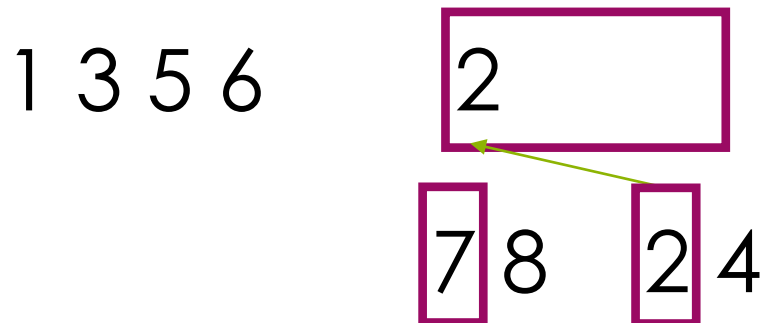
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



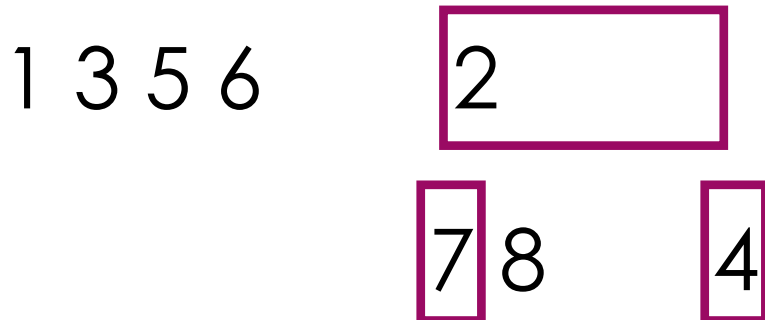
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



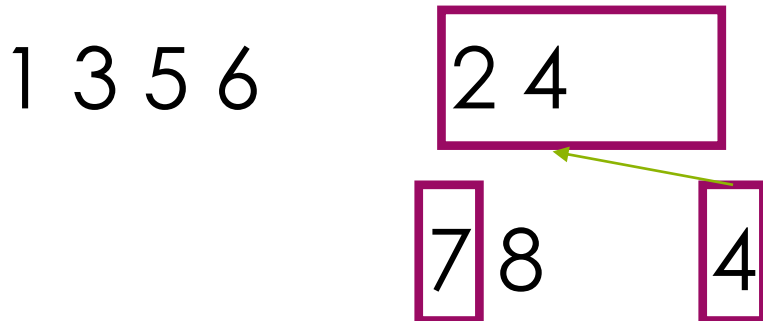
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



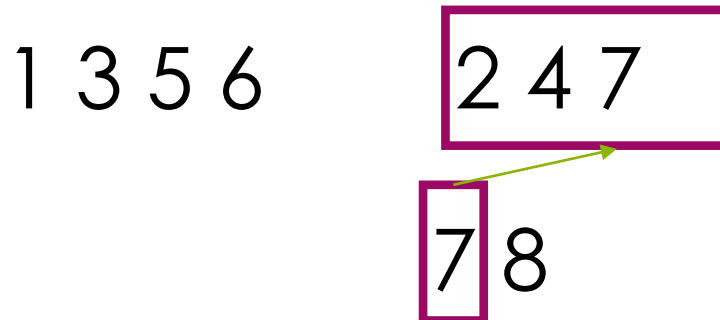
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**

1 3 5 6

2 4 7 8

8



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



1 3 5 6

2 4 7 8



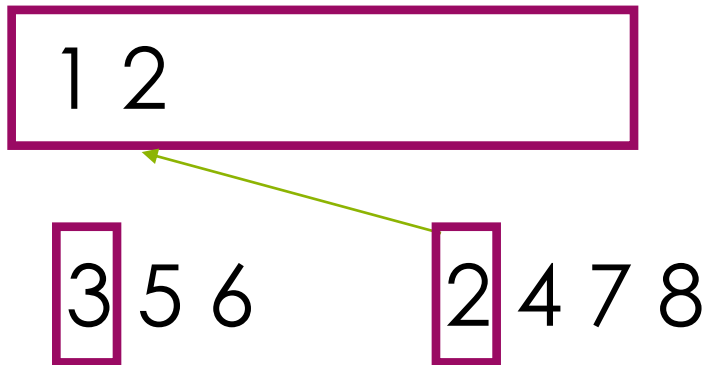
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



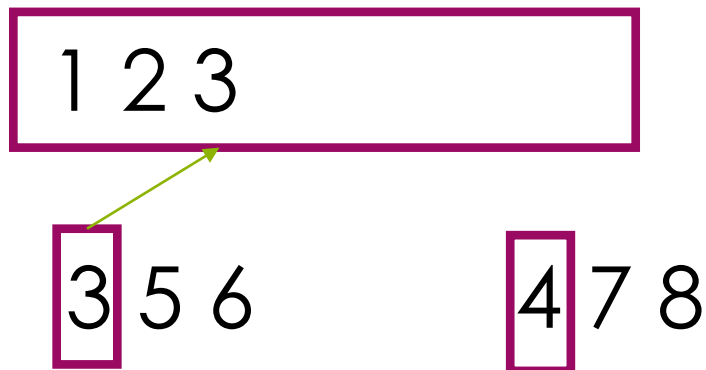
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



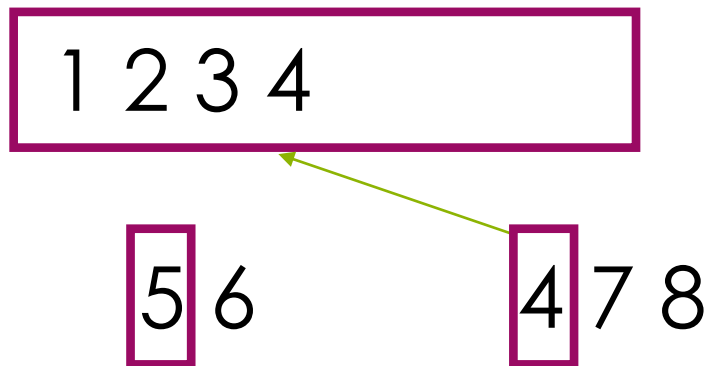
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



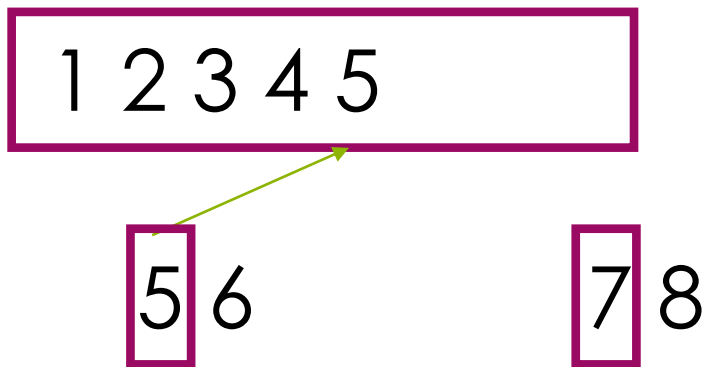
Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. **Then we merge the lists together making sure the items are in the correct order.**

And so on...



Merge Sort

Unlike Bubble Sort, we repeatedly split the lists into smaller and smaller lists until there is only one item in each list. Then we merge the lists together making sure the items are in the correct order.

1 2 3 4 5 6 7 8



Activity: Merge Sort

Using the following numbers, '4 8 2 6 7 3 5 1' perform a merge sort to sort the numbers in a numerical order from smallest to largest. Afterwards write down the steps you performed to complete this task in your workbooks.

The list of numbers to sort is:

4, 8, 2, 6, 7, 3, 5, 1



Search Algorithms

Linear Search

A linear search is a simple search algorithm where a list is searched until the required value is found.

For example if the value we are looking for is 4.

6 5 3 1 8 7 2 4

Linear Search

A linear search is a simple search process where a list is searched until the required value is found.

For example if the required value is 4.

6 5 3 1 8 7 2 4

Linear Search

A linear search is a simple search process where a list is searched until the required value is found.

For example if the required value is 4.

6 5 3 1 8 7 2 4

Linear Search

A linear search is a simple search process where a list is searched until the required value is found.

For example if the required value is 4.

6 5 3 1 8 7 2 4

Linear Search

A linear search is a simple search process where a list is searched until the required value is found.

For example if the required value is 4.

6 5 3 1 8 7 2 4

Linear Search

A linear search is a simple search process where a list is searched until the required value is found.

For example if the required value is 4.

6 5 3 1 8 **7** 2 4

Linear Search

A linear search is a simple search process where a list is searched until the required value is found.

For example if the required value is 4.

6 5 3 1 8 7 **2** 4

Linear Search

A linear search is a simple search process where a list is searched until the required value is found.

For example if the required value is 4.

6 5 3 1 8 7 2 **4**

Linear Search

A linear search is a simple search process where a list is searched until the required value is found.

For example if the required value is 4.

6 5 3 1 8 7 2 **4**

Activity: Linear Search

Using the following numbers shown below, perform a linear search to locate the value '**42**'. Afterwards write down the steps you performed to complete this task in your workbooks.

The list of numbers to search through is:

7, 14, 21, 28, 35, 42, 49

Binary Search

The binary search algorithm (also known as a half interval search) works like this:

1. The middle value in a **sorted** list is inspected to see if it matches the search value.
2. If the middle value is greater than the search value, the upper half of the list is discarded. If it is less than the search value, the lower half is discarded.
3. This process is repeated, with the list halving in size each time until the search value is found.

For example if the value we are searching for is **4**.

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Binary Search

The binary search algorithm (also known as a half interval search) works like this:

1. The middle value in a **sorted** list is inspected to see if it matches the search value.
2. If the middle value is greater than the search value, the upper half of the list is discarded. If it is less than the search value, the lower half is discarded.
3. This process is repeated, with the list halving in size each time until the search value is found.

For example if the value we are searching for is **4**.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----

Binary Search

The binary search algorithm (also known as a half interval search) works like this:

1. The middle value in a **sorted** list is inspected to see if it matches the search value.
2. If the middle value is greater than the search value, the upper half of the list is discarded. If it is less than the search value, the lower half is discarded.
3. This process is repeated, with the list halving in size each time until the search value is found.

For example if the value we are searching for is **4**.



1 2 3 4 5 6 7

Binary Search

The binary search algorithm (also known as a half interval search) works like this:

1. The middle value in a **sorted** list is inspected to see if it matches the search value.
2. If the middle value is greater than the search value, the upper half of the list is discarded. If it is less than the search value, the lower half is discarded.
3. This process is repeated, with the list halving in size each time until the search value is found.

For example if the value we are searching for is **4**.

4

Activity: Binary Search

Using the following numbers shown below, perform a linear search to locate the value '**35**'. Afterwards write down the steps you performed to complete this task in your workbooks.

The list of numbers to search through are :

7, 14, 21, 28, 35, 42,
49, 56, 63, 70, 77

(remember with a Binary search algorithm the list of values must be sorted first before the search is performed)

Extension Activity: Square Root

Newton-Raphson Method for a Square root algorithm:

$$a_n = \frac{1}{2} \left(a_{n-1} + \frac{x}{a_{n-1}} \right)$$

This finds the square root of x .

If $x = 25$, we choose $a_0 = 3$

$$a_1 = \frac{1}{2} \left(3 + \frac{25}{3} \right)$$

$$a_1 = 5.666666$$

If we repeat these steps again and again, we then the answer will approach the correct square root of 25.

$$a_2 = \frac{1}{2} \left(5.666666 + \frac{25}{5.666666} \right)$$