2.1.1 Computational thinking

Definition

Computational thinking is a set of problem-solving methods that involve expressing problems and their solutions in ways that a computer could also execute.

Other computational methods

Data mining:
This aims to spots trends and patterns in data.

Algorithms:
A rough list of instructions used to solve a problem.

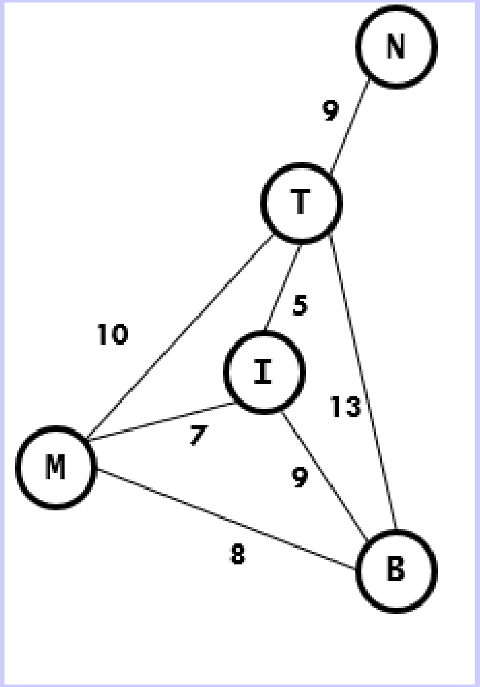
Pattern recognition:
Used to find similarities and make problems easier to solve.

Abstraction

Definition

Abstraction is the removal of unnecessary elements so that the important parts remain, thus making the problem easier to solve.

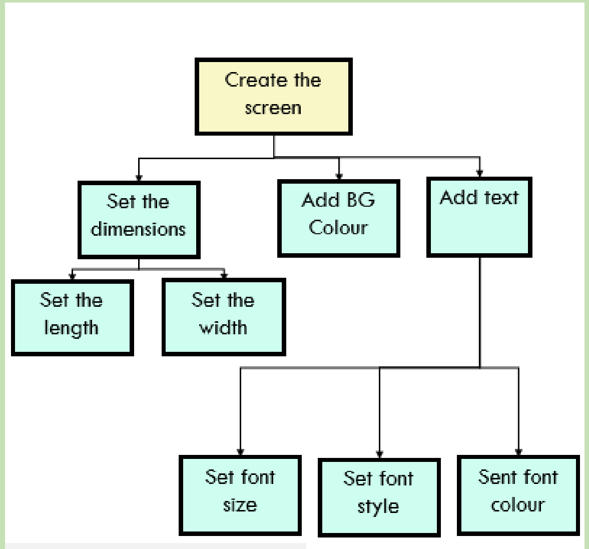
Newport
Telford
Ironbridge
Much Wenlock
Bridgnorth
Cleobury
fordshire
Wol
Kidder



Decomposition

Definition

Decomposition is the process of taking a problem and breaking it down into smaller chunks (known as sub-tasks).



2.1.2 Designing, creating and refining algorithms

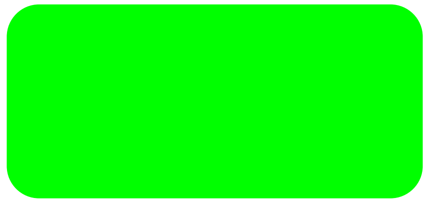
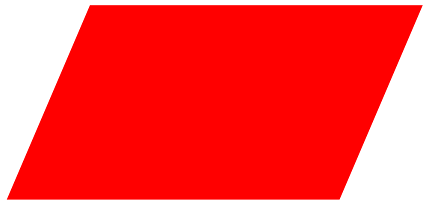
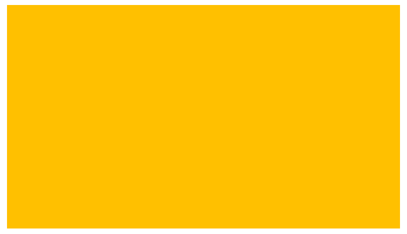
Constructing algorithms

Algorithms can be constructed in many different ways. It could be a basic list of instructions, pseudocode or as a flow chart.

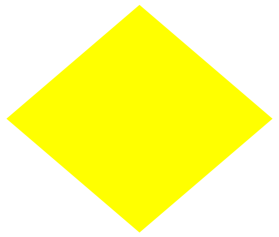
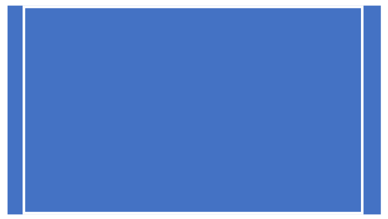
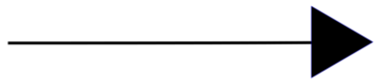
Syntax and Logic errors:

Syntax error: This error occurs when the syntax used does not meet the rules set by the language. A common example is a grammatical mistake (e.g. print spelt incorrectly)	**Logic error:** The program will appear to be working however, it might do what it's intended to do. A common example would be the use of an incorrect operator.

Flow chart symbols:

	**Start/Stop:** This signals the beginning and the end of each algorithm.
	**Input/Output:** Used if data is being inputted into the system. If any data needs to be displayed then output could be used. (e.g. print)
	**Process** This is used to process instructions. This could be used to store variables and their associated values. It could be used to process calculations.

Flow chart symbols:

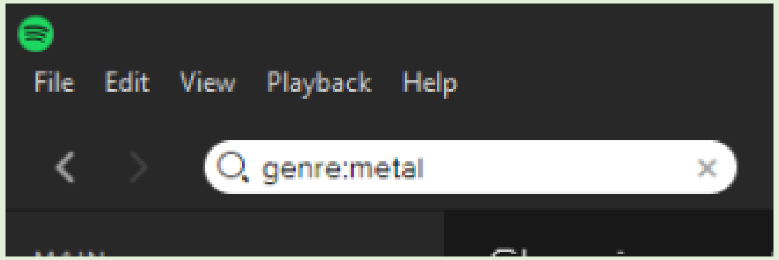
	**Decision** This is used if there is a condition which could lead to an alternative path on the flow chart. Used for Selection Statements
	**Subroutine** This is used to identify any instructions that can be/or need to be re-used over and over again. (e.g. functions and procedures.)
	**Connector** The line used to link all the symbols together. This helps to establish a logical structure to the algorithm.

2.1.3 Searching & Sorting algorithms

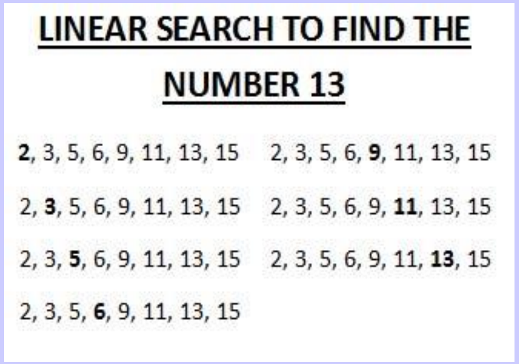
Searching algorithms

Searching Algorithms are designed to check/retrieve an element from any data structure where it is stored.

Real life examples:



Linear Search:



LINEAR SEARCH TO FIND THE NUMBER 13

2, 3, 5, 6, 9, 11, 13, 15
2, **3**, 5, 6, 9, 11, 13, 15
2, 3, **5**, 6, 9, 11, 13, 15
2, 3, 5, **6**, 9, 11, 13, 15
2, 3, 5, 6, **9**, 11, 13, 15
2, 3, 5, 6, 9, **11**, 13, 15
2, 3, 5, 6, 9, 11, **13**, 15

Pro

Quick if the data required is at the beginning of the list.

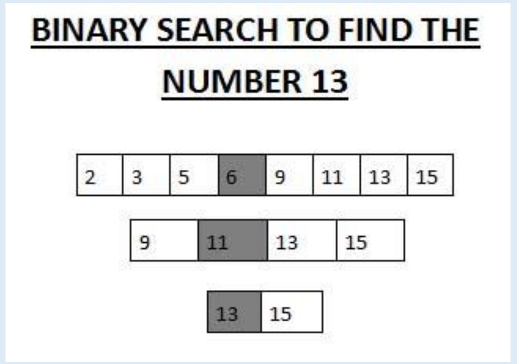
Con

As the list grows, it may become inefficient.

Stages:

- Examine the first value held in the list.
- Check to see if the value at that position matches the value searched for.
- If it matches, the value is found.
- If not, move to the next item in the list and repeat steps 1-3 until found.
- If all the items have been checked and no match is found, send a message.

Binary Search:



BINARY SEARCH TO FIND THE NUMBER 13

Pro

The 'divide and conquer' approach means it will remove half of the result after each search.

Con

Will not always outperform a linear search.

Stages:

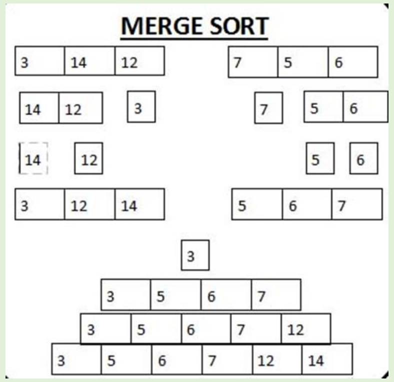
- Start by setting the counter to the middle position in the list to find the media value.
- If the value held there is a match, the search ends.
- If the value at the midpoint is less than the value to be found, the list is divided in half. The lower half of the list is ignored and the search keeps to the upper half of the list.
- Otherwise, if the value at the midpoint is greater than the value to be found, the upper half of the list is ignored and the search keeps to the lower half of the list.
- Repeat steps 1 to 3 until found.

2.1.3 Searching & Sorting algorithms

Sorting algorithms

Sorting Algorithms are designed to take an unsorted list and re-arrange it into a logical order (ascending or descending)

Merge Sort:



Stages:

- Split a list of items until they can't be split any longer.
- At this point, the items are merged back together, but this time in the correct order.

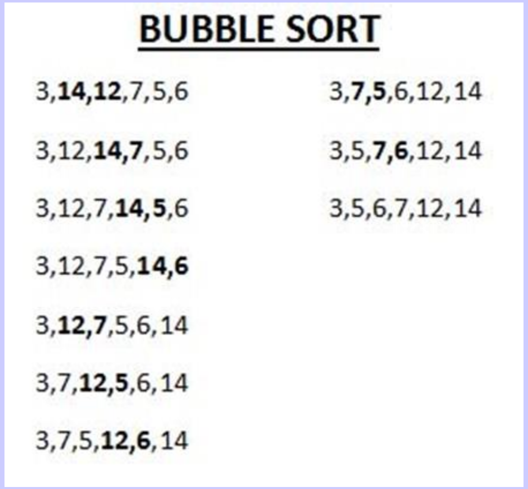
Pro

It is quicker for larger lists as it doesn't go through the whole list several times.

Con

Uses more memory space to store the sub elements of the initial split list.

Bubble Sort:



BUBBLE SORT

3,**14,12**,7,5,6

3,12,**14,7**,5,6

3,12,7,**14,5**,6

3,12,7,5,**14,6**

3,**12,7**,5,6,14

3,7,**12,5**,6,14

3,7,5,**12,6**,14

3,**7,5**,6,12,14

3,5,**7,6**,12,14

3,5,6,7,12,14

Stages:

- It will go through the array by checking a value and it's neighbouring value.
- If the two numbers are not sorted then the numbers are swapped.
- This process continues until the entire list is sorted.

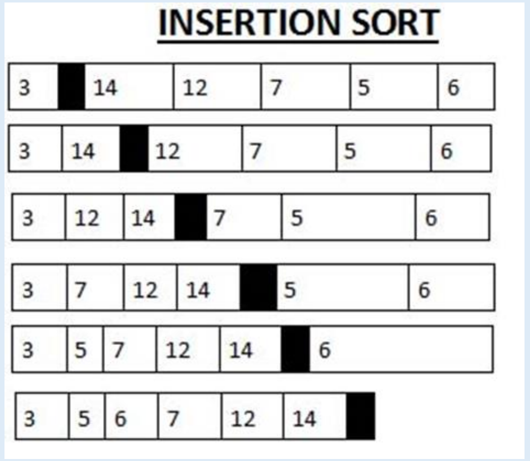
Pro

Easy to write and code.

Con

Time consuming, it could take a while to sort all the data.

Insertion Sort:



Stages:

- Insertion sort involves going through a pile, taking one item, comparing it to the first, swapping places.
- If one item is larger than another and continuing this process until the minimum item is in the correct location.

Pro

Efficient for sorting of small data/data that is almost sorted

Con

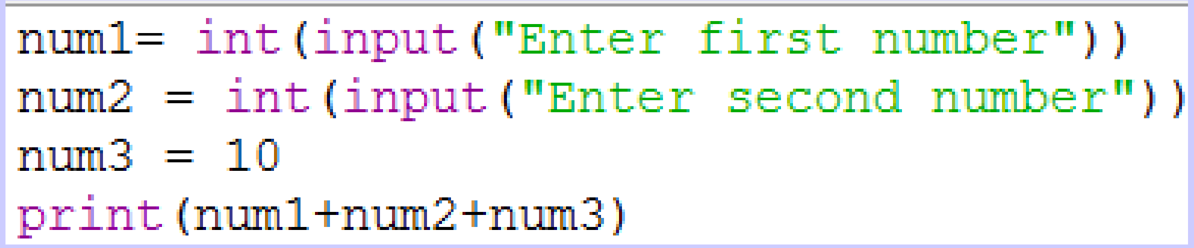
Less efficient for sorting of large data

2.2. Programming fundamentals

2.2.1 The use of variables, constants, operators, inputs, outputs and assignments.

An assignment operator is used to assign a value to a variable or constant. However, there are a range of operators that a serve a different purpose.

Variables and Constants:



```
num1= int(input("Enter first number"))
num2 = int(input("Enter second number"))
num3 = 10
print(num1+num2+num3)
```

Variable

A named storage location that is used to store a value that can change at any point during the program.
For example, in the code above, num1 and num2 are variables because the input could be different every time the program is run.

Constant

A named storage location that is used to store a value that cannot change automatically and will remain the same each time to program is run. It can only change if the user manually changes the value.
For example, in the code above num3 is a constant.

Comparison Operators:

Operator	Meaning
==	Equal to
!=	Not equal to
>	Greater than
<	Less than
=>	Equal or more than
<=	Equal or less than

Logical Operators:

- AND
- OR
- NOT

Arithmetic Operators:

Operator	Python representation	Meaning
+	+	Addition
-	-	Subtraction
*	*	Multiplication
/	/	Division
DIV	//	Floor Division
^	**	Exponentiation (Powers)
MOD	%	Modulus (Remainder)

2.2 Programming Fundamentals

2.2.1 Programming Constructs

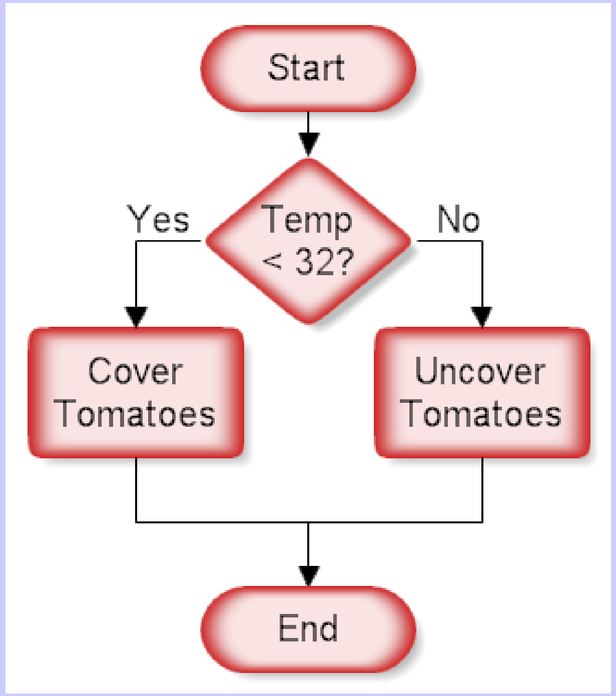
Programming constructs are the building blocks used for any program developed and they are: Sequence, Selection and Iteration.

Sequence

```
num1= int(input("Enter first number"))
num2 = int(input("Enter second number"))
num3 = 10
print(num1+num2+num3)
```

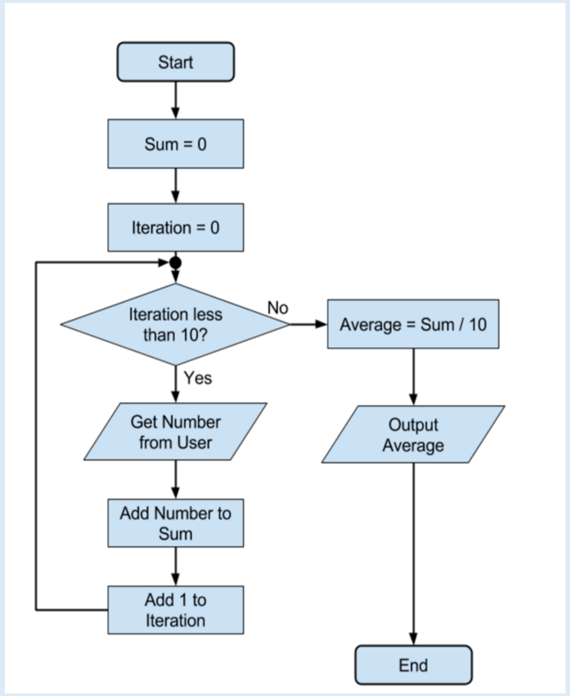
Definition:
A sequence refers to a logical order of items. In the context of programming, algorithms always use a sequence because it's written line by line.

Selection



Definition
Selection is the process in which an outcome depends on whether a certain condition is met. In programming, selection (IF) statements are commonly used for this.

Iteration



Definition
Iteration is the process of repeating steps. In programming, there are two common types used: FOR Loops and WHILE Loops.

A FOR loop is a counter-controlled loop. This means code will only repeat a certain number of times.
A WHILE loop is a condition-controlled loop. This means the code will continue to repeat until a certain condition is met.

2.2 Programming Fundamentals

2.2.2 Data types

This is a particular kind of data item, as defined by the values it can take, the programming language used, or the operations that can be performed on it.

Data types:

Integer
Used to represent a whole number.

Float/Real
Used to represent real values so this could be numbers that include a decimal value.

String
A collection of alphanumeric characters enclosed in quotation marks.

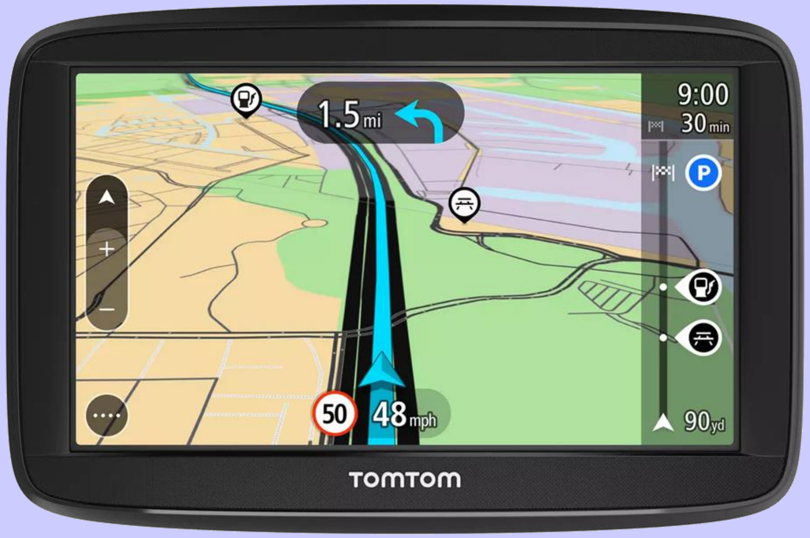
Character
Represents a single character from a string..

Boolean
An outcome represented by one of two states (TRUE/FALSE)

Context:

Float/Real
Specific distance in miles (e.g. 1.5 miles)

String:
Mph for Miles per hour



Integer:
Speed and Yards as a whole number

Character:
The letter P for Parking

What is casting?

Casting is when you convert from one data type to another. In the example below the input function would be set as a default to a string. By adding int in front of the input, we're telling the program that we would like to user to enter an integer.

```
num1= int(input("Enter first number"))
num2 = int(input("Enter second number"))
num3 = 10
print(num1+num2+num3)
```

2.2. Programming fundamentals

2.2.3 String Manipulation

A string is a sequence of characters that contain letters, numbers and symbols. It is commonly enclosed in quotation marks however, there are techniques we can use to change the way the string looks.

Substrings:

```
firstname = "Donald"
surname = "Trump"
print(firstname[:1])
print(surname[:2])
```

```
Don
T
```

Description:
A substring is used to take a portion of the string.

OCR Exam Reference Language Equivalent:

- firstname = “Donald”
- surname = “Trump”
- print(firstname.substring(0,1))
- print(surname.substring(0,2))

How to check the length of a string:

```
print(firstname, "is", len(firstname),"characters long")
print(surname, "is", len(surname),"characters long")
```

Description:
The len function will check the length of a string. In the OCR Exam Reference Language it will be referred to as *length*.

Upper and Lower functions:

```
firstname = "Donald"
surname = "Trump"
print(firstname[:3])
print(surname[:2].upper())
```

```
Don
TR
```

Description:
The upper function which switch the string to an uppercase letter. Lower can be used to make the string lowercase.

OCR Exam Reference Language Equivalent:

- firstname = “Donald”
- surname = “Trump”
- print(firstname.substring(0,3))
- print(surname.substring(0,2)).upper

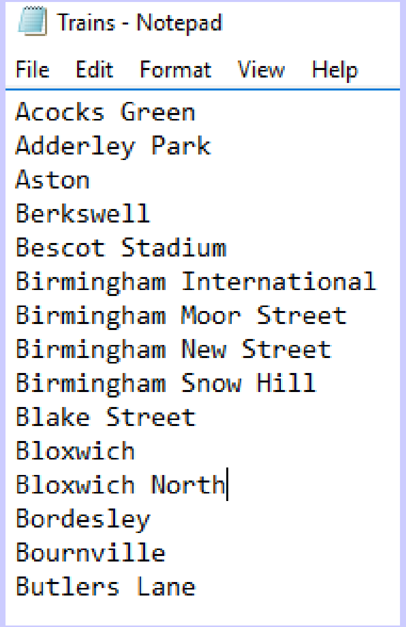
2.2. Programming fundamentals

2.2.3 File handling

Notepad creates, reads, and writes text files. Most other text editor programs and program editor programs use text files. Many application programs are written to process input data from a text file and to write their output to a text file.

Read

```
f = open("Trains.txt","r")
print(f.readline())
f.close()
```



```
Acocks Green
Adderley Park
Aston
Berkswell
Bescot Stadium
Birmingham International
Birmingham Moor Street
Birmingham New Street
Birmingham Snow Hill
Blake Street
Bloxwich
Bloxwich North
Bordesley
Bournville
Butlers Lane
```

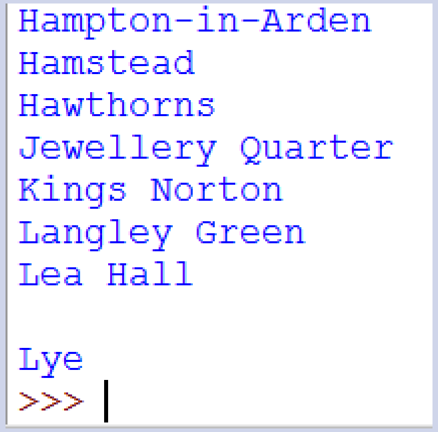
Description:
The ‘r’ function is the command for reading the contents of a text file.

OCR Exam Reference Language Equivalent:

- f = open (“Trains.txt”)
- print(f.readLine())
- f.close()

Append

```
f = open("Trains.txt", "a")
f.write("Lye")
f.close()
```



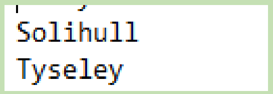
```
Hampton-in-Arden
Hamstead
Hawthorns
Jewellery Quarter
Kings Norton
Langley Green
Lea Hall

Lye
>>> 
```

Description:
The ‘a’ function is the command used for updating the contents of an existing file.

Difference between “w” and “a”
If you used “w” to update an existing file, it could overwrite the original content..

Write:

```
f = open("Trains2.txt","w")
f.write("Solihull\n")
f.write("Tyseley\n")
f.close()
```



```
Solihull
Tyseley
```

Description:
The ‘w’ function is the command used for writing to a new text file.

2.2. Programming fundamentals

2.2.3 Subprograms

In computer programming, a subroutine is a sequence of program instructions that performs a specific task, packaged as a unit. This unit can then be used in programs wherever that particular task should be performed.

Functions

```
def vat(Price):
    Total= Price * 1.2
    return Total
```

Description:
A function is a subprogram that can return a result based on it's defined parameters.

OCR Exam Reference Language Equivalent:

- Function vat (Price)
    - Total = Price * 1.2
    - return Total
- end function

What is a parameter?
A parameter is a special variable used within a function to return a result.

Benefits of using subprograms:

Usability:
It creates re-usable code.

Debugging:
The problem maybe in the subprogram itself.

Testing:
Only need to test the subprogram

Efficiency:
Less code required as it can be reused.

Delegation:
Easy split between developers

Decomposition:
Breaks the problem down.

Procedure

```
def message():
    print("Hello world")
    print("Welcome")


message()
```

Description:
A procedure is a subprogram that will not return a result, but information can still be passed through it.

OCR Exam Reference Language Equivalent:

- Procedure message()
    - print(“Hello world”)
    - print(“Welcome”)
- end procedure

2.2. Programming fundamentals

2.2.3 Arrays

In computer science, a data structure is a data organization, management, and storage format that enables efficient access and modification. More precisely, a data structure is a collection of data values, the relationships among them, and the functions or operations that can be applied to the data.

Reasons why arrays are used:

Reasons:

- An array allows multiple items of data to be stored under one identifier.
- It can be stored in a table like structure and reduces the number of variables required as you can set above.
- This creates a more elegant and efficient solution.
- In Python these are referred to as lists.
- There are many data structures used such as: linked lists, hash tables, trees, graphs etc…

1D Array:

Names	Sam	Jessica	David	Gemma	Dom

```
names = ["Sam","Jessica","David","Gemma","Dom"]
```

Description:
A 1D array is a simple data structure that stores a collection of data in a constant block of memory. It's known as a single dimensional array which stores data as a list.

2D Array:

	0	1	2	3	4
0	Sam	Jessica	David	Gemma	Dom
1	M	F	M	F	M

```
p = [["Sam","M"],["Jessica","F"],["David","M"],["Gemma","F"],["Dom","M"]]
```

Description:
A 2D array is a type of array that stores multiple data elements in a table like format with a number of rows and columns. This type of array is commonly referred to as multi-dimensional.

2.2 Programming Fundamentals

2.2.3 SQL & Data Structures

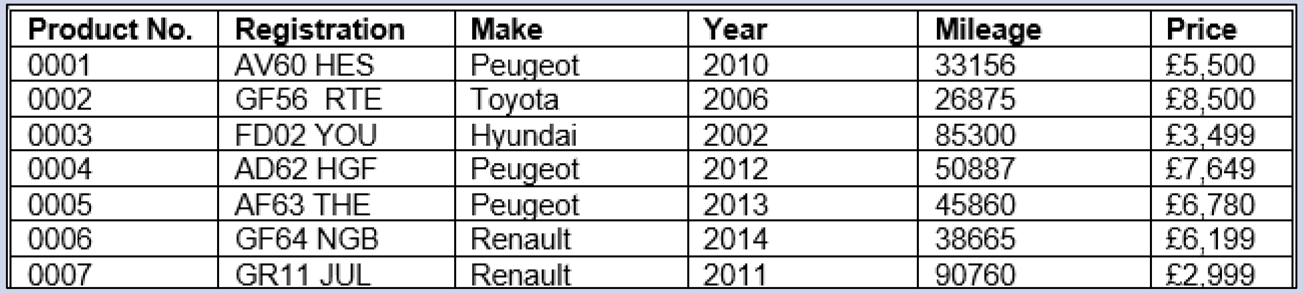
It stands for Structured Query Language and its code uses to create, access and maintain databases.

Did you know?

You can use wildcard characters to run a more efficient query. For example the * can be used to select all fields.

SELECT * FROM Cars WHERE Mileage > 40000

Using SQL commands in a database:



Product No.	Registration	Make	Year	Mileage	Price
0001	AV60 HES	Peugeot	2010	33156	£5,500
0002	GF56 RTE	Toyota	2006	26875	£8,500
0003	FD02 YOU	Hyundai	2002	85300	£3,499
0004	AD62 HGF	Peugeot	2012	50887	£7,649
0005	AF63 THE	Peugeot	2013	45860	£6,780
0006	GF64 NGB	Renault	2014	38665	£6,199
0007	GR11 JUL	Renault	2011	90760	£2,999

SQL Commands

SELECT
This command will request (Select) fields that they want to appear in the final results.

FROM
This command means the source in which the information came from. So this would be the name of the database.

WHERE
This command will request specific information from the selected fields. This makes the search more refined

What is a Database?

Definition:
A database allows you to store records that can be accessed, modified and deleted.

Product code	Brand	Product type	Description	Colour	Price	Ratings	Stock
2230013	Amazon	All-new Echo Dot (4th gen)	Smart speaker with Alexa	Charcoal	49.99	136	45

Field
A field is a category of data. So each heading in the grey box is classified as a field.

Primary Key
This is a field that will uniquely identify a record and removing any duplicates.

Record
A record is an individual set of data. So in this case, the Amazon Echo Dot is a record.

2.3 Producing robust programs

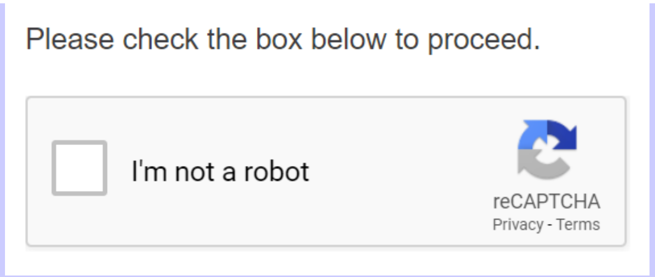
2.3.1 Authentication

Authentication is the process of determining the identity of the user. Identify what each of these methods are and describe how they work.

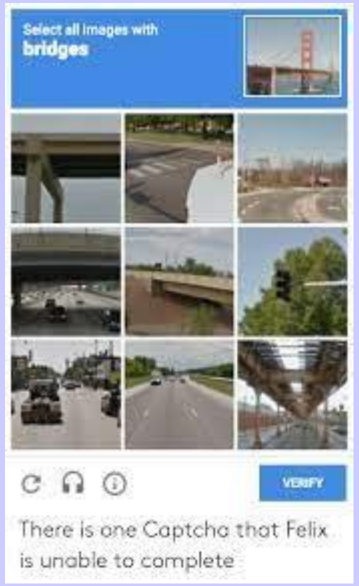
Exam tip:

Other examples of authentication include: PIN, Biometrics (Facial/Thumb/Eye recognition), Passport No. and swipe cards. Anything that uniquely identifies you.

CAPTCHA


Please check the box below to proceed.

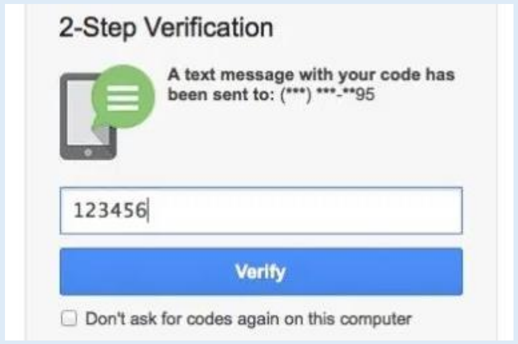
I'm not a robot



There is one Captcha that Felix is unable to complete

Description:
This is s a security application that determines whether data is being entered by humans or bots. They use images because bots cannot read images.

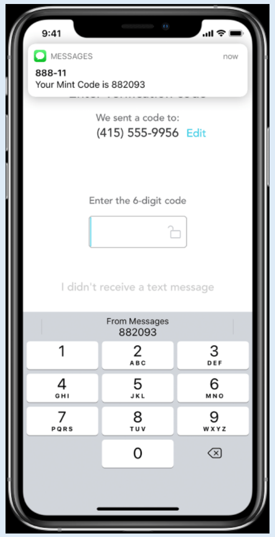
2-Step Verification


2-Step Verification

A text message with your code has been sent to: (***) ***-**95

123456

Don't ask for codes again on this computer



888-11
Your Mint Code is 882093

We sent a code to:
(415) 555-9956 Edit

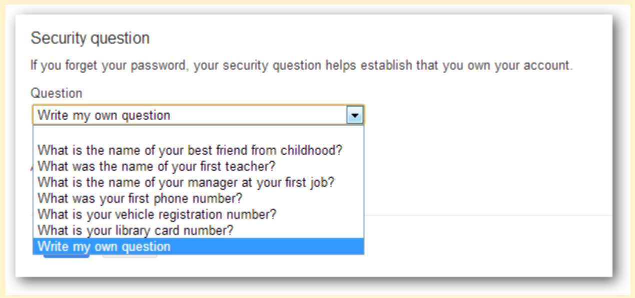
Enter the 6-digit code

I didn't receive a text message

From Messages
882093

Description:
This can be used to confirm a new account or recover an old account. The provider will send a verification code to the phone number used to register the account.

Security Question


Security question

If you forget your password, your security question helps establish that you own your account.

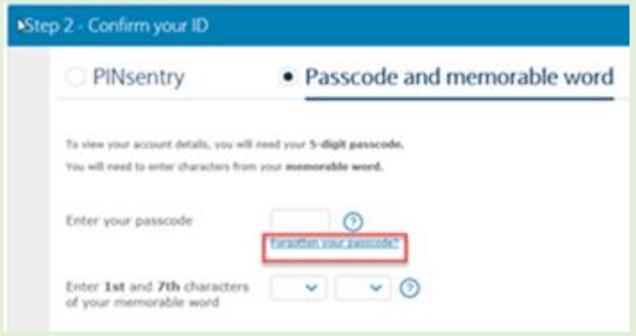
Question

Write my own question

What is the name of your best friend from childhood?
What was the name of your first teacher?
What is the name of your manager at your first job?
What was your first phone number?
What is your vehicle registration number?
What is your library card number?
Write my own question

Description:
This might be used to confirm registration or used if user has forgotten their password.

Specific characters:


Step 2 - Confirm your ID

PINsentry [✓ Passcode and memorable word]

To view your account details, you will need your **5-digit passcode.**
You will need to enter characters from your **memorable word.**

Enter your passcode

Forgotten your passcode?

Enter **1st** and **7th** characters of your memorable word

Description:
This is where users enter certain characters from their password. (e.g. 2nd and 4th)

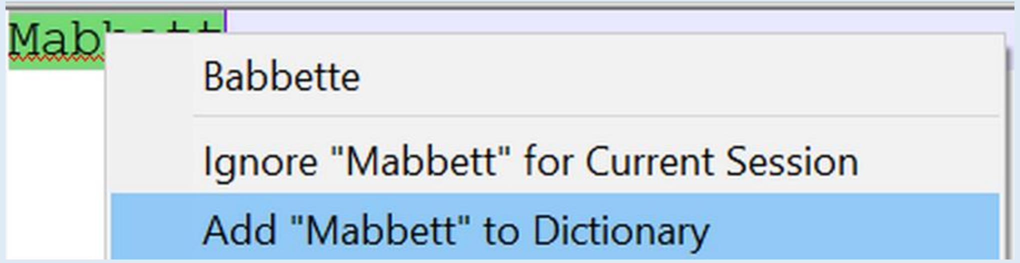
2.3 Producing robust programs

2.3.1 Validation

This is a process that checks data entered is sensible, reasonable and appropriate to be processed by the program. Match up the validation methods with the correct description.

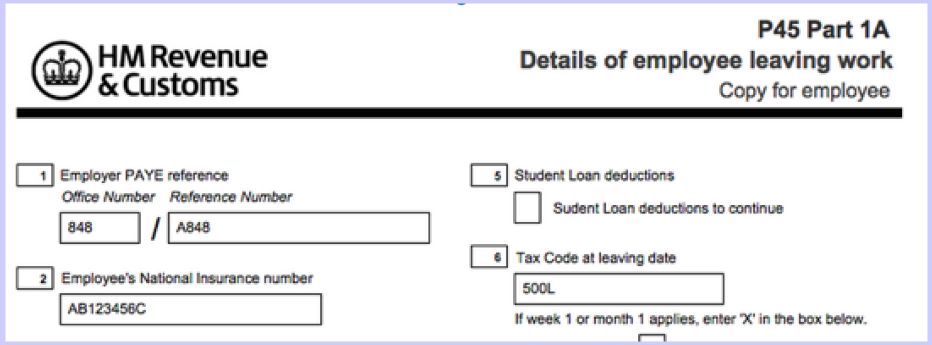
Spell check:

Description:
Used to check the quality of written communication in a document.


Mabbett
Babbette
Ignore "Mabbett" for Current Session
Add "Mabbett" to Dictionary

Format check:

Description:
Used to check data entered is the appropriate data type (e.g. entering a letter when it says enter a letter)



HM Revenue & Customs

P45 Part 1A
Details of employee leaving work
Copy for employee

1 Employer PAYE reference	5 Student Loan deductions
Office Number 848 / Reference Number A848	Sudent Loan deductions to continue
2 Employee's National Insurance number	6 Tax Code at leaving date
AB123456C	500L
	If week 1 or month 1 applies, enter 'X' in the box below.

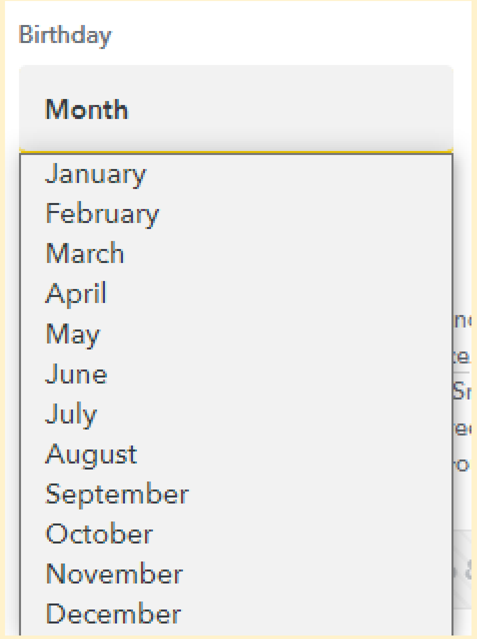
Example:
A national insurance number is a combination of numbers and letters. So the format will be LL NN NN NN L

```
while True:
    number = input("Enter an integer")
    if number.isdigit():
        number = int(number)
        break
    else:
        print("That's not an integer")
```

Coded example

Lookup table:

Description:
Used to find the date and time on an online entry form.



Example:
In this example, there is a pre-populated list of months for the user to select. In some cases, you may need to enter data by selecting dates from a calendar.

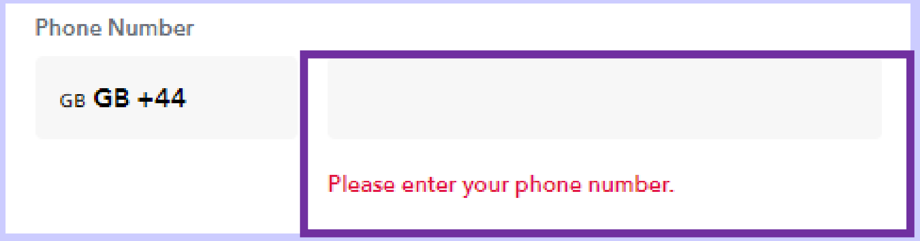
2.3 Producing robust programs

2.3.1 Validation

This is a process that checks data entered is sensible, reasonable and appropriate to be processed by the program. Match up the validation methods with the correct description.

Presence check:

Description:
Used to check that a field has not been left blank.


Phone Number

GB GB +44

Please enter your phone number.

Example:
In this example, the sign up button is greyed out because there is a required field missing. Some text in red has been added to indicate this to the user.

```
Name = ""
print ("Please enter your Name")
Name = input()
while Name == "":
  print ("Sorry, your name must be entered, try again ")
  Name = input()
print ("Thank you " + Name + " Your comment has been posted")
```

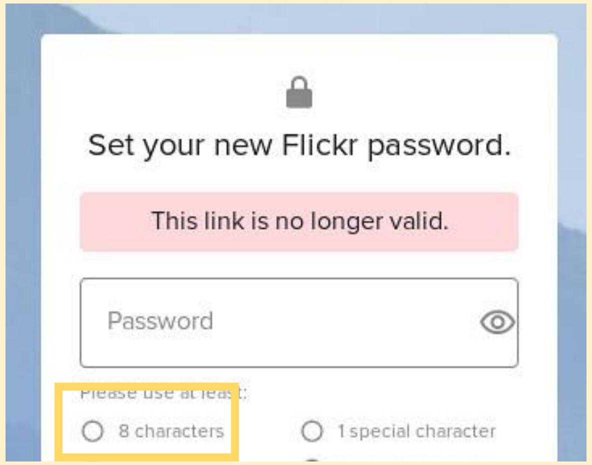
Range check:

```
Please enter a number between 1 and 1011
Please enter a number between 1 and 104
Thank you
```

```
while True:
    option = int(input("Option: "))
    if option >= 1 and option <=4:
        print("You have chosen option",option)
        break
```

Description:
Used to check whether data entered fits within a set criteria.

Length check:


Set your new Flickr password.

This link is no longer valid.

Password

Please use at least:

8 characters

1 special character

Description:
Used to check if the data enter has sufficient amount of characters.

Example:
In this example, to register with this provider. The password needs to be at least 8 characters in length.

```
while True:
    password = input("Enter a new password: ")
    if len(password) < 10:
        print("Password must have more than ten characters")
    else:
        print("Password length correct")
```

2.3 Producing robust programs

2.3.1 Maintainability

It is important that programmers make sure their code is maintainable to ensure it's structured correctly, that they've used meaningful variable names and comments to explain difficult parts of the code. This is extremely beneficial for developers who are collaborating on one project.

Indentation:

```
while guess != Answer: #While user guesses incorrectly.
    if guess < Answer: #Indicates whether they are too high or low.
        print("Too low")
    elif guess > Answer:
        print("Too high")
    elif guess == Answer:
        print("Correct")
    else:
        print("Out of range")
    score = score + 1 #Adds to guesses
    guess = int(input("Please enter a number between 1 and 100"))
```

Description:
Indentation makes it clear to the developer where statements appear within a selection statement or iteration.
It is built into Python's syntax so incorrect use of indentation will lead to a syntax error.

Naming conventions:

```
FirstNum = int(input("Please enter a number"))
```

Description:
CamelCase is a common technique when naming variables. This is because variables cannot contain any spaces which makes the variable stand out.

Tip:

You should always use meaningful variable names for all variables.

Comments:

```
#Guess the number challenge
import random

Answer = random.randint(1,100)#Random number between 1 and a 100.

score = 1 # Record number of guesses
guess = int(input("Please enter a number between 1 and 100"))

while guess != Answer: #While user guesses incorrectly.
    if guess < Answer: #Indicates whether they are too high or low.
```

Description:
Comments can be used to say who wrote the program, the purpose of the program and to explain any difficult bits of code work. This is useful for developers who are collaborating on the same project.

2.3 Producing robust programs

2.3.1 Testing

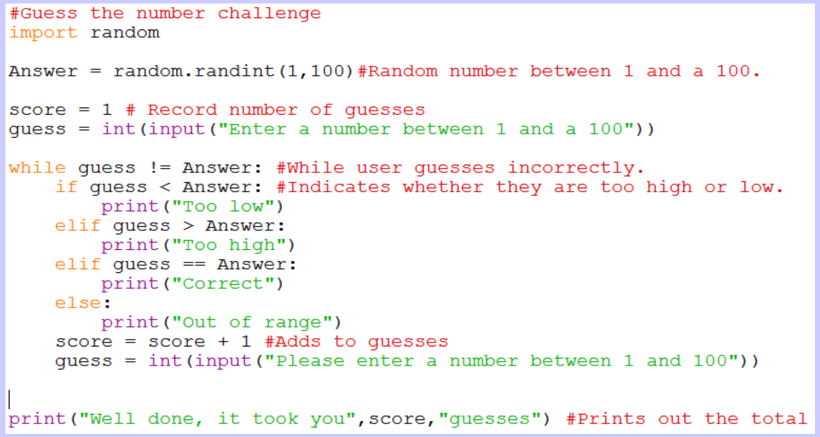
In computer hardware and software development, testing is used at key intervals to determine whether the program meets the needs of the client (end user)

Example test table

Test No.	Description/ Type of test	Expected outcome	Actual outcome	Pass/Fail	Remedial action?

Test data:

Boundary (Valid extreme)
Testing inside the boundary. What will just be accepted? In this example, a valid extreme test would be 1 or 100.



```
#Guess the number challenge
import random

Answer = random.randint(1,100)#Random number between 1 and a 100.

score = 1 # Record number of guesses
guess = int(input("Enter a number between 1 and a 100"))

while guess != Answer: #While user guesses incorrectly.
    if guess < Answer: #Indicates whether they are too high or low.
        print("Too low")
    elif guess > Answer:
        print("Too high")
    elif guess == Answer:
        print("Correct")
    else:
        print("Out of range")
    score = score + 1 #Adds to guesses
    guess = int(input("Please enter a number between 1 and 100"))


print("Well done, it took you",score,"guesses") #Prints out the total
```

Valid:
Testing data that should definitely work or should be accepted by the program. Any number between 2 and 99 would be a valid test in this example.

Boundary (Invalid extreme)
Testing outside the boundary. What will just about not be accepted?
In this example it would be 0 and 101.

Types of testing:
Iterative testing: This is when testing takes place during the development of the program. For example, this might involve testing one block of code to make sure it works before moving onto the next one.

Final testing: This takes place when the development of the code is complete, all tests will then be conducted.

Erroneous:
Purposely testing something extreme that wouldn't work. In this example, using a letter instead of a number.

Why test?

Purpose:
To check the program works properly and identify any errors so they can be fixed. It's important businesses test their websites and apps to make sure they function the way they would expect it to.

2.4 Boolean Logic

2.4.1 Boolean Logic

Logic is concerned with forms of reasoning. The study of logic is essential for students who undertake a wide range of subjects, not only Computer Science.

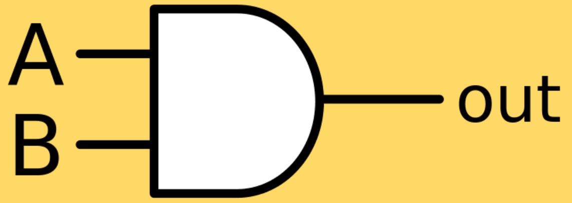
Did you know?

Logic gates are made up of a series of transistors which acts as switches. These switches are either ON (1) or OFF (0)) Transistors are tiny electronic components found on the CPU.

Revision tip:

It is important to learn each notation and what they represent just incase you need to interpret a Boolean expression in the exam.

AND



Description:

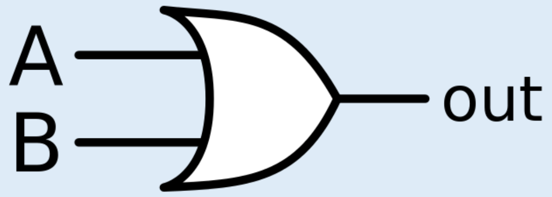
- This is an AND gate.
- Both inputs need to be positive to achieve the same output.
- The notation used to represent AND is ∧

Boolean expression:

- Q = (A AND B)
- Q = (A ∧ B)

A	B	Q
1	1	1
1	0	0
0	1	0
0	0	0

OR



Description:

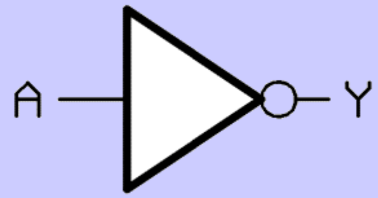
- This is an OR gate.
- Only one input needs to be positive to achieve a positive output.
- The notation used to represent OR is V

Boolean expression:

- OUT = (A OR B)
- OUT = (A ∨ B)

A	B	Q
1	1	1
1	0	1
0	1	1
0	0	0

NOT



Description:

- This is a NOT gate.
- The output will be opposite to an input, a bit like a light switch.
- The notation used to represent NOT is ¬

Boolean expression:

- Y = NOT A
- = ¬ A

A	Y
1	0
0	1

2.5 – Programming languages and Integrated Development Environments

2.5.1 Languages

Examples of high-level programming languages in active use today include Python, Visual Basic, Delphi, Perl, PHP, ECMAScript, Ruby, C#, Java and many others.
needs.

Low-level languages

Machine code
A language directly understood by the CPU because they use a set of transistors that consist of one of two states, in the form of 0's and 1's.

Assembly language
A language that uses mnemonics to represent commands used by specific types of processor. It must be converted to machine code using an assembler.

High-level languages

Description:
The part of software that most computer users don't ever see; it's the code computer programmers can manipulate to change how a piece of software works.

Interpreter	Compiler
Translates and execute one line of source code at a time.	Translates all of the code in one batch, instead of line by line.
Interpreters are easy to use for beginners which makes it a real advantage.	Translated before executing and packaged into a machine code file, usually in the form of an exe (executable file)
If a line contains an error – then the interpreter will stop at that line and go now further.	Instead of stopping at the first error, it will generate a list of errors (if any) all at once.
Interpreted code must be translated each time it's run	The process of compilation is relatively complicated. It spends a lot of time analysing and processing the program.
It is easy to access source code because it's converted into machine code when the program is run.	Compiled code will run faster.

2.5 – Programming languages and Integrated Development

2.5.2 The Integrated Development Environment (IDE)

IDE is a tool built within software that allows you to write programs using code.

Additional IDE features:

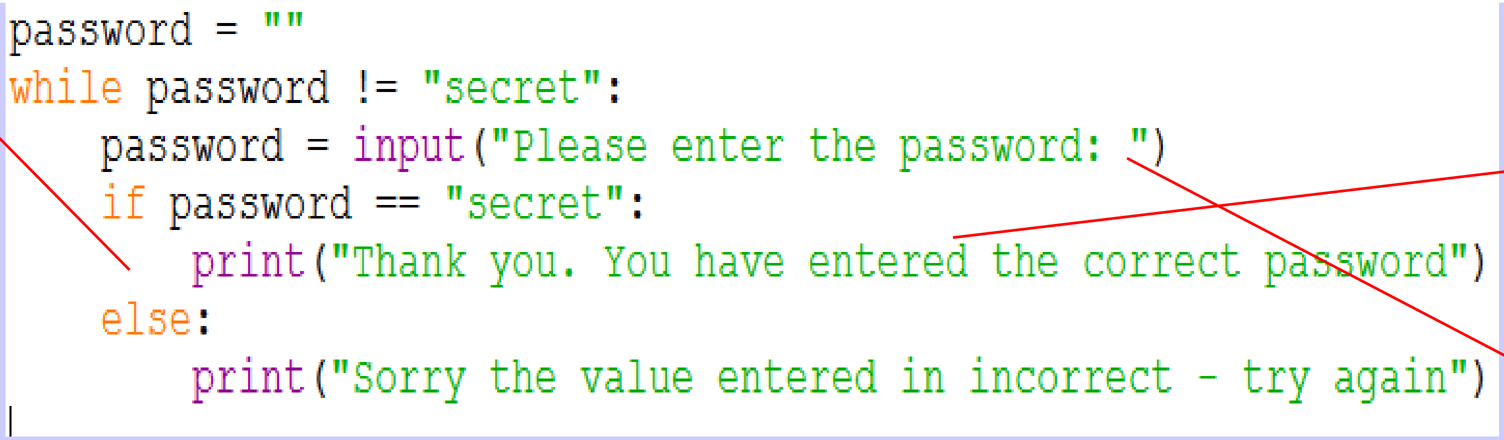
Auto-completion
A feature that automatically detects recognised syntax while you type code.

Libraries
These are in-built functions that can be immediately accessed and added to code.

Run-time environment
As soon as a software program is executed, it is in a runtime state.

IDE Features:

Auto-indentation
Automatically indents the next line if required.

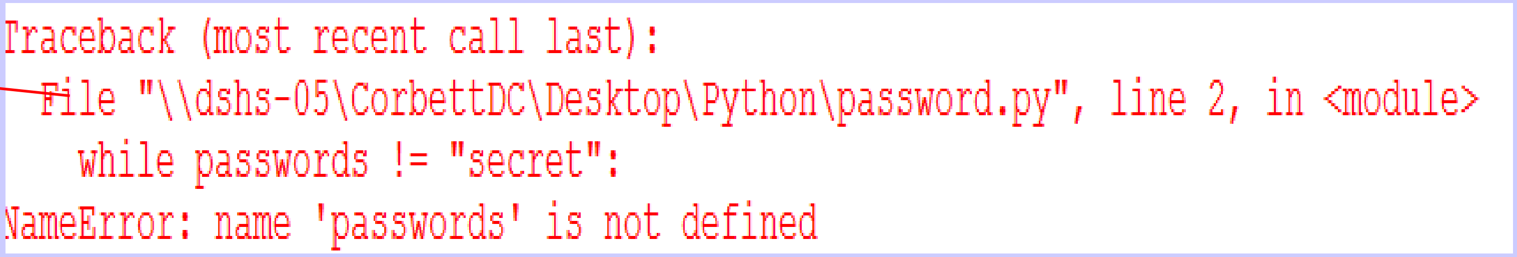


```
password = ""
while password != "secret":
    password = input("Please enter the password: ")
    if password == "secret":
        print("Thank you. You have entered the correct password")
    else:
        print("Sorry the value entered in incorrect - try again")
```

Syntax highlighter
Displays source code in different colours so certain commands in orange, functions in purple etc..

Bracket matching
It highlights matching sets to identify whether you've used the correct amount of brackets.

Debugging tool/Error diagnostics
Allows code to be inspected for errors with suggestions on where the problem lies.



```
Traceback (most recent call last):
  File "\\dshs-05\CorbettDC\Desktop\Python\password.py", line 2, in <module>
    while passwords != "secret":
NameError: name 'passwords' is not defined
```

2.2 Programming Fundamentals

2.2.3 Random number generation

The random module is a built-in module to generate the random variables.

Random functions:

```
import random

x = random.randrange(1,10)
print (x)
```

```
import random

x = random.choice("Hello")
print (x)
```

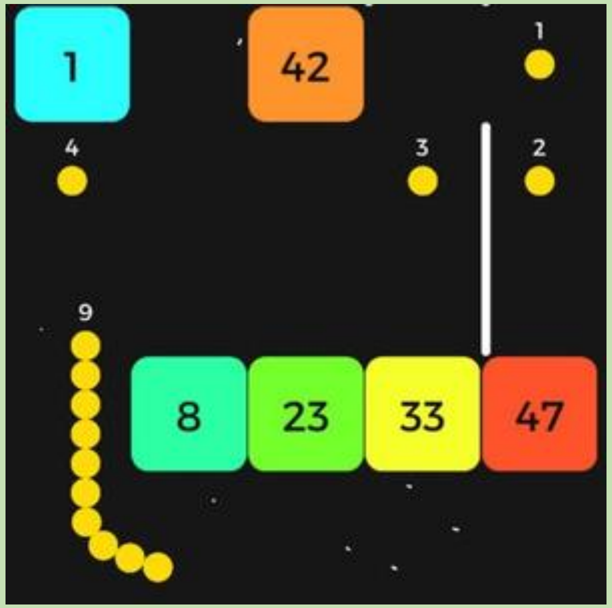
Random module

```
#Guess the number challenge
import random

Answer = random.randint(1,100)#Random number between 1 and a 100.

score = 1 # Record number of guesses
guess = int(input("Enter a number between 1 and a 100"))

while guess != Answer: #While user guesses incorrectly.
    if guess < Answer: #Indicates whether they are too high or low.
        print("Too low")
    elif guess > Answer:
        print("Too high")
    elif guess == Answer:
        print("Correct")
    else:
        print("Out of range")
    score = score + 1 #Adds to guesses
    guess = int(input("Please enter a number between 1 and 100"))

|
print("Well done, it took you",score,"guesses") #Prints out the total
```

Example

In this example the random module has been imported.
The random integer function has been used to generate a random integer between a specified range.

Context



Example

In this game Snake vs Block there are various aspects of the game that have been randomised such as:

- The numbers in the blocks change.
- The blocks change position
- The number of points scored changes.

Tip:

The random module is an in-built library that can be found in Python. Remember you must import the random module into your script if you want to use it.