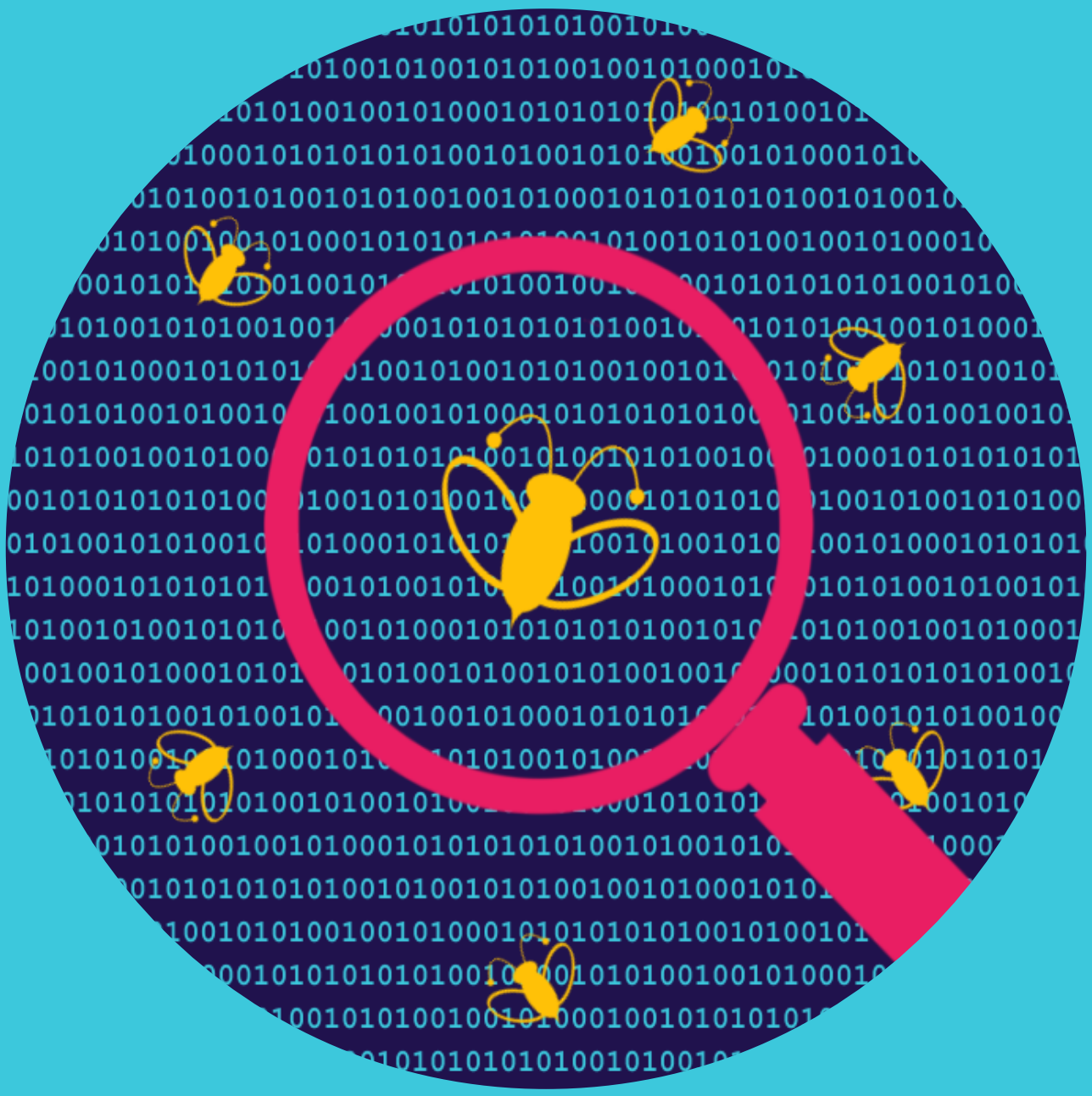
Researchers in Schools



How to code Bug-Free Programs!

Key Stage 4 Programme

Pupil Name

Coursebook Designed by Dr. David Sherratt

Contents

Uni Pathways launch event 1
Introduction to Uni Pathways 1
Pupils' experience of Uni Pathways 2
Virtual campus tours 2
Meet university students! 3
Reflection 3
Timetable and Assignment Submission 5
Timetable – Tutorials 5
Timetable – Homework Assignments 5
Assignment Submission – Lateness and Plagiarism 5
Course Rationale 6
Uni Pathways Mark Scheme 2020 – maths-based assignments 7
Baseline Assignment (essay based): Pupil Feedback Report 9
Final Assignment (essay based): Pupil Feedback Report 10
Subject Vocabulary 11
Tutorial 1 – The relationship of Maths and Computing 12
What is the Purpose of Tutorial 1? 12
Tutorial 2 – Logic and Proofs 17
What is the Purpose of Tutorial 2? 17
Tutorial 3 – Operational Semantics Part I 22
What is the Purpose of Tutorial 3? This tutorial will investigate using logic and proofs to explain computation and execution of operations in a programming language. By the end of the tutorial you will be able to: 22
Tutorial 3 – Homework 27
Tutorial 4 – Operational Semantic Part II 28
What is the Purpose of Tutorial 4? 28
Tutorial 4 – Homework 32
Tutorial 5 – Proofs as Programs 33
What is the Purpose of Tutorial 5? 33
Tutorial 5 – Final Assignment 37
Tutorial 6 – Feedback tutorial 39
What is the Purpose of Tutorial 6? 39
Final assignment feedback from your Uni Pathways Teacher 39
Tutorial 7 – Final tutorial 41
Appendix 1 – Referencing correctly 43
How do I reference? 44
Notes 45

Uni Pathways launch event

Welcome to Uni Pathways! We are delighted to be able to launch the Uni Pathways programme with you. We hope that you are ready to embark on your Uni Pathways journey and that you enjoy the video.
You will be asked to pause the video at times to complete some work in this workbook, so make sure you have a pen / pencil to hand when you start the video.

By the end of the video, you will have

- Learnt about what studying at university means
- Learnt about some of the skills that you will develop during Uni Pathways
- Heard from pupils who have participated in Uni Pathways or The Scholars Programme (which is the same programme!)
- Heard from current university students talking about what life is like at university

If you are in school your teacher will play the video. If you are at home and logged in to a session with your teacher, your teacher will play the video and show it to you. If you are participating independently your teacher will email you the link to the video or the video file. There are opportunities for you to answer some questions, and you will be told when to pause the video to answer them.

Introduction to Uni Pathways

1. Write down what you think a supra-activity is

_____
_____
_____

2. Independent learning is

_____
_____
_____

3. Write down in your own words what resilience means, and come up with a different example to the one mentioned in the PowerPoint.

_____
_____
_____
_____

4. Why do you think analysis and evaluation skills are useful in your academic career?

_____
_____
_____
_____
_____

5. Why do you think analysis and evaluation skills are useful in life in general?

_____
_____
_____
_____
_____

6. What are the different types of learning that university students do?

Pupils' experience of Uni Pathways

You will now watch some pupils talk about their experience of learning a PhD topic and producing a final assignment. Some may refer to The Scholars Programme rather than Uni Pathways. As you heard in the introduction PowerPoint, The Scholars Programme is a very similar programme to Uni Pathways. Listen carefully and then respond to the questions below. Be prepared to share some of your responses with your peers.

1. How did the pupils describe talking about writing a long essay for their final assignment?

2. Write down something that one of the pupils mentioned was particularly interesting in their course.

3. What were some challenges that the pupils met?

Virtual campus tours

You will now go on a virtual tour of some universities! Once you have seen some of the university campuses, respond to the questions below.
Note down something that you liked out of any of the campus tours you saw. It may be a particular building, space, city etc.

1. Were there any similarities between the different university campuses? If so, what are the similarities?

2. If you had to pick one of those universities to go and visit in person, which one would it be and why?

3. Note down some of the societies that you could join at different universities

Meet university students!

You are about to virtually meet or hear from some current university students.
If you are virtually meeting them, think about what questions you would like to ask the students, share your questions with a partner and note those questions down in the space below.

If you are not meeting them but watching some videos that they have made, use the space below to note down what else you would like to find out about university. You can then ask your teacher during Uni Pathways tutorials!

Reflection

Congratulations on completing the launch of the Uni Pathways programme. Before you go, take some time to reflect on what you have learnt by answering the following questions:

1. Are there any aspects of university style learning that you would look forward to? If so, what are they?

2. What challenges do you think students face when learning at university? How would you try to overcome these challenges?

3. What skills do you hope to develop during your Uni Pathways course?

4. What part of Uni Pathways seems the most challenging for you?

5. What are you most looking forward to about Uni Pathways?

Timetable and Assignment Submission

Timetable – Tutorials

Tutorial	Date	Time	Location
1			
2			
3			
4			
5			
6 (Feedback)			
7 (Feedback)			

Timetable – Homework Assignments

Homework Assignment	Description	Due Date
Tutorial 1		
Tutorial 2		
Tutorial 3		
Tutorial 4		
Tutorial 5		

Assignment Submission – Lateness and Plagiarism

Lateness	
Submission after midnight on 9th August	10 marks deducted
Plagiarism	
Some plagiarism	10 marks deducted
Moderate plagiarism	20 marks deducted
Extreme plagiarism	Automatic fail

Course Rationale

Think of your favourite computer game, and pretend the developers have announced a new, amazing, revolutionising game feature. You are excited and have been saving money to buy this extra feature. It's finally released, and you immediately buy it without hesitation. You start to play, and you realise the game is not quite what you expected. It's not as good as you thought, it doesn't do what was promised, and you feel you were lied to and cheated out of some money.

When we are writing complicated computer programs, like a game, sometimes we think we have written a program that do A, B, and C. However, when it comes to running the program, it does X, Y, and Z. We've discovered we didn't build what it was we wanted. Now we have wasted time developing and publishing a program that does the wrong thing. If this happened in a big tech company (like Google, Apple, Microsoft, etc) then a lot of money and resources would have been wasted.

In this course, we will investigate the semantics of programs. We will see how we can explicitly formalise the semantics, using logic and reasoning, so that the meaning of a program can be studied like how we study mathematics. This lets us define more precise and accurate descriptions of our programs and reduces the chances of mistakes/errors/bugs/glitches.

We use maths to write programs, to avoid writing mistakes such as coding the wrong feature into a game. We will explore a perspective of programming that strives for verification rather than implementation.

Uni Pathways Mark Scheme 2020 – maths-based assignments

	Subject Knowledge	Critical Thinking	Written Communication
1st	The work shows a depth of knowledge and understanding of key concepts and mathematical methods, drawing on sources of evidence where appropriate. Knowledge is used to build and support highly effective mathematical arguments and explanations.	Analyses the available data/evidence and identifies all the relevant mathematical information. Identifies and critically evaluates available mathematical approaches and evidence, deciding on their credibility, strength and relative significance. Makes connections between different pieces of information and puts them together to form accurate solutions and verify answers and assumptions.	Mathematical arguments are complete, and the reasoning behind each step has been clearly communicated to the reader. The work has a coherent flow and is well structured. The writing style is appropriate; mathematical language and key mathematical terms are used accurately and effectively to support the arguments and explanations made. There are no, or very few, errors in spelling or grammar. Consistent referencing, appropriate paragraphing and use of correctly labelled tables and figures; matching the style taught in the course.
2:1	The work shows an understanding of key concepts and mathematical methods, drawing on sources of evidence where appropriate. Knowledge is used to build and support effective mathematical arguments and explanations.	Analyses the available data/evidence and identifies all the relevant mathematical information. Engages with multiple approaches and critically evaluates their utility, forming judgements and using reasoning and evidence to inform problem solving strategy. Shows some understanding of the relative value of evidence and data.	Mathematical arguments are clearly made, with few gaps in logical reasoning. The work is well-structured. The writing style is appropriate; mathematical language and key terms are used correctly. There are few errors in spelling or grammar. Mostly consistent referencing and use of tables and figures; matching the style taught in the course.
2:2	The work shows an understanding of key concepts and mathematical methods, with no major misconceptions. Beginning to apply this knowledge to build and support effective mathematical arguments and explanations.	Identifies and uses basic mathematical evidence and reasoning. Evaluates the quality of mathematical reasoning and evidence, forming judgements. Not yet showing understanding of the relative value of evidence and reasoning.	Mathematical arguments have been constructed, but there is little explanation from step to step. The work has some structure. The writing style can sometimes be informal; occasionally key terms are not used when it would be appropriate to do so.

			There are some errors in grammar and spelling, but these do not get in the way of communicating the content. Referencing has some consistency; matching the style taught in the course. Limited use of tables and figures.
3rd	Shows a developing understanding of key concepts and mathematical methods, with some misconceptions. Not yet applying this knowledge to build and support mathematical arguments and explanations.	Beginning to analysis mathematical evidence and reasoning. Describes or attempts to solve the problem without any/or little evaluation of whether the approach was reasonable or accurate.	Mathematical arguments to solve problems are poorly sequenced, with little or no explanation. The grammar, spelling, style, and structure of the work need improving in order to communicate ideas to the reader. Key terms are not always used correctly. Limited, or no use of referencing, tables and figures.

Baseline Assignment (essay based): Pupil Feedback Report

Name of Pupil	
Name of School	
Name of RIS teacher	
Title of Assignment	

How your assignment is graded:

Grade	Marks	What this means
1st	70+	Performing to an excellent standard at A-level
2:1	60-69	Performing to a good standard at A-level
2:2	50-59	Performing to an excellent standard at GCSE
3rd	40-49	Performing to a good standard at GCSE
Working towards a pass	0-39	Performing below a good standard at GCSE
Did not submit	DNS	No assignment received by The Brilliant Club

Lateness	
Any lateness	10 marks deducted
Plagiarism	
Some plagiarism	10 marks deducted
Moderate plagiarism	20 marks deducted
Extreme plagiarism	Automatic fail

Marks

OVERALL MARK / 100		FINAL MARK / 100 (including any deductions)	
DEDUCTED MARKS		FINAL GRADE	

If marks have been deducted (e.g. late submission, plagiarism) the teacher should give an explanation in this section:

Mark Breakdown and Feedback

Subject knowledge	
	mark
Critical thinking	
	mark
Written communication	
	mark

Final Assignment (essay based): Pupil Feedback Report

Name of Pupil	
Name of School	
Name of RIS teacher	
Title of Assignment	

How your assignment is graded:

Grade	Marks	What this means
1st	70+	Performing to an excellent standard at A-level
2:1	60-69	Performing to a good standard at A-level
2:2	50-59	Performing to an excellent standard at GCSE
3rd	40-49	Performing to a good standard at GCSE
Working towards a pass	0-39	Performing below a good standard at GCSE
Did not submit	DNS	No assignment received by The Brilliant Club

Lateness	
Any lateness	10 marks deducted
Plagiarism	
Some plagiarism	10 marks deducted
Moderate plagiarism	20 marks deducted
Extreme plagiarism	Automatic fail

Marks

OVERALL MARK / 100		FINAL MARK / 100 (including any deductions)	
DEDUCTED MARKS		FINAL GRADE	

If marks have been deducted (e.g. late submission, plagiarism) the teacher should give an explanation in this section:

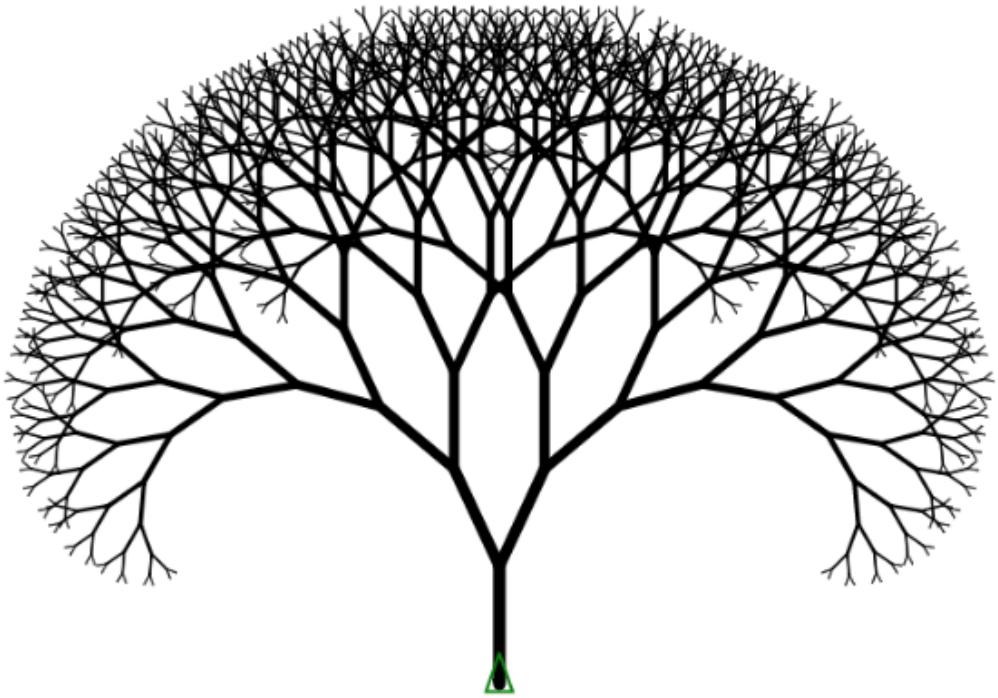
Mark Breakdown and Feedback

Subject knowledge	
	mark
Critical thinking	
	mark
Written communication	
	mark

Subject Vocabulary

Word	Definition	In a sentence
Abstract	Something that only exists as an idea but does not physically exist.	
Denote	To be the name, mark, or sign of something.	A fever often denotes an infection. π denotes 3.14159…
Derivation	The action of obtaining something from a source.	
Function	You apply a function on some input to obtain some output.	The function "+" can be applied on 2 and 3 to get 5
Infer	The process of finding new information based on already proven evidence.	
Logic	A method of thinking. The way someone thinks.	
Operation	The action of applying functions.	Performing the operation "2 + 3" results in 5
Proof	Evidence confirming the truth of something.	
Semantics	In Computer Science, the *meaning* of computer programs.	
Software Industry	Activity concerned with writing, maintaining, and publishing software.	
Sound	Based on valid reason or good judgement.	That is sound advice for healthy living.
Syntax	In Computer Science, the *code* of computer programs.	
Verification	The process of confirming the accuracy or truth of something.	

Tutorial 1 – The relationship of Maths and Computing



What is the Purpose of Tutorial 1?

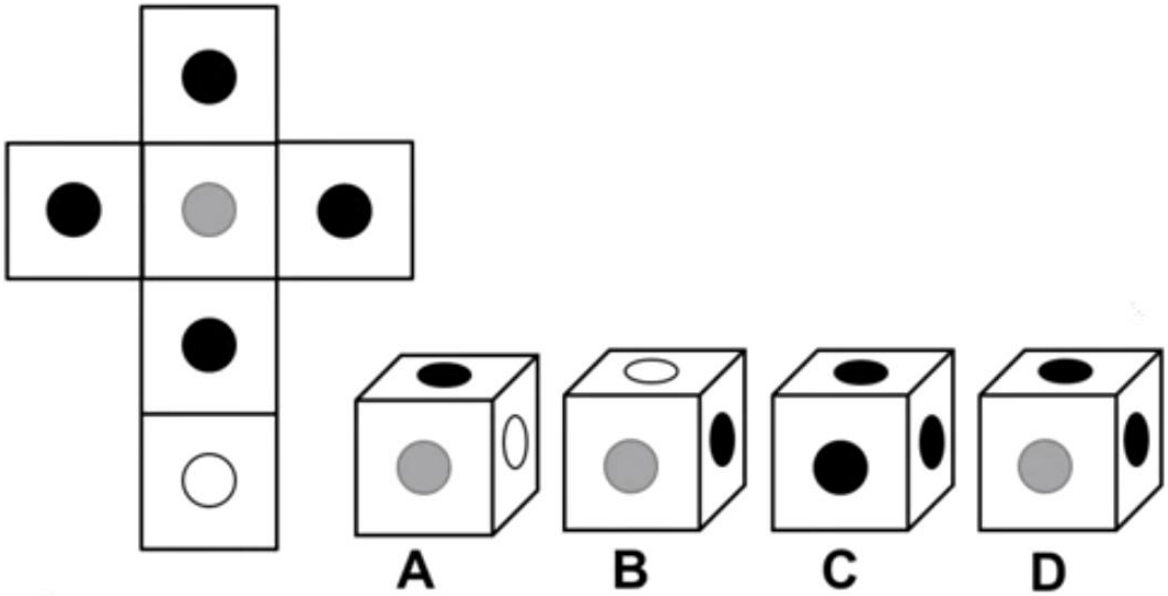
This tutorial will discuss the connection between mathematics and computing.

By the end of the tutorial you will be able to:

- identify what maths is used in computing
- define syntax and semantics

Activity 1: Draw a "*tasty red idea*".

Activity 2: Which cube can be made from this net?



For this question you need to imagine you are building the cube by folding the net. The cube net has shapes on, so when folding the creases to make the cube, you need to work out where each shape will be positioned from the options available.

Answer: _____

Question: Did we just do some maths?

Answer: YES! Logic and Reasoning

Activity 3: Write out 2 English sentences, one which has correct syntax and poor semantics and one with incorrect syntax but meaningful semantics.

Example:

Good Syntax, Poor Semantics	Bad Syntax, Meaningful Semantics
Orange imaginations eat too much	*Beach I go swim happy*

Answer:

Good Syntax, Poor Semantics	Bad Syntax, Meaningful Semantics

Activity 4: Look at the following code for some computer program, can you figure out what it is doing?

```
fruits = ["apple", "banana", "cherry"]

for x in fruits:
  print(x)
```

What does this program do?: _____

_____

What were you given? Answer: _____

What were you working out? Answer: _____

Activity 5: Draw a flowchart that expresses the semantics for the code in Activity 4. You can google "*Flowchart Loops*" and look at images for guidance.

Baseline Assignment

The baseline assignment is to help stretch and challenge your academic skills and subject knowledge. It will also advise the instructor structure this course content to effectively support you. The baseline assignment will involve a *written report* and a *problem sheet*, and you will need to do both.

Success Criteria

- Reference to suitable sources and clear explanations of research findings
- Justification of opinions, using appropriate vocabulary
- Work is neatly presented; spelling and grammar is accurate. Use a formal style of writing.

Written report

Write an article (about 250 words) that answers the question:

How are mathematics and computing similar?

This question is open and has no right or wrong answer, but in your answer, you will need to *justify* your opinions. Justification could involve giving examples or facts. A good justification will have a reference to a resource so that someone reading your article can see what you have read. For example, see below.

> I think that mathematics is important for computers, because the first mechanical computer was designed by the mathematician Charles Babbage [1]. The original purpose of a computer is to perform calculations and replaced human calculators by the end of the 19th century [2]. This means that mathematics was the reason computers were invented in the first place, and the original purpose of the computer was to perform calculations. The operations of a computer must therefore be linked with mathematics, specifically logic and reasoning. A computer function can correspond to a mathematical function.
>
> 92 words
>
> [1] http://www.bbc.co.uk/history/historic_figures/babbage_charles.shtml
> [2] https://www.computerhope.com/issues/ch000984.htm

The list below contains a few examples of what you can research about to help you answer the question. You do not have to research all these, nor are you limited to only researching these topics.

- Hilbert's Program
- Pilot ACE Machine
- Cryptography/Encryption
- Logic gates
- Binary
- Algorithm complexity
- Artificial Intelligence (AI)

Also don't forget to **PEE**

- Make your **P**oint
- Give **E**vidence
- **E**xplain your opinion

Problem Sheet

Below are 3 samples of python code. You can go to https://repl.it/ and create a new repl, choosing the language python. After doing this, you can type, under main.py, the code below. After clicking run, you will see an output on the screen.

Each code sample given to you is syntax, but what is the semantics? For each sample, write down what is happening, i.e. the semantics. You can explain this in plain English, or even better you can write the semantics as a flowchart (see Activity 5).

Don't be afraid to change the code a little to see how that affects the program. This will help you understand the semantics in even more detail.

Code Sample #1

```
num1 = 1.5
num2 = 6.3

sum = num1 + num2

print(sum)
```

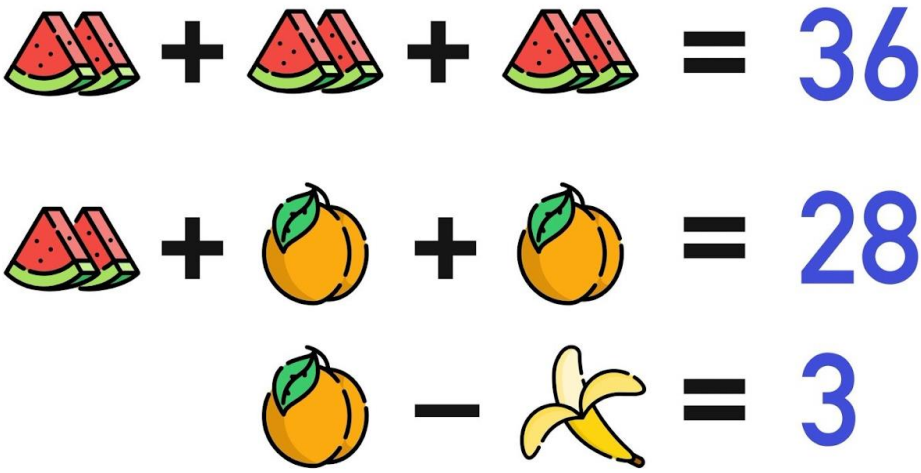
Code Sample #2

```
num1 = 10
num2 = 14
num3 = 12

if (num1 >= num2) and (num1 >= num3):
    largest = num1
elif (num2 >= num1) and (num2 >= num3):
    largest = num2
else:
    largest = num3

print(largest)
```

Code Sample #3

```
def fact(n):
  if n == 0:
    return 1
  else:
    return n*(fact(n-1))

print(fact(5))
```

Tutorial 2 – Logic and Proofs



What is the Purpose of Tutorial 2?

This tutorial will investigate the idea of logic and constructing proofs.

By the end of the tutorial you will be able to:

- apply inference rules to construct a proof
- identify errors in reasoning and logic
- recognize the importance and motivation of logic

Activity 1: Can you solve the puzzle?

= _____

= _____

= _____

Activity 2: Use the following inference rules to derive Kim is an architect.

$$\frac{\text{Kim}}{\text{Is good at maths}} \qquad \frac{\text{Kim}}{\text{Is good at art}} \qquad \frac{\text{Kim}}{\text{Is good at programming}}$$

$$\frac{\text{good at art OR music}}{\text{creative}} \qquad \frac{\text{good at science OR programming}}{\text{clever}}$$

$$\frac{\text{creative AND genius}}{\text{Architect}} \qquad \frac{\text{good at maths AND clever}}{\text{genius}}$$

Question: What's similar between Activity 1 and Activity 2?

Activity 3: Which of the following rules are sound and which are unsound? If the rule is unsound, explain why.

Inference Rule	Sound or Unsound?
$\dfrac{x = y}{3x = 3y}$	
$\dfrac{x + y = z}{x = z + y}$	
$\dfrac{x = \text{TRUE AND } y = \text{TRUE}}{x = \text{TRUE}}$	
$\dfrac{x = \text{TRUE}}{x = \text{TRUE AND } y = \text{TRUE}}$	
$\dfrac{x = \text{FALSE OR } y = \text{TRUE}}{x = \text{TRUE IMPLIES } y = \text{TRUE}}$	

Activity 4: How can we interpret the following inference rule?

$$\frac{A \rightarrow B \quad \text{AND} \quad A}{B}$$

Activity 5: We use logic and derivations to prove things. Discuss in pairs what would we want to prove for the following topics.

Topic	Something to prove
Geography	The world is round/flat.
Biology	
Physics	
Chemistry	
Maths	
English	
History	
Spanish/Languages	
Business	
Religious Education	
Law	

Tutorial 2 – Homework

Observe the inference rules below.

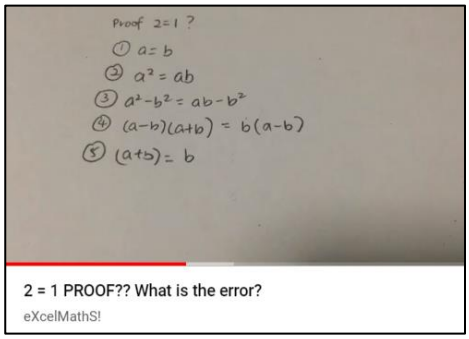
$$\frac{x = y}{zx = zy} \qquad \frac{x = y}{x - z = y - z}$$

$$\frac{x = y}{x + z = y + z} \qquad \frac{x = y}{x \div z = y \div z}$$

$$\frac{x + y = z \qquad x = y}{2y = z} \qquad \frac{x^2 - y^2 = xy - y^2}{(x - y)(x + y) = y(x - y)}$$

If the rules were all sound, we wouldn't be able to prove something false with them. However, we can use these rules to prove that 2 = 1, which is obviously wrong.

Watch the following video to see the proof of 2 = 1
https://www.youtube.com/watch?v=upG1dlAdJqs



Answer the following questions.

1. Which rule is unsound?
2. Use the inference rules above to write out the derivation of 2 = 1.

Tutorial 3 – Operational Semantics Part I

What is the Purpose of Tutorial 3?

This tutorial will investigate using logic and proofs to explain computation and execution of operations in a programming language.

By the end of the tutorial you will be able to:

- Define the terms evaluation, configuration, and state.
- Define formulas in Backus-Naur form.
- Explain the purpose of operational semantics.

Activity 1: Given the following inference rules, can you derive the formula $x = 3 \div 4$ from the assumption $18 \div 3x(12 - 4) = 1$?
You can use mathematical calculations anywhere in the derivation.
For example, replace $12 - 4$ with 8.

$$\frac{a \div b = c}{a = b \times c} \qquad \frac{a = b \times c}{a \div b = c} \qquad \frac{a = b \times c}{a \div c = b} \qquad \frac{a = b}{b = a}$$

The problem with informal semantics
An extract from the Algol 60 report [01] trying to describe a programming language precisely:

> "Finally the procedure body, modified as above, is inserted in place of the procedure statement and executed. If a procedure is called from a place outside the scope of any nonlocal quantity of the procedure body, the conflicts between the identifiers inserted through this process of body replacement and the identifiers whose declarations are valid at the place of the procedure statement or function designator will be avoided through suitable systematic changes of the latter identifiers."

[01] Backus et al. Report on the algorithmic language ALGOL 60, J. Numerische Mathematik 2:1, 1960

This is clearly *gobbledygook*.

Better alternative
To explain the semantics of a program, we can define a **formal** semantics. We will learn to read and write **precise, mathematical definitions** of program semantics to prove that programs have certain properties.

We focus on defining operational semantics, which describe the execution of programs mathematically.

> Question: Why is a formal mathematical description of a program less ambiguous than an informal description?

Reading Syntax

Before getting into meaning we need to take a step back and first precisely define language syntax. Backus-Naur Form (BNF) is a standard grammar formalism for defining language syntax. All BNFs are composed of terminals, non-terminals, and production rules. Terminals are traditionally uppercase, non-terminals lowercase. Production rules are how non-terminals are defined.

<non-terminal> ::= <form 1> | ... | <form n>

Each "form" above describes a string of terminals and non-terminals.

Consider the following as an example, where the non-terminal is S and the terminals are b and a.

S = b | Sa

Examples of words part of this language are b, ba, baaaaa.
Examples of words not accepted are baabaa, ab, a.

Consider the following language for Boolean logic, where **B** is a Boolean formula.

B ::= true | false | not B | B and B | B or B

Activity 2**:** Are the following formulas part of the Boolean logic language?

1. true or not true
2. not true and and false
3. false
4. false false
5. true or B
6. true and true or true or true and true

Stretch Activity: Write the BNF grammar for arithmetic
Hint: Numbers can be represented as 0, S0 = 1, SS0 = 2 where S = successor
For arithmetic operators, use addition +, subtraction -, and multiplication *.

States
States are a mapping from variable names to values, e.g.

$$\begin{array}{l} x \rightarrow 4 \\ y \rightarrow 5 \\ z \rightarrow 0 \end{array}$$

Configurations
A configuration is the information needed to execute a program.
For us, this will be the program and the current state or the program.

Example:

$$\{ \begin{array}{l} x \rightarrow 4 \\ y \rightarrow 5 \\ z \rightarrow 0 \end{array} \mid \begin{array}{l} z = x \\ x = y \\ y = z \end{array} \}$$

Evaluations
Evaluations show how configurations evolve over time as the program is executed.

$$C_1 \Rightarrow C_2 \Rightarrow \ldots \Rightarrow C_{n-1} \Rightarrow C_n$$

Evaluations tell us **step-by-step** how configurations evolve.

We can make two general rules. Look at the configuration: $\{ S \mid x = y; p \}$

- *S* is the state
- x = *y* is an assignment statement, and x and y are variables
- *p* is the rest of the program

We can then make the inference rule

$$\frac{\{ S \mid x = y; p \}}{\{ S \mid x = v; p \}}$$

where $y \rightarrow v$ is in S. Note *v* is a value. Then we use another inference rule:

$$\frac{\{ S \mid x = v; p \}}{\{ S' \mid p \}}$$

where *S'* is a new state, such that *S'* = *S* except x in *S'* now maps to *y*.

Activity 3: Write out the evaluation of the configuration

$$\left\{ \begin{array}{l} x \rightarrow 4 \\ y \rightarrow 5 \\ z \rightarrow 0 \end{array} \;\middle|\; \begin{array}{l} z = x \\ x = y \\ y = z \end{array} \right\}$$

Do this as a class, with the teacher, on the board.

Tutorial 3 – Homework

Question 1
Complete the following inference rules for Boolean values.

$$\frac{\{S \mid True\ and\ True\}}{\{\ S \mid True\ \}} \qquad \frac{\{S \mid True\ and\ False\}}{_____} \qquad \frac{\{S \mid True\ or\ False\}}{_____} \qquad \frac{\{S \mid False\ or\ False\}}{_____}$$

$$\frac{\{\ x \rightarrow True,\ S \mid True\ and\ x\}}{_____} \qquad \frac{\{\ x \rightarrow False,\ S \mid True\ and\ x\}}{_____}$$

$$\frac{\{\ x \rightarrow True,\ y \rightarrow False,\ S \mid y\ and\ x\}}{_____} \qquad \frac{\{\ x \rightarrow False,\ S \mid x\ and\ x\}}{_____}$$

Question 2:
Look at the following configuration, what is the program p and state S?
(Remember that = is assignment, == is comparison).
Write out the semantic evaluation for the following configuration.
Think about what rules you may need, e.g. the '*and*' from the previous question.

```
{  x → True
   y → True    |   z == x and y   }
   z → False
```

Tutorial 4 – Operational Semantic Part II

What is the Purpose of Tutorial 4?

This tutorial will investigate using logic and proofs to explain computation and execution of operations in a programming language.

By the end of the tutorial you will be able to:

- Understand the inference rules for 2 programming constructs
- Formulate derivations for at least 2 programming constructs
- Formulate derivations for blocks of code, demonstrating semantics

Activity 1: Write out the semantic evaluation for the following configuration.

$$\left\{ \begin{array}{l} x \rightarrow \texttt{True} \\ y \rightarrow \texttt{True} \\ z \rightarrow \texttt{False} \end{array} \;\middle|\; \texttt{z == x or y} \right\}$$

Branching

Branching is a programming structure commonly found in most, if not all, programming languages. You might know this as the "*if-statement*". There is a condition that needs to be satisfied before some code can be executed. Look at the if-statement below.

```
if n == 5:
    x = True;
```

The variable x is only set to True if the variable n is equal to 5, so the line of code is executed depending on the condition.

We can represent this as two inference rules, one for when the condition is met and one for when the condition is not met. Note that ⇓ denotes a derivation.

$$\frac{\{\, S \mid \mathit{if}\ b\ \mathit{then}\ p_1;\ p \,\} \qquad \begin{array}{c}\{\, S \mid b \,\} \\ \Downarrow \\ \{\, S \mid \mathit{True} \,\}\end{array}}{\{\, S \mid p_1;\ p \,\}} \qquad \frac{\{\, S \mid \mathit{if}\ b\ \mathit{then}\ p_1;\ p \,\} \qquad \begin{array}{c}\{\, S \mid b \,\} \\ \Downarrow \\ \{\, S \mid \mathit{False} \,\}\end{array}}{\{\, S \mid p \,\}}$$

Activity 2: Apply the above inference rules on the following configurations.

$$\left\{ \begin{array}{l} x \rightarrow \texttt{True} \\ y \rightarrow 4 \\ z \rightarrow \texttt{False} \end{array} \;\middle|\; \begin{array}{l} \texttt{if x:} \\ \quad \texttt{z = True} \end{array} \right\}$$

$$\left\{ \begin{array}{l} x \rightarrow \texttt{True} \\ y \rightarrow 4 \\ z \rightarrow \texttt{False} \end{array} \;\middle|\; \begin{array}{l} \texttt{if x and z:} \\ \quad \texttt{z = 5} \end{array} \right\}$$

$$\left\{ \begin{array}{l} x \rightarrow \texttt{True} \\ y \rightarrow 4 \\ z \rightarrow \texttt{False} \end{array} \;\middle|\; \begin{array}{l} \texttt{if x or z:} \\ \quad \texttt{z = 5} \\ \quad \texttt{y = y + z} \end{array} \right\}$$

We have seen the semantics for if-statements, but if-statements can also have an "*else*" construct. When the condition of the if-statement is false, we execute the code in the else construct.

Activity 3: What is the conclusion to each of the following inference rules.

First, the rule where the condition is true.

$$\frac{\{\, S \mid b \,\} \Downarrow \{\, S \mid True \,\} \qquad \{\, S \mid if\ b\ then\ p_1\ else\ p_2;\ p \,\}}{\quad}$$

And the rule where the condition is false.

$$\frac{\{\, S \mid b \,\} \Downarrow \{\, S \mid False \,\} \qquad \{\, S \mid if\ b\ then\ p_1\ else\ p_2;\ p \,\}}{\quad}$$

Activity 4: Write out the operational semantics derivation for the following configuration.

```
{ w → True                  if y or w:
  x → 5                       x = x + z
  y → True        |         else:                }
  z → 6                       x = x

                              x = x + 1
```

The final configuration will not have a program, as we should have executed everything.

Repetition

Now think about a while-loop. As a configuration it would look something like

$$\{ S,\ while\ b\ do\ p_1\ ;\ p \}\ .$$

If the condition is False, then the semantics would simply be

$$\frac{\{ S \mid b \} \Downarrow \{ S \mid False \} \qquad \{ S \mid while\ b\ do\ p_1;\ p \}}{\{ S \mid p \}}$$

However, if the condition is True, then we need to execute the code inside the loop. After executing it, we need to check the condition again. Therefore, the rule is:

$$\frac{\{ S \mid b \} \Downarrow \{ S \mid True \} \qquad \{ S \mid while\ b\ do\ p_1;\ p \}}{\{ S \mid p_1;\ while\ b\ do\ p_1;\ p \}}$$

Activity 5: Write out the operational semantics derivation for the following configuration.

$$\left\{ \begin{array}{l} n \rightarrow 3 \\ i \rightarrow 1 \end{array} \ \middle|\ \begin{array}{l} \texttt{while n != 1:} \\ \quad \texttt{i = i * n} \\ \quad \texttt{n = n - 1} \end{array} \right\}$$

Note: `!=` is an operator and is read as "*is not equal to*".

Tutorial 4 – Homework

A **leap year** is a year that contains an additional day added to keep the calendar year synchronized with the astronomical year or seasonal year.

Every year that is exactly divisible by four is a leap year, except for years that are exactly divisible by 100, but these centurial years are leap years if they are exactly divisible by 400. For example, the years 1700, 1800, and 1900 are not leap years, but the years 1600 and 2000 are.

Look at the following Python code. It takes a year and checked whether that year was a leap year or not. Note: The % denotes the remainder function, e.g. 5 % 3 = 2.

```
year = 2000
leapYear = False

if (year % 4) == 0:
    if (year % 100) == 0:
        if (year % 400) == 0:
            leapYear = True
        else:
            leapYear = False
    else:
        leapYear = False
else:
    leapYear = False

print (leapYear)
```

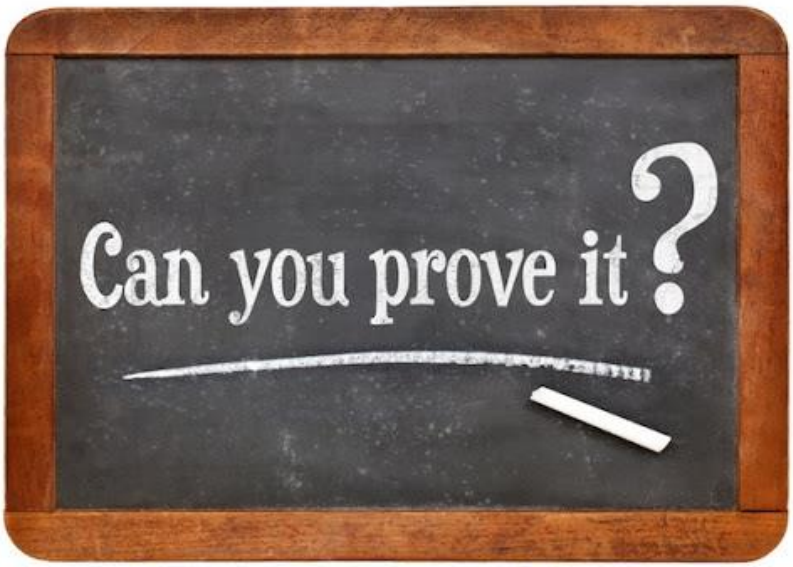
Look at the following configuration.

$$\left\{ \begin{array}{l} \texttt{year} \rightarrow 2000 \\ \texttt{leapYear} \rightarrow \texttt{False} \end{array} \;\middle|\; \begin{array}{l} \texttt{if (year \% 4) == 0:} \\ \quad \texttt{if (year \% 100) == 0:} \\ \quad\quad \texttt{if (year \% 400) == 0:} \\ \quad\quad\quad \texttt{leapYear = True} \\ \quad\quad \texttt{else:} \\ \quad\quad\quad \texttt{leapYear = False} \\ \quad \texttt{else:} \\ \quad\quad \texttt{leapYear = False} \\ \texttt{else:} \\ \quad \texttt{leapYear = False} \end{array} \right\}$$

Write out the operational semantics derivation for this program. You do not have to write the program out for every line and can use abbreviations to make your derivation neater.

You can perform the calculations in the derivation, i.e. replace 2000 % 4 with 0.

Tutorial 5 – Proofs as Programs



What is the Purpose of Tutorial 5?

This tutorial will look more generally into comparing maths and computing and will introduce the concept of the Curry-Howard correspondence theorem.

By the end of the tutorial you will be able to:

- Relate formulas to data types
- Define the Curry-Howard correspondence theorem
- Describe the benefits of looking at programs as proofs

Activity 1: Write down as many data types you can think of, e.g. integer

So far, we have seen the idea of building proofs to describe the semantics of a computer program, but can we generalise this to any proof of anything? Below is the inference rule we looked at in Tutorial 2. If we replace A and B with data types, it would be more natural to think of this as applying a function to something of type A in order to obtain something of type B. We can start to think of the formula A → B as a function.

$$\frac{A \rightarrow B \quad \text{AND} \quad A}{B}$$

Activity 2: Look at the following formulas and think about how they might be interpreted computationally. HINT: Replace the letters with data types.

$$(A, B) \rightarrow C \qquad\qquad A \rightarrow (B \rightarrow C)$$

This correspondence between mathematics/logic and computation was noticed by Haskell Curry and William Howard, thus we call it the **Curry-Howard correspondence**.

They noticed the relationship and declared:

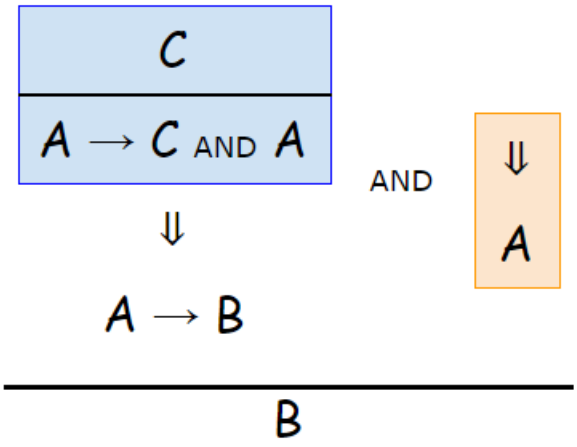
Formulas	=	Data Types
Proofs	=	Programs
Manipulation	=	Execution

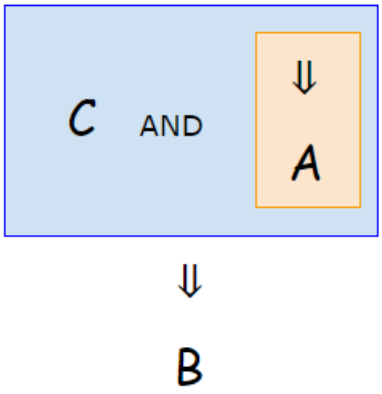
We haven't seen much about proof manipulation in this course, so we will cover an example now.

Observe the two inference rules, function introduction and function application

$$\frac{B}{A \rightarrow B \text{ AND } A} \qquad\qquad \frac{A \rightarrow B \quad \text{AND} \quad A}{B}$$

Now look at the situation when we are applying a function to something. We can "*execute*" this function by moving the derivation around so that we no longer introduce the function.





Programming languages have already been designed using this perspective of computing, i.e. viewing proofs as programs. Below is an example of a program in Coq, a proof assistant language.

```
Theorem forward_small : (forall A B : Prop, A -> (A->B) -> B).
Proof.
 intros A.
 intros B.
 intros proof_of_A.
 intros A_implies_B.
 pose (proof_of_B := A_implies_B proof_of_A).
 exact proof_of_B.
Qed.
```

This program is just the application rule we've seen on the previous page. This style of programming is good for software *formal verification*, i.e. the act of proving correctness of an algorithm with respect to an underlying specification.

Question: How does this style/method of looking at programs different from the operational semantics we looked at in Tutorial 3 and 4?

Why are we doing this again?

This perspective on programming has many advantages and disadvantages.

Activity 3: Why do you think this perspective on programming will help with:

Verifying the correctness of a program (Does what we think it should)
Verifying if two different programs are equivalent
Computers generating its own code

Tutorial 5 – Final Assignment

The final assignment involves a *written report* and a *problem sheet*, and you will need to do both.

Written report
Write a report (about 1000 words) that answers the question:

How are mathematics and computing similar?

In your report, please include

- An introduction to the idea of semantics, including references.
- An explanation of the Curry-Howard Theorem, including references.
- A conclusion, giving your own opinion about thinking of programs as proofs, it's benefits and limitations, including references.

You can use the above points to help structure your essay.

Essays will be marked on how clear your explanations are, so use examples and references to your advantage. A good essay should make clear the relationship between the theory discussed during tutorials and the real world.

Your essay should be structured using appropriate subheadings, including introduction and conclusion.

Label examples (e.g. Example 1), and comment about them in the text. Include references, citing work from published books, articles, videos, etc.

You will be assessed on the quality of your written communication. Marks will also be given for good spelling, punctuation and grammar, as well as good use of subject vocabulary.

Also don't forget to **PEE**
- Make your **P**oint
- Give **E**vidence
- **E**xplain your opinion

Understand	Be able to do
The importance of semantics (in both computer science, and in communication). Understanding proofs and logics Operational semantics The connection between computing and mathematics, and why this perspective is unique and relevant.	Analyse given data and identify all relevant mathematical information. Demonstrate a high level of knowledge and understanding. Critically think about a new perspective of computing.

Problem Sheet

Below are 3 examples of configurations. These configurations are taken from the worksheet from Homework 1. Evaluate each of the configurations and determine the final state of the program.

Note: #3 uses a function, which is stored in the state and will be used multiple times.

Code Sample #1

```
{ num1 → 1.5
  num2 → 6.3   |  sum = num1 + num2  }
```

Code Sample #2

```
                 if (num1 >= num2) and (num1 >= num3):
                     largest = num1
  num1 → 10      elif (num2 >= num1) and (num2 >= num3):
{ num2 → 14   |      largest = num2                      }
  num3 → 12      else:
                     largest = num3
```

Code Sample #3

```
                 if n == 0:
                   return 1
{ def fact(n): → else:                  | fact(2) }
                   return n*(fact(n-1))
```

Tutorial 6 – Feedback tutorial

What is the Purpose of Tutorial 6?

- To receive feedback on your final assignment
- To respond to the feedback from your Uni Pathways teacher
- To write targets for improvement on your final assignment

Final assignment feedback from your Uni Pathways Teacher

(Remember to look at the mark scheme to help you understand what you have done well so far, and how you can do even better in your final assignment)

Here are three things that my Uni Pathways Teacher thought I did well in my draft assignment
•   •   •

Here are three things that my Uni Pathways Teacher thinks that I could do to get a higher mark in my final assignment
•   •   •

Tasks from my Uni Pathways Teacher to do during the feedback tutorial to help me improve
•   •   •
My response:

Actions I will take to improve my final assignment after this tutorial…
•   •   •

Hand in date for my final assignment:	

Tutorial 7 – Final tutorial

What is the Purpose of Tutorial 7?

- To receive feedback and a grade on your final assignment.
- To reflect on the programme including what you enjoyed and what was challenging.
- To ask any questions you may have about university.

Final assignment feedback from my Uni Pathways Teacher

Final mark:	University style grade:
Feedback: Here are three things that my Uni Pathways teacher thought I did well in my final assignment • • •	

Here are three things that my Uni Pathways teacher thinks I should remember for when I am doing this kind of study in the future
• • •

University

What questions do you still have about University after taking part in Uni Pathways?
• • •

Reflecting on Uni Pathways

What did you most enjoy about Uni Pathways?
• • •

What did you find challenging about the programme?	How did you overcome these challenges?
• • •	• • •

Appendix 1 – Referencing correctly

When you get to university, you will need to include references in the assignments that you write, so we would like you to start getting into the habit of referencing in your Brilliant Club assignment. This is really important, because it will help you to avoid plagiarism. Plagiarism is when you take someone else's work or ideas and pass them off as your own. Whether plagiarism is deliberate or accidental, the consequences can be severe. In order to avoid losing marks in your final assignment, or even failing, you must be careful to reference your sources correctly.

What is a reference?

A reference is just a note in your assignment which says if you have referred to or been influenced by another source such as book, website or article. For example, if you use the internet to research a particular subject, and you want to include a specific piece of information from this website, you will need to reference it.

Why should I reference?

Referencing is important in your work for the following reasons:

- It gives credit to the authors of any sources you have referred to or been influenced by.
- It supports the arguments you make in your assignments.
- It demonstrates the variety of sources you have used.
- It helps to prevent you losing marks, or failing, due to plagiarism.

When should you use a reference?

You should use a reference when you:

- Quote directly from another source.
- Summarise or rephrase another piece of work.
- Include a specific statistic or fact from a source.

How do I reference?

There are a number of different ways of referencing, and these often vary depending on what subject you are studying. The most important to thing is to be consistent. This means that you need to stick to the same system throughout your whole assignment. Here is a basic system of referencing that you can use, which consists of the following two parts:

1. **A marker in your assignment**: After you have used a reference in your assignment (you have read something and included it in your work as a quote, or re-written it your own words) you should mark this is in your text with a number, e.g. [1]. The next time you use a reference you should use the next number, e.g. [2].
2. **Bibliography**: This is just a list of the references you have used in your assignment. In the bibliography, you list your references by the numbers you have used, and include as much information as you have about the reference. The list below gives what should be included for different sources.

    a. **Websites** – Author (if possible), title of the web page, website address, [date you accessed it, in square brackets].

    E.g. Dan Snow, 'How did so many soldiers survive the trenches?', http://www.bbc.co.uk/guides/z3kgjxs#zg2dtfr [11 July 2014].

    b. **Books** – Author, date published, title of book (in italics), pages where the information came from.

    E.g. S. Dubner and S. Levitt, (2006) *Freakonomics*, 7-9.

    c. **Articles** – Author, 'title of the article' (with quotation marks), *where the article comes from* (newspaper, journal etc.), date of the article.

    E.g. Maev Kennedy, 'The lights to go out across the UK to mark First World War's centenary', Guardian, 10 July 2014.

Notes

UNI PATHWAYS

UNI PATHWAYS

thebrilliantclub.org