National 5 Computer Science Crash Course

Last Updated: 12/01/21

Contents

Contents

Computer Systems

Before you start, I would suggest you get yourself a Hodder & Gibson SQA National 5 Computer Science Book to aid in studying and revision as it contains a lot of good information and practice questions for you.

Software Development

Development Methodologies

Computational Thinking

Computational thinking is designing a solution to a problem so it can be solved by a computer.

Analysis: Understanding the problem

Design: Work out steps to solve it

Implementation: Create a working program based on design

Testing: Making sure there’s no mistakes

Documentation: Describing what each part does

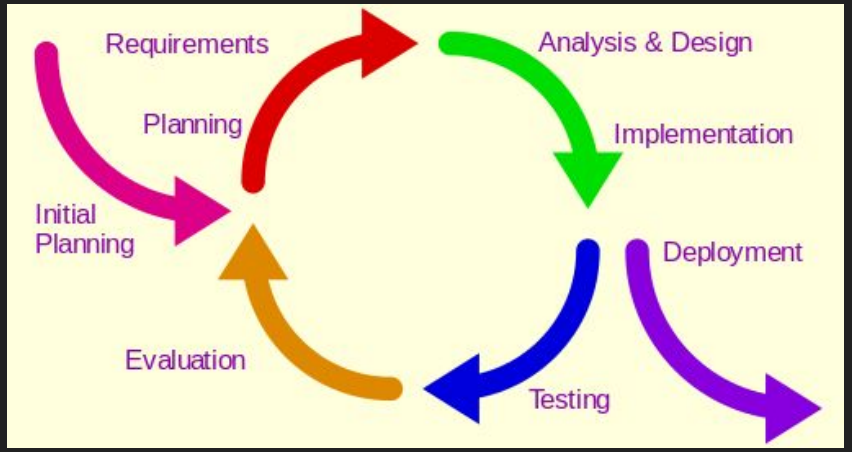
Evaluation: How well does the solution fulfill the original request

Algorithms

An Algorithm is working out the series of steps to solve a problem

Video:

The Waterfall Model (Iterative development methodology)



Some Advantages:

- Simple to understand and use
- Move from one stage to the next
- Ideal for small Projects

Iterative:

- Previous Stages can be revisited and improved upon

There is often multiple cycles until deployment

Questions

Attempt All Questions without looking back at previous slides

1. What is Computational Thinking?
2. What is an algorithm?
3. State the term for:
   - a. Looking at and understanding a problem
   - b. Working out a series of steps to solve a problem
   - c. Changing a design into a program
   - d. Checking to find whether a program contains mistakes
   - e. Describing what each part of a program does
   - f. Ensuring that the software fulfills the original requirements
4. Write an algorithm for making a cup of tea. Be exact in your steps

Software Analysis

Analysis

Analysis: Looking at and understanding a problem

- Meeting the client

Software Specification produced at the end of this stage

Example of Specification:
https://belitsoft.com/custom-application-development-services/software-requirements-specification-document-example-international-standard

Analysis: Purpose and Functional Requirements

Identify inputs, processes and outputs for the following scenario:

- “I want a program that takes in a person’s name and age, works out the year they were born and tells them.”

Answer:

“I want a program that takes in a person’s name and age, works out the year they were born and tells them.”

Inputs: Name & Age

Processes: Works out year they were born

Outputs: Year they were born

Questions

Analyse the following problems and produce the **purpose** and **functional requirements** for each. Including **inputs**, **processes** and **outputs** for each.

Once Finished, Attempt to code the Questions as well

(dc): Don’t Code

1. “A Program is to be written to simulate the basic Arithmetic functions of a calculator. The user should be asked to input two numbers and the sum (+), difference (-), product (*) and quotient (/) should be displayed”
2. “A program is needed to calculate the speed at which an arrow is travelling at is passes two sensors. It measures the time it takes the arrow to travel a marked distance between two sensors and calculates the speed of the arrow. The program should show the speed at which the arrow was travelling” (dc)

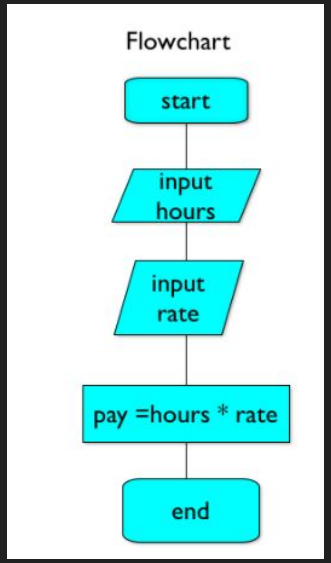
Questions

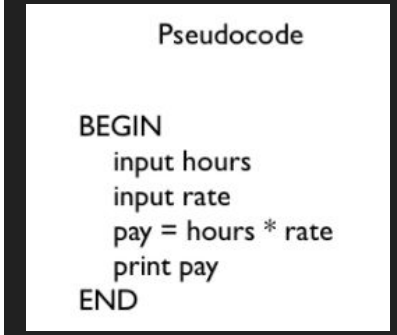
1. “A School needs a program to grade students’ marks after an exam. The program should take in an integer between 0 and 100 and output a grade”

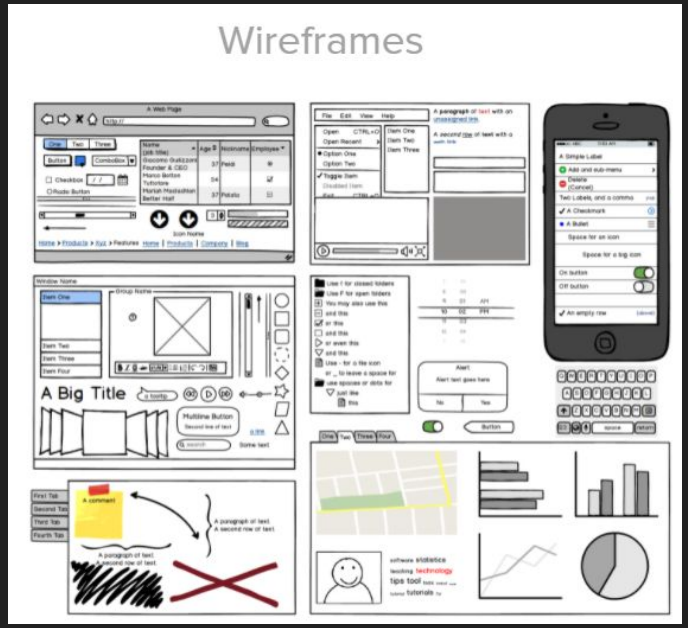
Design

Design

Working out the steps to solve a problem







Design

Design Notation are ways of representing a program or algorithm

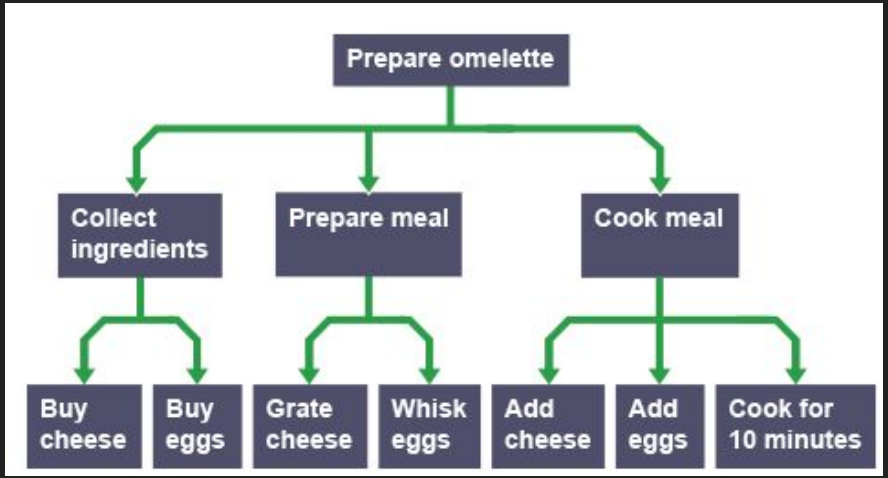
Graphical Design Notation uses shapes to describe a design (Structure diagrams and flowcharts)

Pseudocode uses ordinary English to define steps in a problem

Structure Diagrams

Breaking down a problem into smaller and smaller problems (sub-problems)

Looked at from top down and left to right



(Flowchart) different boxes mean different things

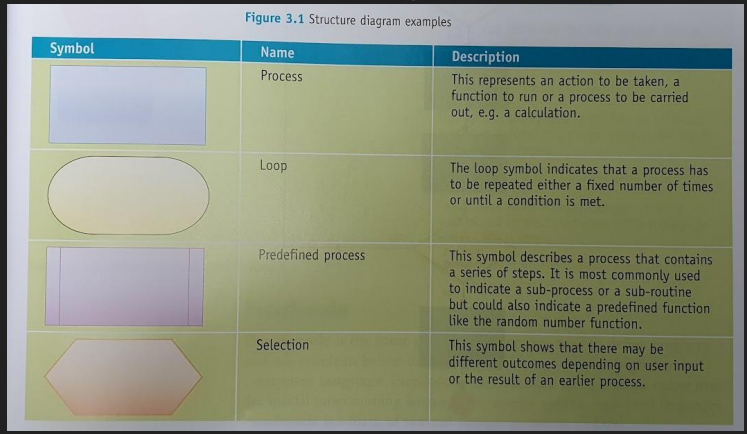


Figure 3.1 Structure diagram examples

Symbol	Name	Description
	Process	This represents an action to be taken, a function to run or a process to be carried out, e.g. a calculation.
	Loop	The loop symbol indicates that a process has to be repeated either a fixed number of times or until a condition is met.
	Predefined process	This symbol describes a process that contains a series of steps. It is most commonly used to indicate a sub-process or a sub-routine but could also indicate a predefined function like the random number function.
	Selection	This symbol shows that there may be different outcomes depending on user input or the result of an earlier process.

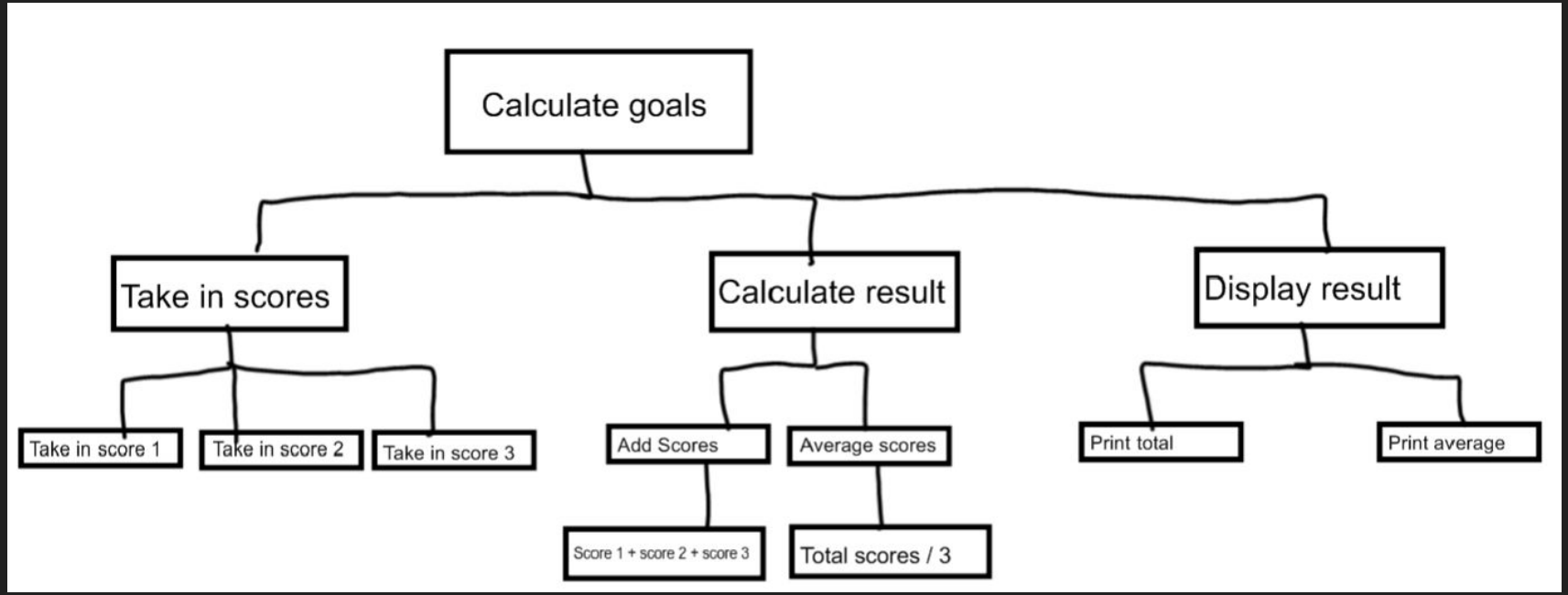
Questions

Draw a structure diagram for the following specification

1. “I want a program that takes in three football scores, works out the total goals scored and the average, then outputs a message with the total and average”.

Attempt this before looking at the answer below

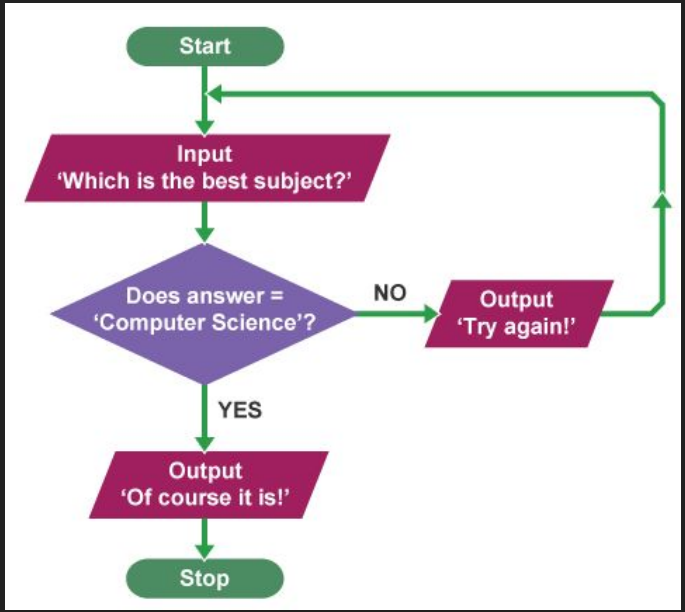
Answer


Calculate goals
Take in scores
Calculate result
Display result
Take in score 1
Take in score 2
Take in score 3
Add Scores
Average scores
Print total
Print average
Score 1 + score 2 + score 3
Total scores / 3

Flowcharts

Represents a set of instructions including flow of program

It has specific symbols for set actions



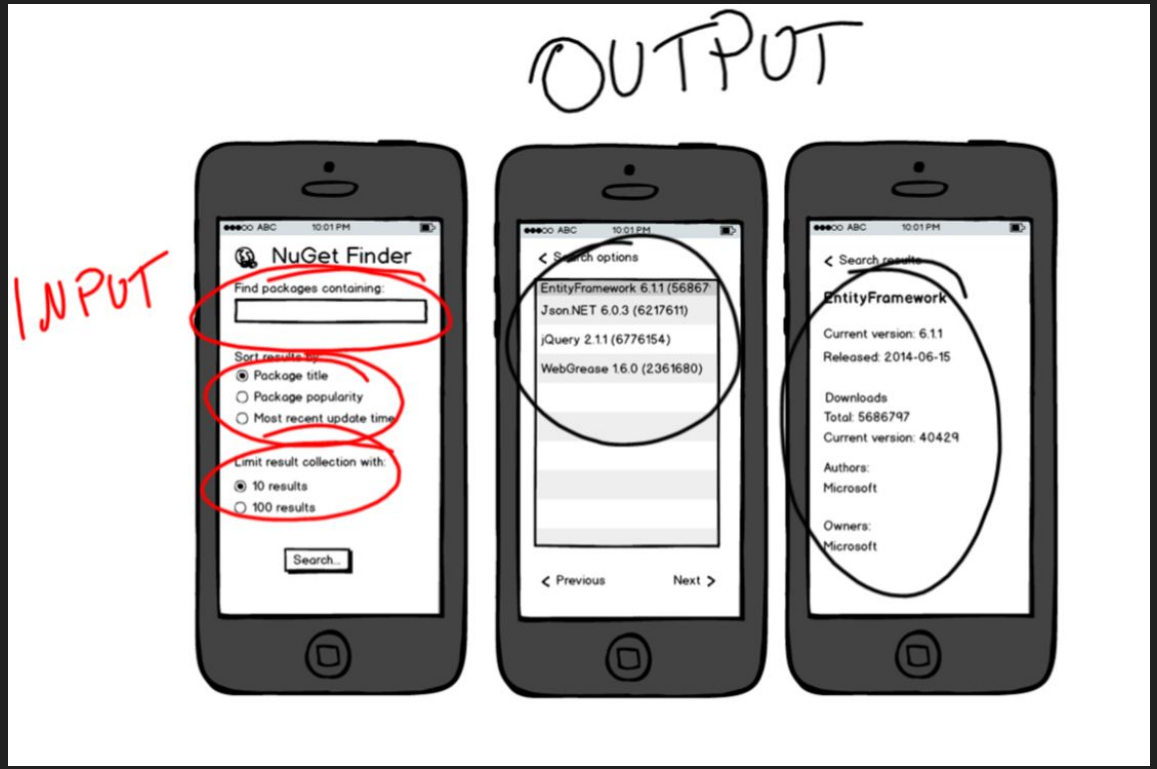
Name	Symbol	Usage
Start or Stop	Start/Stop	The beginning and end points in the sequence.
Process	Process	An instruction or a command.
Decision	Decision	A decision, either yes or no.
Input or Output	Input/Output	An input is data received by a computer. An output is a signal or data sent from a computer.
Connector		A jump from one point in the sequence to another.
Direction of flow		Connects the symbols. The arrow shows the direction of flow of instructions.

Pseudocode

Using English-like text to describe steps in a program

1. INPUT “which is the best subject?”
2. WHILE answer != “Computing Science” THEN:
3. 	SEND “Try Again” TO DISPLAY
4. SEND “Of course it is” TO DISPLAY

Wireframe


OUTPUT
INPUT
NuGet Finder
Find packages containing:
Sort results by:
Package title
Package popularity
Most recent update time
Limit result collection with:
10 results
100 results
Search..
< Search options
EntityFramework 6.1.1 (56867
Json.NET 6.0.3 (621761)
jQuery 2.1.1 (6776154)
WebGrease 1.6.0 (2361680)
< Previous
Next >
< Search results
EntityFramework
Current version: 6.1.1
Released: 2014-06-15
Downloads
Total: 5686797
Current version: 40429
Authors:
Microsoft
Owners:
Microsoft

Practical Questions

1. Using a graphical design notation of your own choice, write algorithms for the following problems
   - “Calculate the circumference of a circle given the radius as input (2πR)”
   - “Input validation for days 1-31 with a suitable message”
   - “A quiz with 4 questions and a 2nd chance to get the correct answer after a hint is given”
2. Investigate online tools for creating design notations. Some Examples
   - https://cacoo.com
3. Create programs to solve each of the problems in Q1 Using a Programming Language of your choice

Questions

1. What is design?
2. What is an algorithm
3. What is design notation?
4. Name and describe one design notation which you are familiar with
5. What is graphical design notation?
6. If you were asked to design a software application, which design notation would you choose? Explain why you chose this application
7. What is Pseudocode?
8. What language is used in pseudocode?
9. What makes pseudocode so useful when describing the design of a program
10. In programming what is a wireframe?
11. What details might be contained in a wireframe?
12. Using a wireframe, design user interfaces for the following programs

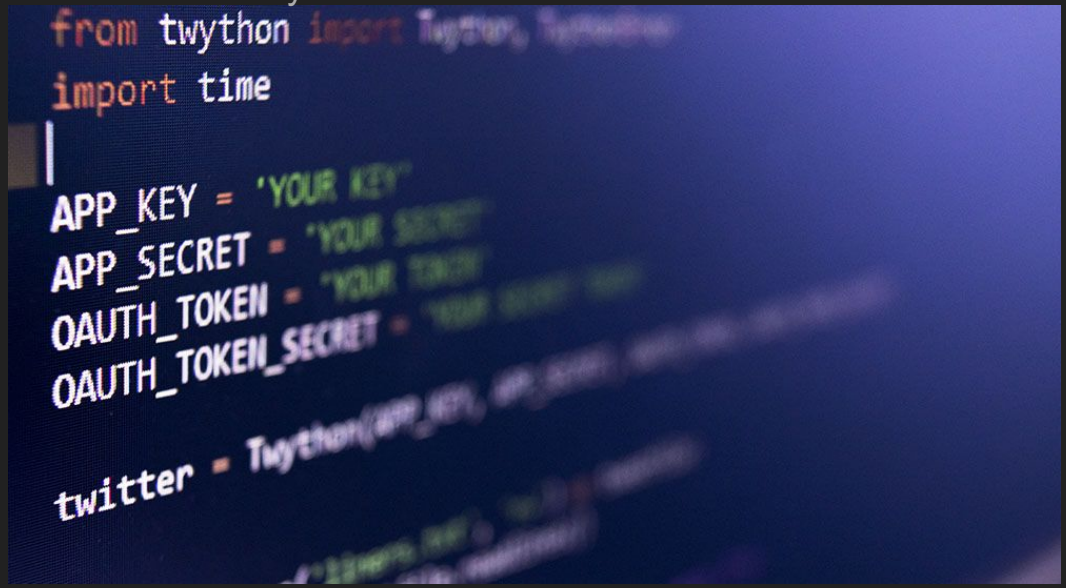
- A simple program designed to calculate the volume of a room given the length, breadth and height
- CardsCo need a program that will print invitations to parties. The input to the program should include date, time place and event.

Implementation

Implementation

Actually coding your designed solution

Python Code For a Twitter Bot



Implementation

A **Variable** is the name given to a **single storage location in memory** and the **data** that is stored in it.

A **Character** is a **symbol**, **letter** or **number** e.g letter = “J”

A **String** is a list of **characters** e.g name = “Chris”

An **Integer** is a **whole** number e.g age = 32

A **Real number** is a **decimal** number e.g pi = 3.141592

A **Boolean** is either **True** or **False** e.g isTrue = True

Example of a variable

```
1   name = "Desperate Dan"
2   age = 52
3
4   print(name)
5   print(age)
```

Questions

1. What is a variable?
2. Name two data types
3. What is string data?
4. What is numeric (integer) data?
5. What is numeric (real) data?
6. What is Boolean Data?

Data Structures

An **Array** is a variable that can hold multiple values of one data type

E.g names = [“Aaron”, “Orrin”, “Ewan”, “Andreas”, “QiHao”]

An Array always starts at index 0 and goes up from there

E.g names[0] = “Aaron”

Computational Constructs

Computational Constructs

What are Computational Constructs?

Computational Constructs are the **building blocks** or **parts** of a programming language used to **create a program**

Some examples: if-else, while, for, try-except

Assignment is the “giving” of a value to a variable

E.g name = “Benji”

Age = 15

Age += 1

Computational Constructs

Arithmetic Operations is a process that is carried out on an item of data

e.g +, -, /, *

age += 1

Agediff = age1 - age2

monthsOld = age * 12

decadesOld = age / 10

Question: Write Code that takes in your age and works out the year you were born

Computational Constructs

String Concatenation (The Joining to 2 strings into 1)

e.g firstName = “Benji”

surname = “Thompson”

fullName = firstName + surname

Question:

Write code that asks for first & surnames, then joins them together with a space

Computational Constructs

Relational Operators

= Equals

== Compared to

> Greater Than

< Less Than

=> Greater Than or Equal To

<= Less Than or Equal To

!= Not Equal To

Computational Constructs

AND Logical Operator

Both Conditions must be true for the output to be true

e.g if age > 10 < 15: //Do Something...

OR Logical Operator if age > 10 || age < 15: //Do Something

Only one condition must be true for the output to be true

NOT Logical Operator if age != 10: //Do Something

Condition must be false for the output to be true

Computational Constructs

Simple Conditional Statement (Only one condition)

```
If age > 18:
    //Do Something…
```

Complex Conditional Statement (More than one condition)

```
If age > 18 and age < 35:
    //Do something...
```

Questions

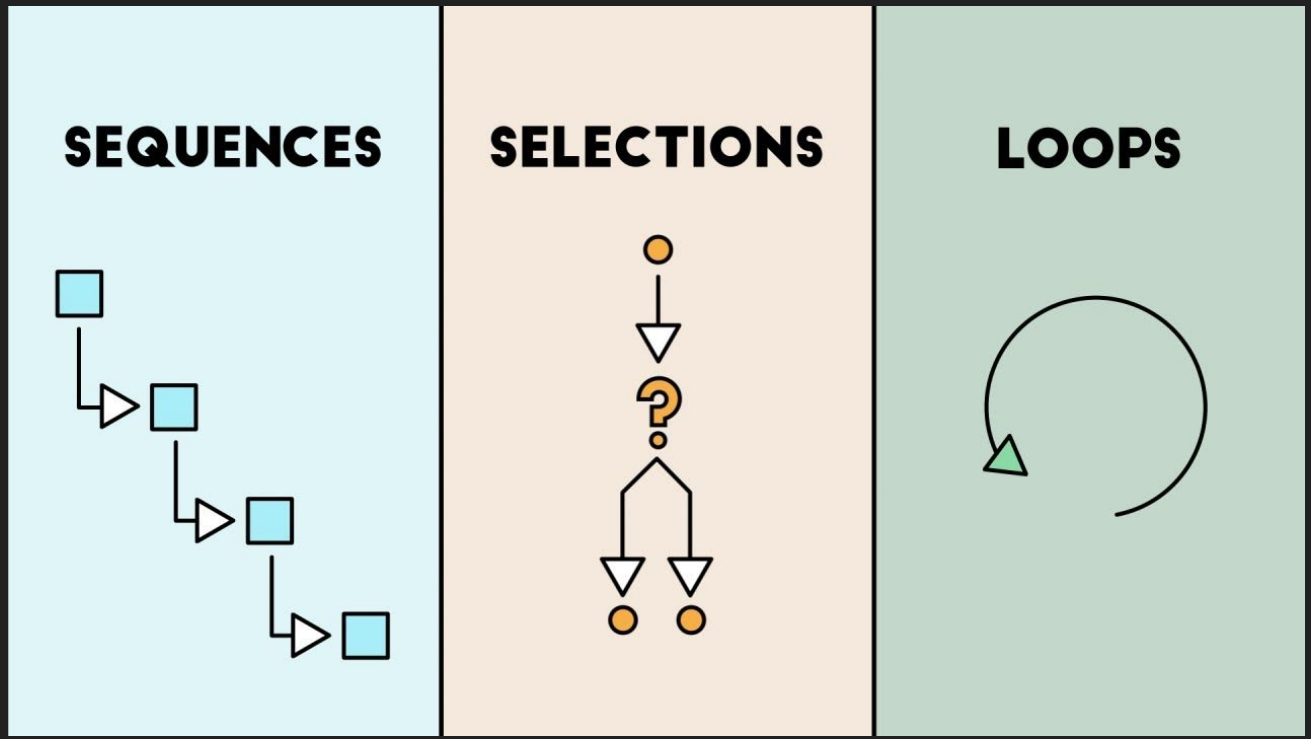
1. What is an assignment used for?
2. Write assignment statements for your name and age using a programming language of your choice
3. Name two operations used in programming languages
4. Which type of operations produce an answer of true or false?
5. What is concatenation
6. State two relational operators
7. Which type of operation uses “NOT”?

Practical Questions

1. Write code that asks for your age and tells you if you can claim your pension

Implementation

Control Structures **control the order that lines of code are evaluated in**



Implementation

Sequence is the order in which steps are done in

```
algorithm to add two numbers
1.    Ask for the first number
2.    Get the first number                                    ← INPUT
3.    Ask for the second number
4.    Get the second number                                   ← INPUT
5.    Calculate total as first number + second number         ← PROCESS
6.    Display the total                                       ← OUTPUT
```

Implementation

Fixed loops loop code a predefined number of times

e.g for i in range(10):

    //Do Something…

The STEP command steps different amounts during loops

e.g for i in range(0,100,2):

    //Do Something...

Implementation

Conditional Loops only loops as long as the condition is true

e.g password = “qwerty”

```
pWord = input(“Password”)
While pWord != password:
    //Do Something…
```

Question: Store two numbers and ask the user to add them up, loop for as long as their answer is wrong

Implementation

Predefined Functions is code that has already been written and performs an action, **returning** a value.

Procedures is code that has already been written and performs an action, **not** returning a value.

Procedures and functions are both examples of **sub-programs**. They perform specific task(s) as part of a larger program.

Both Procedures and Functions sometimes require **parameters**, this is the data that the subprogram requires in order to function

Some Examples:

len (Length) Which returns a number of character in a String

Round (Round) Which round a number to a specific number of decimal places

Random (random) Which chooses a random element from a list

Range (range) Which chooses a random element inside the specified range

Question: Write a program to pick 6 lottery numbers from 1 - 59

Implementation

Standard Algorithm: A Programmable “Recipe” for doing certain tasks

At National 5 these are:

- Input Validation
- Running Total within a Loop
- Traversing a 1D Array

Input Validation

Making Sure the Input Given is Acceptable

```
3   # Take in and validate a student's
    percentage grade
4
5   grade = int(input("Enter percentage")
    )
6
7   while grade <0 or grade > 100:
8       grade = int(input("INVALID: Enter
        percentage"))
```

Running Total Within a Loop

How far the loop has gone through

```
SET total TO 0
SET points to [1,3,1,2,4,5,6]
FOR i FROM 0 to 6 DO:
    SET total to points[i] + total

SEND total TO DISPLAY
```

```
total = 0
points = [1,2,3,4,5,6]

for i in range(len(points)):
  total = total + points[i]

print (total)
```

Traversing a 1D Array

Running Through All Values in an array

```
****METHOD 1******
SET giant TO ["fee","fi","fo","fum"]
FOR EACH line FROM giant DO
        print giant[line]


****METHOD 2******
SET giant TO ["fee","fi","fo","fum"]
FOR i FROM 0 to LENGTH(giant) DO
        print giant[i]
```

```
giant = ["fee", "fi", "fo", "fum"]

for i in range(len(giant)):
  print(giant[i])
```

Questions

1. What is a sub-program
2. Name two types of sub-program
3. What is the difference between a procedure and a function?
4. What is a predefined function?
5. State one example of a predefined function
6. Using a programming language of your choice, explain what the predefined function you named in 5 does
7. What is a parameter?

Testing, Documentation & Evaluation

Testing

Systematically and comprehensively testing your program to make sure it works

Test Case	Explanation	Example where a score should be between 0 and 50
Normal	Data that you would expect to work or be accepted and that lies within the range	2, 45
Extreme	Data at the lower and upper limits of the range	0, 50
Exceptional	Data that should not be accepted by the program	-7, Yaney

Testing

Error Types: Syntax Error

An error that occurs when the rules of the programming language are broken

```
name = "my name is John"

prints(len(name))
```

Error Types: Execution Error

An error that occurs while a program is running

```
number1 = 5
result = number1 / 0
```

Error Types: Logic Error

An error that occurs when mistakes are made in the design of a program. The program will run but it won't give you the expected output.

```
number1 = 5

result = 0

number1 = number1 + 1

for i in range(100):
  if number1 == 5:
    print("you're the winner")

  number1 = number1 + 1
```

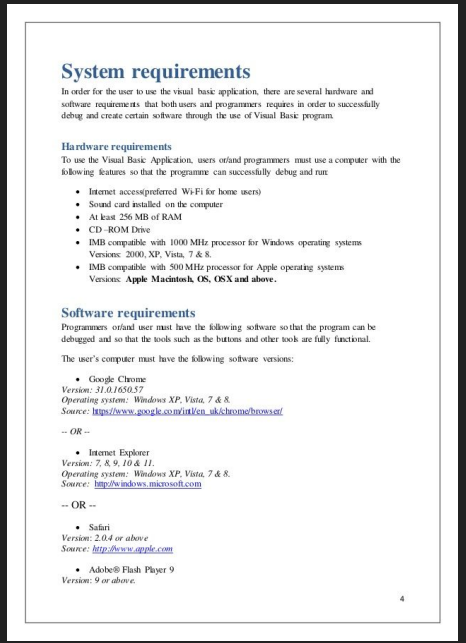
Test Tables

Tables that has the expected output and the actual output side by side

Test Table - Accepting input of a score that falls between 1 and 99 inclusive				
Category	**Test Data**	**Expected Result**	**Actual Result**	**Passed Testing**
Normal	19	Data accepted	Data accepted	✔
Normal	67	Data accepted	Data accepted	✔
Extreme	1	Data accepted	Data accepted	✔
Extreme	99	Data accepted	Data accepted	✔
Exceptional	0	Data rejected	Data rejected	✔
Exceptional	100	Data rejected	Data accepted	✘
Exceptional	yesterday	Data rejected	Data rejected	✔

Documentation

Things like User Guides and Technical Guides



System requirements

In order for the user to use the visual basic application, there are several hardware and software requirements that both users and programmers requires in order to successfully debug and create certain software through the use of Visual Basic program.

Hardware requirements

To use the Visual Basic Application, users or/and programmers must use a computer with the following features so that the programme can successfully debug and run:

- Internet access(preferred Wi-Fi for home users)
- Sound card installed on the computer
- At least 256 MB of RAM
- CD –ROM Drive
- IMB compatible with 1000 MHz processor for Windows operating systems Versions: 2000, XP, Vista, 7 & 8.
- IMB compatible with 500 MHz processor for Apple operating systems Versions: **Apple Macintosh, OS, OSX and above.**

Software requirements

Programmers or/and user must have the following software so that the program can be debugged and so that the tools such as the buttons and other tools are fully functional.

The user's computer must have the following software versions:

- Google Chrome

Version: 31.0.1650.57
Operating system: Windows XP, Vista, 7 & 8.
Source: https://www.google.com/intl/en_uk/chrome/browser/

-- OR --

- Internet Explorer

Version: 7, 8, 9, 10 & 11.
Operating system: Windows XP, Vista, 7 & 8.
Source: http//windows.microsoft.com

-- OR --

- Safari

Version: 2.0.4 or above
Source: http://www.apple.com

- Adobe® Flash Player 9

Version: 9 or above.

Evaluation

Evaluate the final program against software specification (Analysis stage)

Fitness for Purpose: It Does as specified in the specification

Efficiency of Code: How fast does the code run and how much memory does it use

Robustness (Testing Stage): Can the code take unexpected inputs and not crash?

Readability of Code (Implementation Stage): How easy is it to read & understand your code.

Readability Includes

- Internal Commentary (Comments)
- Meaningful Variable Names (age = 10 instead of a = 10)
- Indentation / White Space (Spaces or tabs between lines of code)

Questions

Q13c: https://www.sqa.org.uk/pastpapers/papers/papers/2019/N5_Computing-Science_QP_2019.pdf#page=12

Q5: https://www.sqa.org.uk/pastpapers/papers/papers/2019/N5_Computing-Science_QP_2019.pdf#page=4

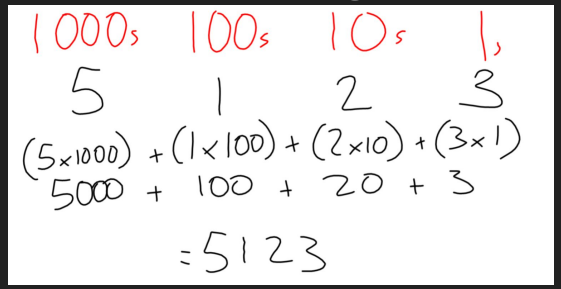
Q21c: https://www.sqa.org.uk/files_ccc/ComputingScienceSQPRN5.pdf#page=24

Computer Systems

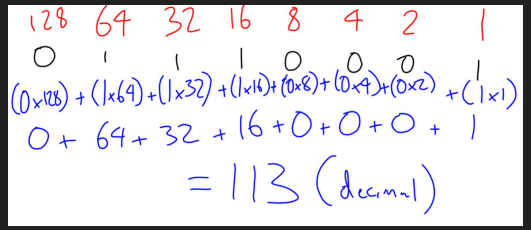
Data Representation

Binary

How we do things in decimal (Base 10)



How we do things in Binary (Base 2)



Floating Point Representation

0.314×10^{25}

314 is the mantissa

25 is the exponent

Extension (Do not need for test)

x10 is the base

ASCII, Bitmapped Graphics & Vectors

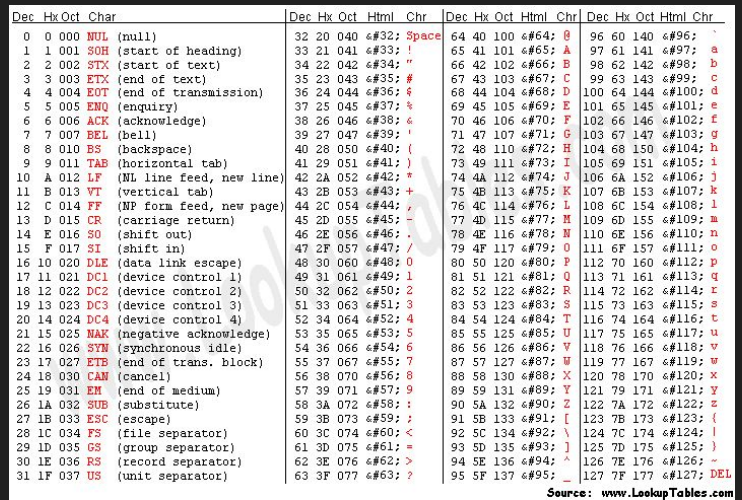
ASCII

How a standard set of characters are stored.

Text only, no information on the size, the font, the colour etc

Using 8 bits per character (Extended ASCII)

This is only 7 bits



Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Bitmapped Graphics

Bitmapped Graphics are essentially a grid of pixels, each pixel has a set colour palette.

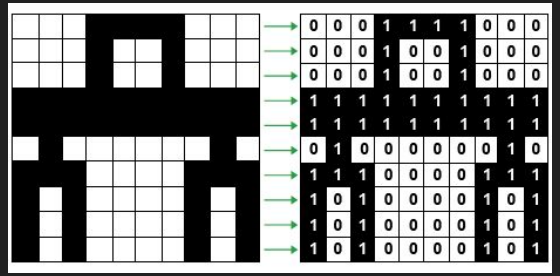
1 bit: 2 colours

2 bit: 4 colours

8 bit: 256 colours

16 bit: 65536 colours

24 bit: (True colour): 16777216 colours



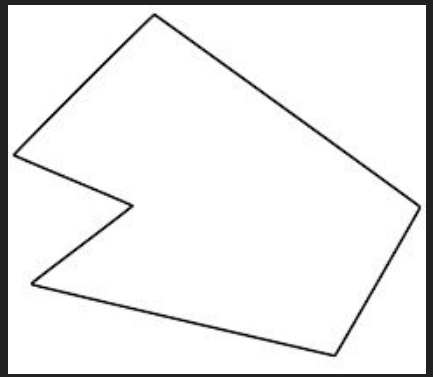
Vector

Vector Graphics store shape **objects** and their **attributes**

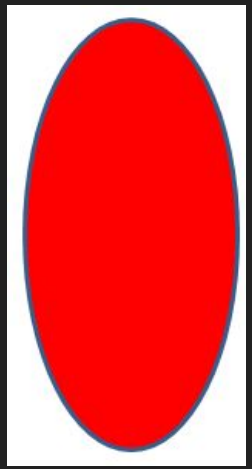
- No pixelation, no matter how magnified.
- Shapes can be layered.

Some common Vector Shapes

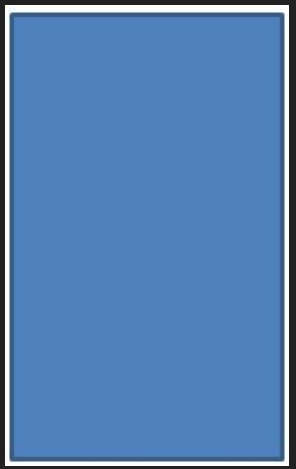
Polygon



Ellipses



Rectangle



Line

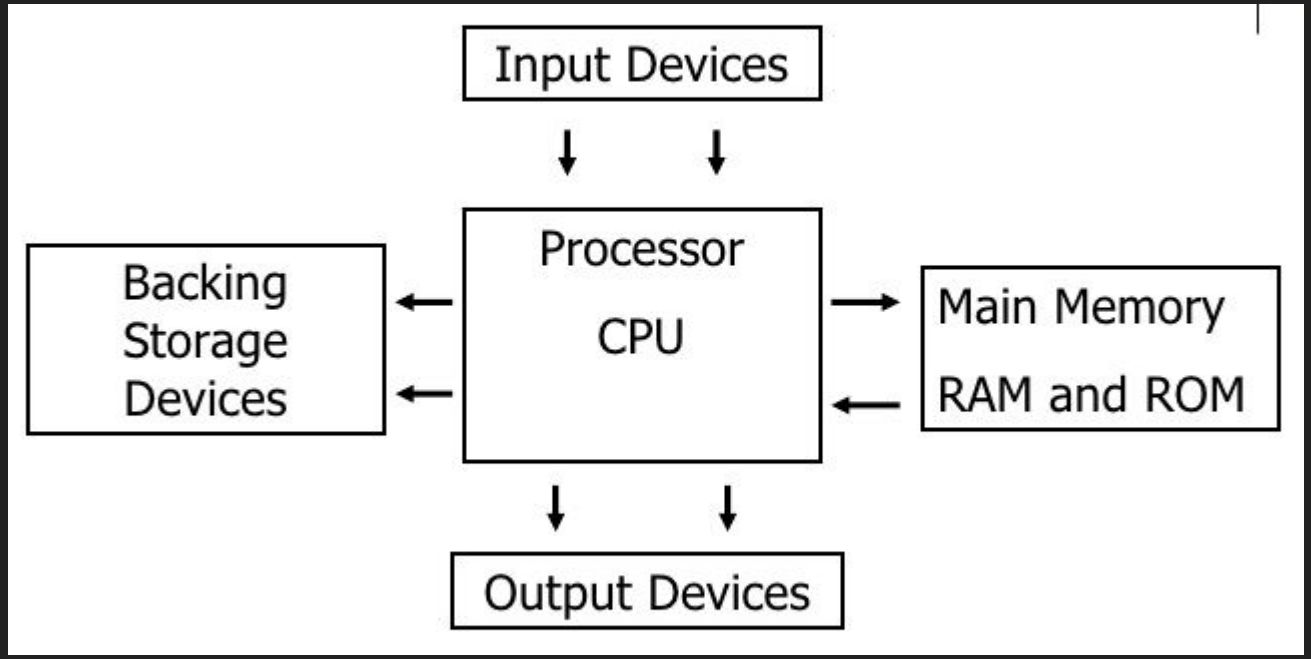
Vector Attributes

Things you can change about an object

- Fill Colour
- Line Colour
- Coordinates (Position)

Computer Architecture

Basic Computer Architecture



RAM

RAM (Or Random Access Memory) stores programs and data whilst the computer is running

- Programs loaded from HD into RAM
- RAM is faster to access than HD
- Data is lost when the PC is turned off.

Processor

The CPU (Or Central Processing Unit) does all the sorting, searching, calculating and decision making.

The **control unit** controls the order in which instructions are executed.

The ALU (Arithmetic & Logic Unit) Carries out Calculations and decisions

e.g 2 + 2

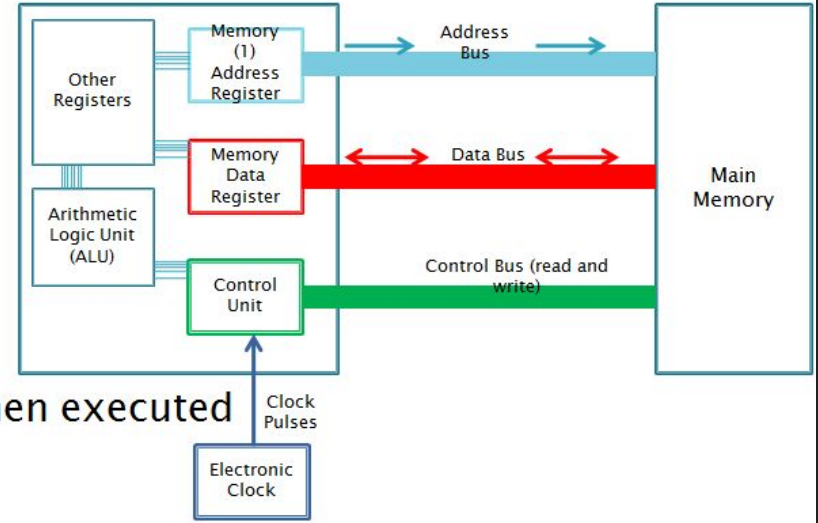
Registers are mini storage devices on the CPU. Registers usually hold

- The data being processed
- The instructions being run
- Memory addresses that need to be accessed

Fetch-Execute Cycle

Fetch Request

- Processor sets up address bus with required address
- Processor activates the read line
- Instruction transferred from memory to processor by using the data bus (fetch)



- Instruction decoded then executed

Compiler & Interpreter

Translation

Computers don’t natively support (Understand) high level languages.

The Code has to be Translated into machine code (binary) using either

- An Interpreter
- A compiler

A Compiler compiles & runs all the code in one go

- The code runs very fast
- But the Machine Code can’t be easily edited once compiled (C++ is an example)

An Interpreter Compiles & runs code one line at a time and

- It can be used to easily find bugs
- But the Program runs slowly (Python is an example)

Cyber Security

Firewalls & Encryption

A **Firewall** prevents unauthorised access **to or from a private network.**

Encryption is the process of **putting data into a code** to prevent it being seen by **unauthorized users**

Questions

1. What is computer architecture?
2. What is hardware?
3. What is software?
4. What is a device?
5. What is “software” another term for?
6. Name 5 parts of a computer?
7. In which part(s) of the computer is the calculating and decision making done?
8. What is the name given to the set of instructions that controls how a computer works?
9. What is computer memory used for? Made up of?
10. How does the processor know where to find data stored in memory?

Congratulations, you made it to the end and are now ready for your test, make sure to keep revising using these slides and good luck!