**LESSON ACTIVITIES FOR**

**Introduction and Block 1: Problem Solving & Programming**

**Lesson 1 activities**

**Activity 1.1**

Write a program to print out “Hello World”.

**Activity 1.2**

Write a program that writes your name on the screen.

**Activity 1.3**

Copy and run the following program.

print("hello " \* 10)

Write a program which prints your name 6 times.

**Lesson 2 activities**

**Activity 2.1**

Are the following statements true or false?

|  |  |
| --- | --- |
| Python was released in 2010 | True or false? |
| Python was named after the TV series “Monty Python’s Flying Circus” | True or false? |
| Python is proprietary software which is expensive to buy | True or false? |
| Python was written by Bill Gates | True or false? |
| YouTube is written in Python | True or false? |

**Activity 2.2**

Make a note of what these useful keyboard shortcuts do.

|  |  |
| --- | --- |
| **Keyboard shortcut** | **What does the keyboard shortcut do?** |
| Control c |  |
| Control v |  |
| Control a |  |
| Control x |  |
| Control f |  |
| Control n |  |
| Control p |  |
| Control s |  |
| Control z |  |

**Activity 2.3**

Copy and run the following program:

print("This is the end")

print("Hold your breath and count to ten”)

print("Feel the earth move and then”)

print("Hear my heart burst again”)

Write a program that writes four lines of the lyrics of your favourite song on the screen.

**Activity 2.4**

Copy and run the following program.

print("the sky fall","from the sky")

Now answer these questions: what happens? what effect does the “,” have?

**Activity 2.5**

Escape sequences can be used to alter how the information is displayed on the screen. An escape sequence is a back slash “\”

Experiment with the following escape sequences and complete the table.

*Hint: Use alt p to display the last entered command in the IDLE shell.*

print ("\tQuestion what goes woof\t\tdogs\t\t\trabbits")

print("\n\nwhat kind of snake is good at maths?\n\nAn adder\n\n")

print("\n\nGoodbye\n\n")

|  |  |
| --- | --- |
| **Escape sequence** | **Effect** |
| \t |  |
| \n |  |
| \\ |  |
| \’ |  |
| \” |  |

**Activity 2.6**

Print the following text using just one line of code.

Help, I need somebody  
Help, not just anybody  
Help, you know, I need someone

**Activity 2.7**

Write a program to solve the problem.

The problem: Display your initials 5 characters high on the screen using print commands.

X x x x

Xx x x X

X x x Xxxxxx

X xx x x

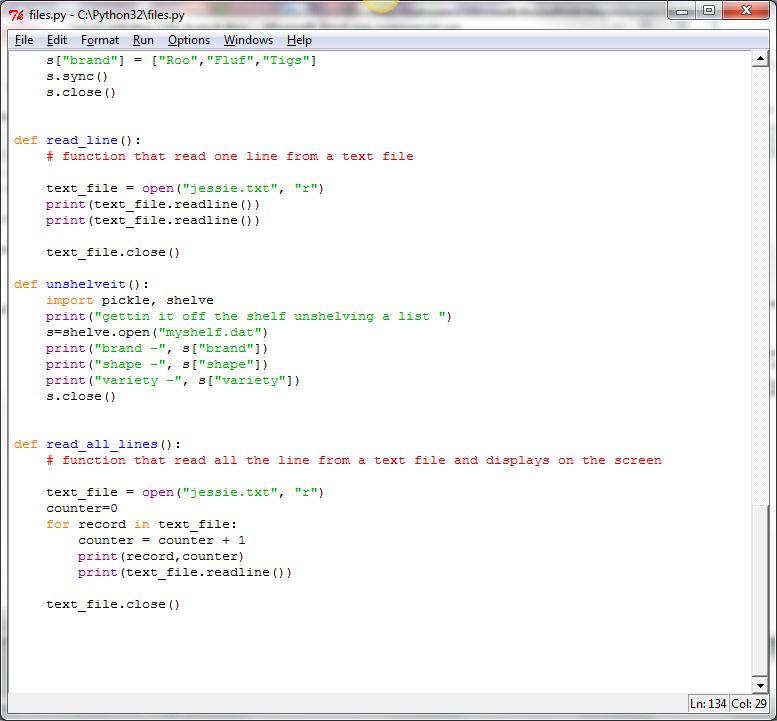
X x x x

**Activity 2.8**

An algorithm is a sequence of steps to perform a particular task. An algorithm can be represented in different ways.

Match the words with the representation.

**flowchart pesudocode structured English written description program code**



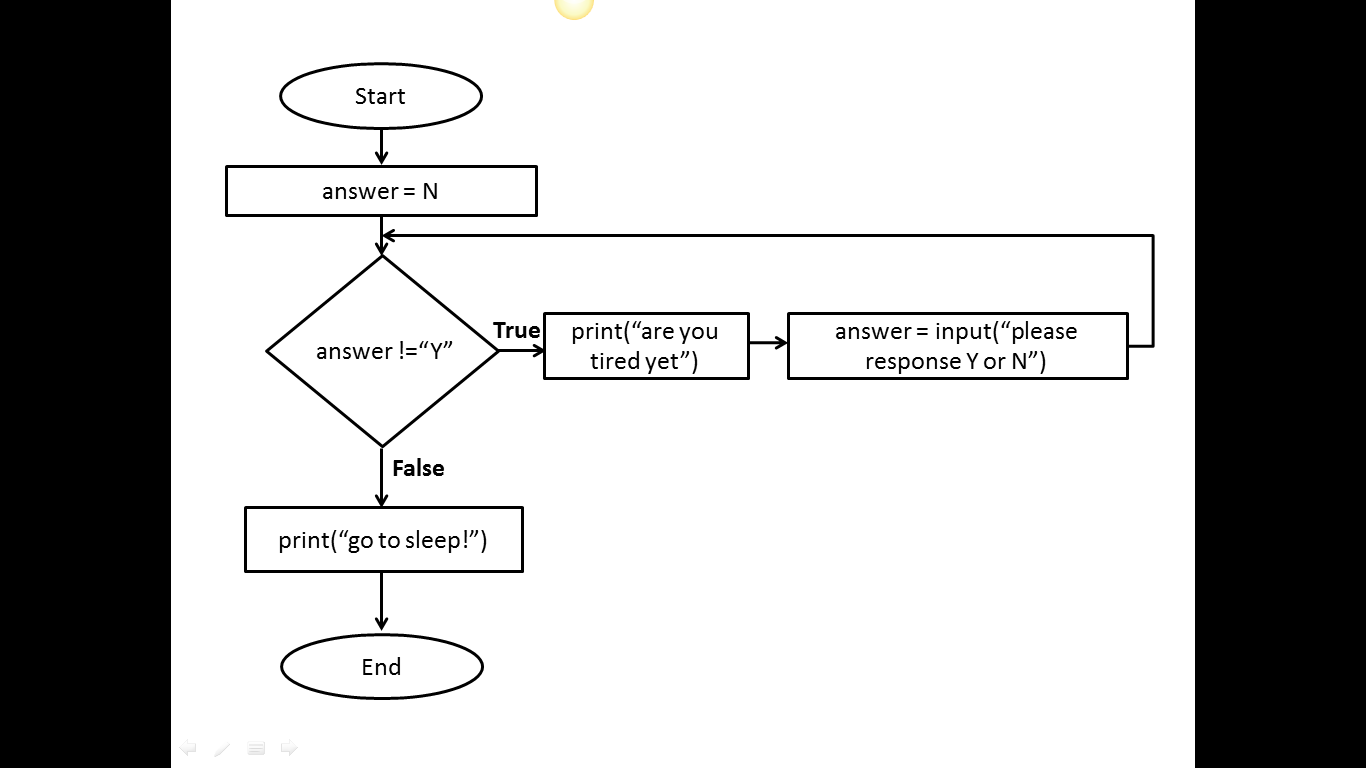
RECEIVE myName FROM (STRING) KEYBOARD

RECEIVE myAge FROM (INTEGER) KEYBOARD

SET AgeInTen = myAge + 10

SEND myName “will be” AgeInTen “in 10 years time”

Write a program that prompts to enter a name and age. The program then adds 10 onto the age and displays the text “<name entered> will be <age + 10> in 10 years time.



If animal has 4 legs then

If animal has a tail

If animal answers to “puss”

animal = cat

endif

endif

endif

**Activity 2.9**

Explore ASCII art (Wikipedia) and write a Python program to display your ASCII art creation.

.--. /\ \_\_\_\_

'--' /\_\_\ (^.\_.^)~ <(o.o )>

**Lesson 3 activities**

**Activity 3.1**

Copy and run the following lines of program. Complete the table to explain what the mathematical operators do.

>>> 60/5

>>> 987+34

>>> 564\*89

>>> 2\*\*5

>>> 43-5

>>> 11//2

>>> 11%2

|  |  |
| --- | --- |
| **Mathematical operator symbol** | **Operation** |
| / |  |
| + |  |
| \* |  |
| \*\* |  |
| - |  |
| // |  |
| % |  |

**Activity 3.2**

Make up some mathematical calculations of your own and add an example to the table for each mathematical operator.

|  |  |  |  |
| --- | --- | --- | --- |
| **Mathematical operator symbol** | **Operation** | **Example** | **Answer** |
| / | divide |  |  |
| + | add |  |  |
| \* | multiply |  |  |
| \*\* | exponential |  |  |
| - | subtract |  |  |
| // | integer division |  |  |
| % | modulus (remainder after the division) |  |  |

**Activity 3.3**

Write the program to display the text on the screen and calculate the missing gaps.

8 cats have 4 legs each

The cats have \_\_\_ legs in total

A farmer with 1089 sheep sells 56 of them

The farmer has \_\_\_\_\_ sheep left

4 children pick 56 flowers each

The children each have \_\_\_\_ flowers

**Activity 3.4**

Copy and run the following lines of program. What effect do the parentheses () have?

>>> 5 \* 3 / 6 + 4

>>> (5 \* 3) / (6 + 4)

**Activity 3.5**

Predict what you think will be the answer to the command below then run the command. Explain the answer.

15 / 2 \* 3 + 2

**Activity 3.6**

Make up some multiple-choice questions of your own on precedence, each with 4 answers. Check your answers are correct by using the interactive Python shell. Try your questions out on other students. You must be able to explain why the answer is correct.

Two examples are given below.

What is the correct answer to the following expression?

>>> 7 + 4 \* 5

1. 55
2. 27
3. 16
4. 33

What is the correct answer to the following expression?

>>> 6 -2 / 2 + 5

1. 0.57
2. 10
3. 7
4. 11

**Activity 3.7**

Devise your own mnemonic for remembering the order of precedence.

|  |  |
| --- | --- |
| **Python order of precedence** | **Mnemonic** |
| Parenthesis (brackets) |  |
| Exponential |  |
| Division and multiplication |  |
| Addition and subtraction |  |

**Lesson 4 activities**

**Activity 4.1**

Copy and run the following code and explain the result.

>>> print hello world

**Activity 4.2**

Copy and run the following code and explain the result.

>>>Print("hello world")

**Activity 4.3**

Python produces different types of error. Match up the type of error with its description.

|  |
| --- |
| **Type of error** |
| TypeError |
| RuntimeError |
| NameError |
| ZeroDivisionError |
| KeyBoardInterrupt |

|  |
| --- |
| **Description of the error** |
| Dividing a number by zero |
| When a program is interrupted from the keyboard by pressing control+c |
| When a name is used that is not known about (often a variable name that is misspelt) |
| When an operation is attempt that is invalid for that type of data. |
| An error occurs when the program is running. |

Experiment with Python to see if you can create each type of error.

A full list can be found here <http://www.tutorialspoint.com/python/standard_exceptions.htm>

**Activity 4.4**

Copy and run the following code and explain the result.

# Programmer Amy Smith 6th September 2013

**Activity 4.5**

Use *file/new window* to open a script file in the IDLE.

Copy the following code into the file.

Save the file using *file/save*. Always give the filename an extension of .py. Use a meaningful filename and store in a suitable folder.

Run the commands by using *run/run module*

print("So long and thanks for all the fish")

print("The answer is",6\*7)

Close Python and then restart Python, open the file and run the program again.

**Activity 4.6**

Following the same steps as in Activity 4.5, write a program that displays your name and then your age. (Use *file/new window* and save the file using *file/save*.) Don’t forget to use a meaningful filename.

**Activity 4.7**

**Using the Python IDLE**

Write your own summary sheet to describe how to run python programs including how to save and open .py files. Include any other commands that you have found useful.

|  |  |
| --- | --- |
| **Using the Python IDLE** |  |
| How to open a new window |  |
| How to save a file |  |
| How to open a file |  |
| How to run a program |  |
|  |  |
|  |  |
|  |  |
| **Useful tips** |  |
| alt p | Displays the last line you entered |
| Commands in the file are not colour coded | Save as a .py file |
|  |  |
|  |  |

**Lesson 5 activities**

**Activity 5.1**

**Where does the name “Boolean” come from?**

Boolean variables are named after George Boole who invented the mathematics of digital logic. Find a picture of George Boole and insert it here. When was he born?

**Activity 5.2**

**Finding the data type:** Use the “type” function to find out the data types for these values.

>>>type("Fred")

>>> type(198)

>>> type(88.9)

>>>type(True)

>>>type(False)

*Remember that True and False must start with a capital letter.*

**Activity 5.3**

The type function returns the data type of an expression.

For each of the following expressions predict the data type then use the type command to check your prediction.

|  |  |  |  |
| --- | --- | --- | --- |
| **expression** | **Predicted data type** | **Type command** | **Result** |
| “hello world” |  | type("hello world") |  |
| False |  | type(False) |  |
| 15 |  | type(15) |  |
| 35.6 |  | type(35.6) |  |
| -999 |  | type(-999) |  |
| “15” |  | type(“15”) |  |
| “False” |  | type(“False”) |  |
| True |  | type(True) |  |
| 0.001 |  | type(0.001) |  |

**Activity 5.4**

The interactive shell is a useful place to explore variables.

Copy and run the following code in the interactive shell.

>>>myName="Fred Smith"

>>>myAge=14

>>>print(myName,myAge)

>>>myName

>>>myName="Amy Jones"

>>>myName

Create a variable called myName and assign it to your name.

Create a variable called myAge and assign it to your age

Create a variable called myEyes and assign it to your eye colour.

Create a variable called myHeight and assign it to your height.

Write the commands to display on the screen your name, age, eye colour and height.

*Hint – Remember that a single “=” is used to assign a value to a variable.*

**Activity 5.5**

**Variable names**

A programmer is trying to decide which would be a valid name for a variable which represents a house number.

Which of the following variable assignments are valid? Why are the others not valid?

|  |  |  |
| --- | --- | --- |
|  | **Valid or invalid variable name?** | **Reason why not valid** |
| 8HouseNumber = 288 |  |  |
| houseNumber = 288 |  |  |
| house Number = 288 |  |  |
| house\_number = 288 |  |  |
| import = 288 |  |  |

What type of error do you get when using an invalid variable name?

**Activity 5.6**

Copy and run the following code and explain the result.

# Programmer Amy Jones 12/8/2013

# adds two numbers

numberOne=15

numberTwo=23

answer=numberOne + numberTwo

print("The answer is “,answer)

Amend the program to add another variable called numberThree. Assign the value 76 to this variable. The answer should add up all three numbers.

**Activity 5.7**

**Data types in Python**

Complete the table to describe the four data types.

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type** | **Python abbreviation** | **Explanation** | **Example** |
| integer | int |  |  |
| string | str |  |  |
| float | float |  |  |
| boolean | bool |  |  |

*Hint – quotation marks are used to show the start and end point of a string “this is a string”.*

**Lesson 6 activities**

**Activity 6.1**

**Input function**

The input function allows the user to input a value and assign it to a variable.

Copy and run the following program.

myAnswer=input("Please enter your name: ")

print(myAnswer)

Re-run the program several times, entering different names each time.

Write a program that asks for your name and your favourite food and displays on the screen the name and their favourite food.

**Activity 6.2**

**Int function**

Copy and run the following program.

age=input("Please enter your age: ")

agePlusTen = age + 10

print("You will be",agePlusTen,"in 10 years")

Explain why it does not work.

Correct the program.

**Activity 6.3**

Write a program that asks you to enter a number then displays the number doubled.

**Activity 6.4**

**String formatting**

The string method .format gives you more control over the formatting of your output than printing using space-separated values. The string formatting commands are given in curly brackets {}.

Copy and run the following commands.

>>>foodOne="fish"

>>>foodTwo="chips"

>>>print("{0} and {1}".format(foodOne,foodTwo))

fish and chips

>>>print("{1} and {0}".format(foodOne,foodTwo))

chips and fish

>>>print("{1} {1} {1} and {0} {0} {0}".format(foodOne,foodTwo))

chipschipschips and fishfishfish

Create these variables:

one = “cheese”

two =”onion”

Use the .format command to display the following:

My favourite crisps are cheese and onion. I love them!

cheese and onion and cheese and onion and cheese and onion

cheesecheesecheese and oniononiononion

You guessed it. The best crisps are onion and … cheese.

Try altering the flavours assigned to the variables to your favourite flavour! Enjoy.

**Activity 6.5**

**Formatting numbers**

Using the .format method to format the number of decimal places.

Copy and run the following commands.

>>> number = 98.1468297645

>>>print("The answer is {0:.5f}".format(number))

The answer is 98.14683

>>>print("The answer is {0:.4f}".format(number))

The answer is 98.1468

>>>print("The answer is {0:.3f}".format(number))

The answer is 98.147

>>>print("The answer is {0:.1f}".format(number))

The answer is 98.1

>>>print("The answer is {0:.0f}".format(number))

The answer is 98

Assign the number 765.87641987 to a variable and display the number with 5, 2 and no decimal places using .format command.

**Activity 6.6**

Write a program that asks how much your bill is at a restaurant and then asks you to enter the % you want to give in a tip. The program should display the amount of tip to give to the waiter.

MealCost=float(input("Enter total cost of the meal: £"))

TipPercentage=float(input("Enter what percentage to give as a tip (%):"))

MealPlusTip=(MealCost/100 \* TipPercentage) + MealCost

print("Total cost of the meal is £ {0:.2f}".format(MealPlusTip))

**Activity 6.7**

*The following programs should prompt for the input information and display the result appropriately.*

* Write a program that displays the square of a number.
* Write a program that prompts for a number and then displays the cube of a number.
* Write a program to find the perimeter of a square.
* Write a program to find the perimeter of a rectangle.
* Write a program that finds the area of a square.
* Write a program that finds the area of a cube.
* Write a program to convert from pounds to euros.

**Lesson 7 activities**

**Activity 7.1**

**Relational operators**

Password checking is an example of a relational operator. If the password entered is the same as password stored then the condition is true. The operator is “is equal to”.

Brainstorm other examples of condition statements and decide what the operator may be.

|  |  |
| --- | --- |
| **Relation statement** | **Operator** |
|  |  |
|  |  |
|  |  |

**Activity 7.2**

Complete the table of the Python relational operators.

Give an example of each and say whether it will evaluate to true or false. Try out your expression by typing it into the Python interactive shell.

|  |  |  |  |
| --- | --- | --- | --- |
| **Relational operator** | **Operator** | **Example** | **Evaluates to** |
| Equal to |  |  |  |
| Not equal to |  |  |  |
| Greater than |  |  |  |
| Greater than or equal to |  |  |  |
| Less than |  |  |  |
| Less than or equal to |  |  |  |

**Activity 7.3**

**Greater than and less than**

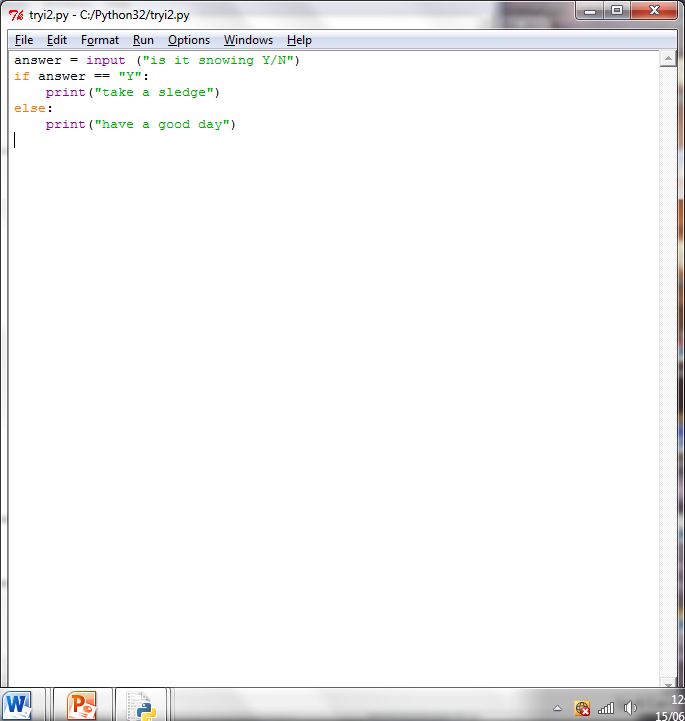
Find a way that works for you of remembering the difference between “less than” < and “greater than” >.

*Hint: As there are only two options you only need to learn one!*

|  |  |  |
| --- | --- | --- |
| **Operator** | > | < |
| **Operator meaning** | Greater than | Less than |
| **How I remember this** |  |  |

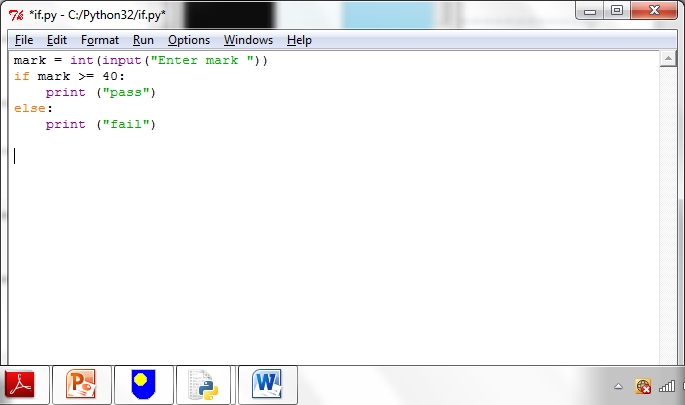
**Activity 7.4**

This program asks if it is snowing and if so tell you to take a sledge otherwise to have a good day. The condition is missing.   
  
Copy and complete the condition to make the program work correctly.



**Activity 7.5**

What condition is needed in this program to display “pass” if the exam mark is greater than 40?  
Copy and complete the condition to make the program work correctly.



**Activity 7.6**

Write a program that asks you to enter the colour of the light at a pedestrian crossing. If the light is green it tells you it is safe to cross otherwise it tells you to stop.

**Activity 7.7**

Write a program that asks you your password. It then asks you to re-enter your password. If they are the same the message “access granted” is displayed. If the passwords are not the same the message “access denied” is displayed.

**Lesson 8 activities**

**Activity 8.1**

**Using elif**

This program simulates a fortune cookie. A random number is used to decide your “fortune”.

Copy and run this program.

# a random number is given by the randint() function

import random

answer= random.randint(1,6)

if answer == 1:

print("You will meet a stranger")

elif answer == 2:

print("Your favourite colour is blue")

elif answer == 3:

print("You enjoy programming")

The program is not complete. Finish the program by including your own “fortunes” for the random numbers 4, 5 and 6.

random.randint(1,6) is a function that returns a random number between 1 and 6. The “import random” command allows the program to access the random.randint() function.

**Activity 8.2**

**Writing readable code: a style guide**

Program code is read more often that it is written. Here are some guidelines on how to write Python code to improve the readability of the code.

**A style guide for Python code**

* Use the 4 spaces indentation for each indentation level
* Separate functions by 2 blank lines
* Use blank lines to separate different parts of the program
* Use meaningful names for variables using CamelCase or with words separated by underscores
* Put imports at the top of the file
* Include one space around each side of an assignment and other operators
* Comments should be complete sentences with the first word capitalised
* Comments should add clarity to explain what the program does and not just repeat what the code already says
* Function names should be written in lowercase with words separated by underscores
* Use meaningful function names which describe the purpose of the function
* Constants are written in CAPITAL\_LETTERS
* Use meaningful constant names which describe the purpose of the constant

Note: Functions are sub-programs which start with the “deffunction\_name():. Constants are variables which never change. You will be covering functions in a later lessons.

For more detail see <http://www.python.org/dev/peps/pep-0008/>

Implement the style guide for this Python code.

def a(s):

if s<50:

print("You have lost")

else:

print("You have won")

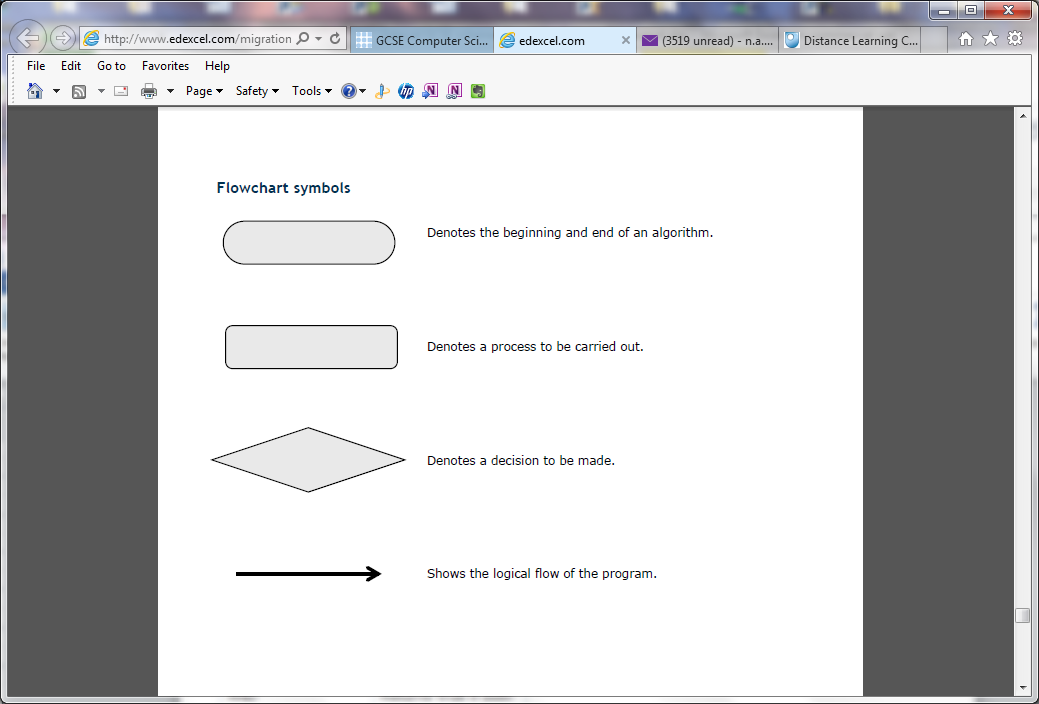
**Activity 8.3**

Select some of your programs and make them more readable using the style guide.**Lesson 9 activities**

**Activity 9.1**

**Flowcharts**

Flowcharts can be used to represent algorithms. Identify the flowchart symbols.



**Activity 9.2**

Study the program.

# password checking

password=input("Please enter password: ")

newPassword=input("Please re-enter password: ")

if password == newPassword:

print("Access granted")

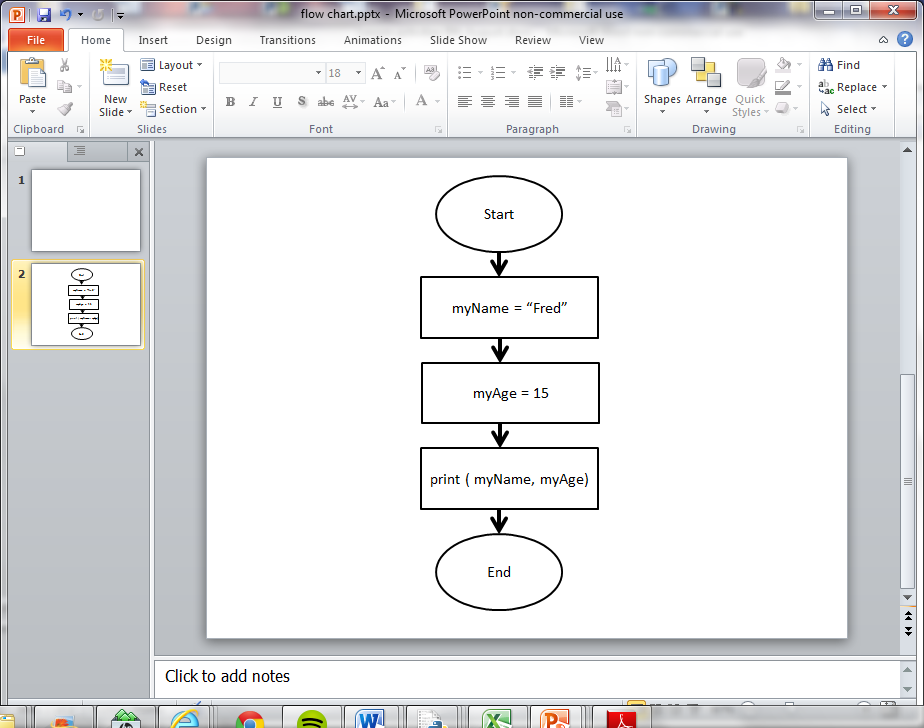
else:

print("Access denied")

Draw the flowchart for this program.

**Activity 9.3**

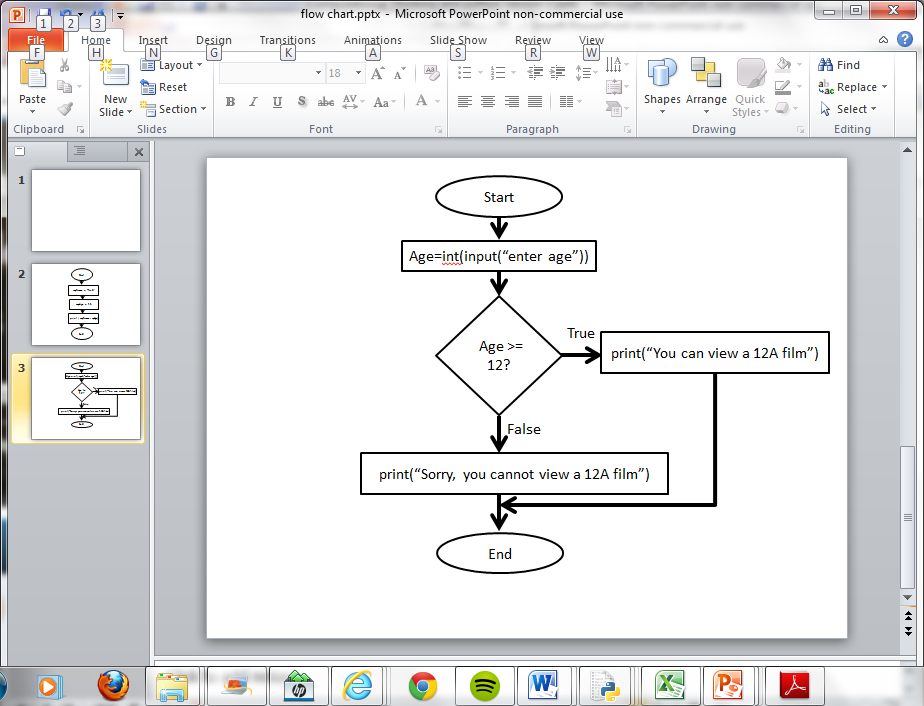
Study the flowchart.



Write a program to implement this flowchart.

**Activity 9.4**

Study the flowchart.



What does this flowchart do?

Write the program to implement this flowchart.

**Activity 9.5**

**Boolean Operators**

Complete the table using the logical (Boolean) operators AND, OR and NOT.

|  |  |
| --- | --- |
| **Symbol** | **Description** |
|  | Returns true if both conditions are true. |
|  | Returns true if any of the conditions are true. |
|  | Reverses the outcome of the expression; true becomes false, false becomes true. |

**Activity 9.6**

Predict the answer to the conditions and then use the interactive shell to test your answer.

|  |  |  |
| --- | --- | --- |
| **Condition** | **Your prediction (true or false?)** | **Result** |
| (78 = 10) **or** (6 = 7) |  |  |
| (78 == 10) **or** (6 == 6) |  |  |
| (78 == 10) **and** (6 == 6) |  |  |
| (1 < 10) **and** (2 < 10) |  |  |
| (1 < 10) **or** (2 < 10) |  |  |
| not ( 5 ==5) |  |  |
| not (6 < 4) |  |  |

*Hint: Try asking yourself the question:*

*Is condition\_1 true* ***OR*** *condition\_2 true – if YES then the answer is true.*

*Is condition\_1 true AND condition\_2 true – if YES then the answer is true.*

**Activity 9.7**

Make up some boolean operator questions of your own. Check your answers are correct by using the interactive Python shell. Try them out on other students. You must be able to explain the correct answer.

An example question is given below.

>>> answer = 50

>>> (answer <40) or (answer >80)

True or False?

**Activity 9.8**

A truth table lists all the possible combinations of true and false outcomes for each condition.

Complete the truth table for AND and OR operators.

**Truth table showing true and false AND conditions**

|  |  |  |
| --- | --- | --- |
| **Condition 1** | **Condition 2** | **Output** |
| false | false |  |
| true | false |  |
| false | true |  |
| true | true |  |

**Truth table showing true and false OR conditions**

|  |  |  |
| --- | --- | --- |
| **Condition 1** | **Condition 2** | **Output** |
| false | false |  |
| true | false |  |
| false | true |  |
| true | true |  |

**Activity 9.9**

Write a program that asks you to enter a year and tells you which key stage it is.

|  |  |
| --- | --- |
| **Key Stage** | **Year Group** |
| Key stage 1 | Years 1 and 2 |
| Key stage 2 | Years 3, 4, 5 and 6 |
| Key stage 3 | Years 7, 8 and 9 |
| Key stage 4 | Years 10 and 11 |

**Activity 9.10**

**Python commands colour coding**

The Python language is colour coded.

Complete the table below giving an example for each type.

|  |  |  |
| --- | --- | --- |
| **Colour coding** | **What does it show** | **Example** |
| green |  |  |
| purple |  |  |
| Black |  |  |
| orange |  |  |
| red |  |  |

**Lesson 10 activities**

**Activity 10.1**

**Strings**

Write the index position for each character in the string.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| H | e | l | l | o |  | w | o | r | l | d | ! |
|  |  |  |  |  |  |  |  |  |  |  |  |

What character is at index 3?

What character is at index 5?

What character is at index 0?

What character is at index 11?

How many characters are there in this string?

**Activity 10.2**

Use the interactive shell to explore how strings can be concatenated.

* Make a **variable** called FirstName and set its value to the string “Robert”
* Make a **variable** called SecondName and set its value to the string “Smith”
* Make a **variable** called FullName and set its value to the value of the **variables**FirstName plus SecondName
* Print the value of the **variable**FullName

**Activity 10.3**

**Slicing strings**

Parts of a string can be extracted (or sliced) by giving the index location in the format.

[start position: end position]

Start position is the index at which the slice starts (remember that indexing starts at zero)

End position is the index AFTER the last index required.

Use the interactive shell to explore how strings can be sliced.

Make a variable called word and set its value to the string “PIZZA”.

>>> word="PIZZA"

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P | I | Z | Z | A |
| 0 | 1 | 2 | 3 | 4 |

What do these commands do?

>>>word[2:5]

>>>word[1:2]

>>>word[0:3]

>>>word[0:5]

*Tip: Remember that the index starts at zero not one.*

**Activity 10.4**

A variable has been created and assigned the string “watch bbcclick today”

myVariable = “watch #bbcclick today”

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| w | a | t | c | h |  | # | b | b | c | c | l | i | c | k |  | t | o | d | a | y |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

What command would you use to extract the following slices?

#bbcclick

watch

today

Check your answer using the interactive shell.

**Activity 10.5**

**Using string methods**

Any variable with a string assigned is a member of the class called string. String methods can be used to manipulate that string.

Explore string methods by working through the commands in the interactive shell and describing what happens.

>>>myVariable="There's a starman waiting in the sky"

>>>myVariable.upper()

>>>myVariable.replace("a","x")

>>>myVariable.title()

>>>myVariable.swapcase()

**Activity 10.6**

**Review using the string .format method**

Using the string .format method to format output.

There are two ways to output information to the screen. Using the print() function or by giving the expression.

Try these now.

>>>myName="David Bowie"

>>>print(myName)

David Bowie

>>>myName

'David Bowie'

The string .format is used to give more control over formatting.

It allows you to use placeholders and then specify the variable you want to print in the string. The place holders are shown using {}.

>>> first="David"

>>> second="Bowie"

>>>print("The best music is from {0} {1}!".format(first,second))

The best music is from David Bowie!

It also allows you to choose how to format the output.

**Lesson 11 activities**

**Activity 11.1**

Complete the table below to summarise the list commands.

|  |  |
| --- | --- |
| **Things to do with lists** | **Commands** |
| Create a list |  |
| Reference an item in a list |  |
| Delete an item in a list |  |
| Append an item to the end of a list |  |

**Activity 11.2**

**Lists (arrays)**

Lists are a data structure in which you can store things. Lists are assigned to a name using square brackets:

>>>mylist=["apple","oranges","lemon","pear","lime"]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| apples | oranges | lemon | pear | lime |
| 0 | 1 | 2 | 3 | 4 |

Each item in a list is given an index location which starts from position zero (0).

Parts of a list can be referenced by giving the index location as an integer number.

List name[index]

A range of values can be displayed using:

[start index: end index]

* Start index is the index to start at (remember that indexing starts at zero)
* End position is the index AFTER the index required.

Make the list and experiment with the list commands.

>>>mylist=["apple","oranges","lemon","pear","lime"]

What does **mylist[1]** display?

What does **mylist[1:3]** display?

What does **mylist[-1]** display?

What command will display just apple?

What command will display lemon and pear?

Make a new list called myfood containing your 5 favourite foods.

Display the whole list.

Display the item at index position 3

Display the item at index position 0

Display the items at index position 1 to 4.

**Activity 11.3**

**Using lists**

Make the list which contains the class marks for Amy Jones.

Marks = ['Amy', 'Jones', 'English', 67, 'Maths', 76, 'Computer Science', 96]

Answer the following questions:

The English teacher entered the mark incorrectly; it should be 72 not 67. Alter this item in the list.

Add the marks for Physics to the end of the list. “Physics”, 65

The Maths marks are all wrong. Remove the items for “Maths” and the score 76

Write a program to find the average score for the 3 subjects.

**Activity 11.4**

**Using Python docs help**

Use Python docs to find out more about lists.

Use *help/python docs* then select the *Python tutorial* and go to *3.1.4 Lists.*

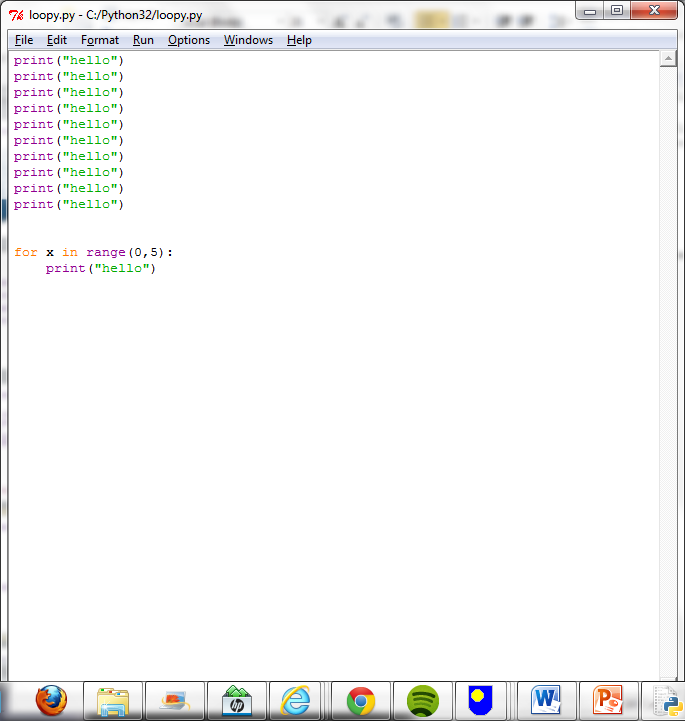
Read through the discussion of lists and try out the examples. Make a note of three more facts about lists to share in the next class.

Python is avery powerful programming language which is used in Universities and commercial organisations. You do not need to know all the details provided in the Python docs but, with practice, you should be able to find information about Python that can be very useful.**Lesson 12 activities**

**Activity 12.1**

**For loops**

Copy and run this program



Write a program that prints out “I like Kate Bush” 10 times.

**Activity 12.2**

Copy and run the following program.

for number in range(10):

print(number)

Explain what does this program does.

Write a program that print the numbers from 1 to 15.

Write a program that prints the numbers from 1 to 8 with the square of each number, both on the same line.

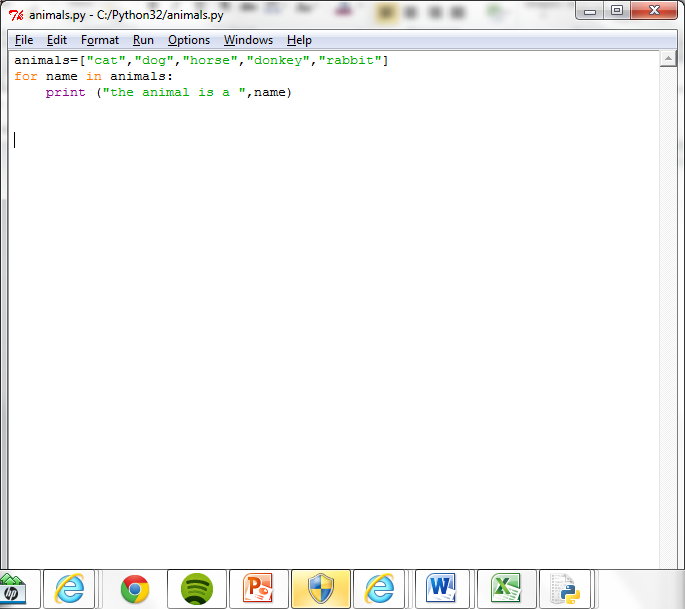
Write a program to print out the 9 times table (up to 20 x 9)

Write a program that ask you which times table to print out and then displays that table.

Write a program that asks for a start value and end value and then counts in fives between the numbers. *(Hint: range(start value, end value, step))*

**Activity 12.3**

Copy this program into a file and run the program.



Explain the purpose of the “name” variable.

Write a program that creates a list of things you would take to a desert island. The program should then display each item in the list a line at a time.

**Activity 12.4**

Re-arrange the program statements to write a program that will print out the names of the animals that begin with the character “c”.

print(next)

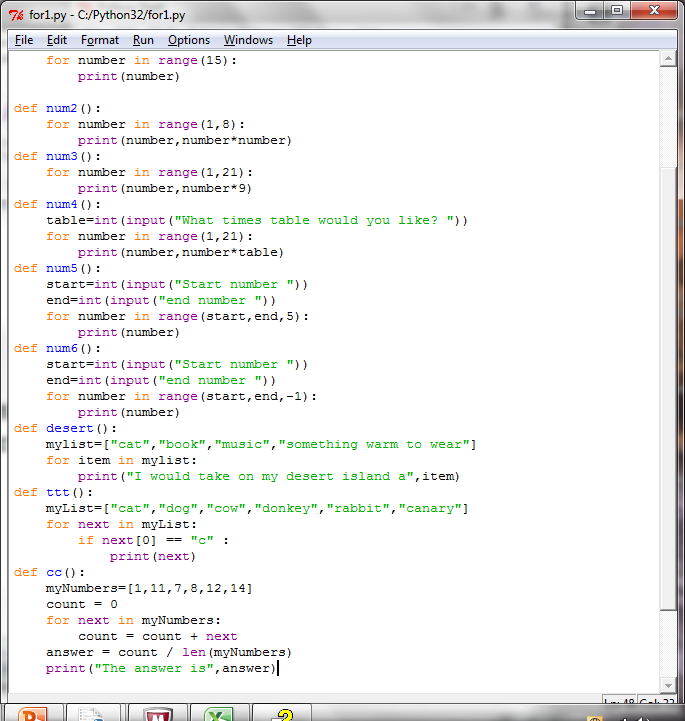
if next[0] == "c" :

myList=["cat","dog","cow","donkey","rabbit","canary"]

for next in myList:

**Activity 12.5**

Study this program code.



What will this program do?

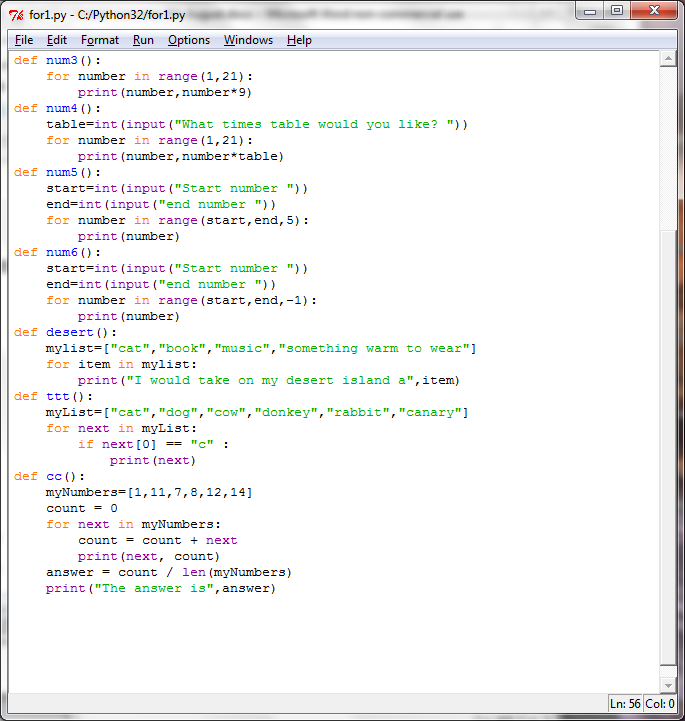
Explain what the variable next is used for?

Explain what the variable countis used for?

What does len (myNumbers) do?

What happens if you add or append more numbers in the list

If a print statement was added to the for loop as shown below what would be displayed?



**Activity 12.6**

Write a program that finds the largest number in this list.

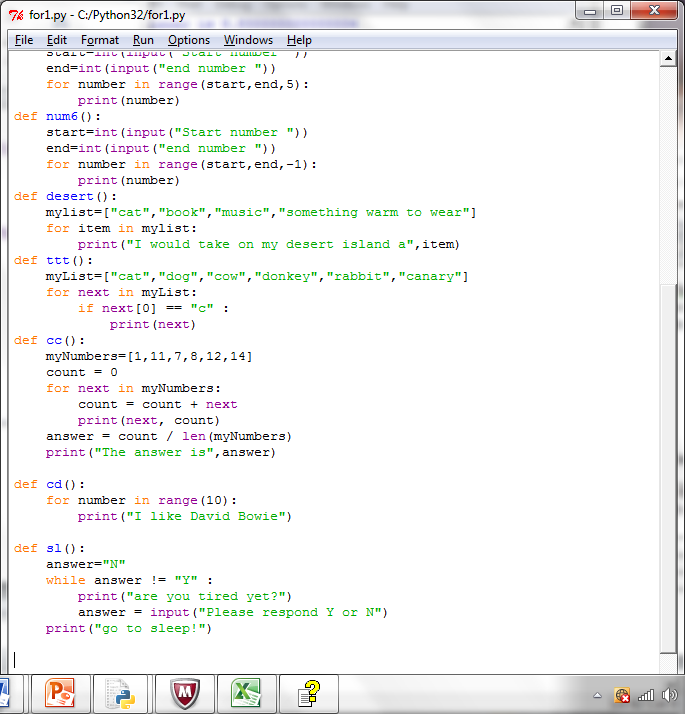
myNumbers=[19,6,7,9,2,25,16]**Lesson 13 activities**

**Activity 13.1**

**While command**

What does this while command do?

What is the condition that is being tested and how does it change?



**Activity 13.2**

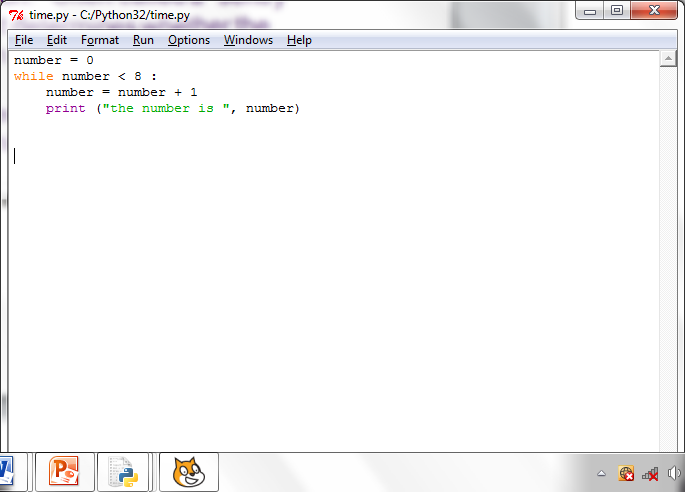
Write a program that asks you if you are hungry. While you reply “N” the program repeats the question showing a counter for how many times you have replied “N”. When you reply “Y” the program tells you to get something to eat!

**Activity 13.3**

What does this program do?

What is the “sentry” variable?

What is the condition?

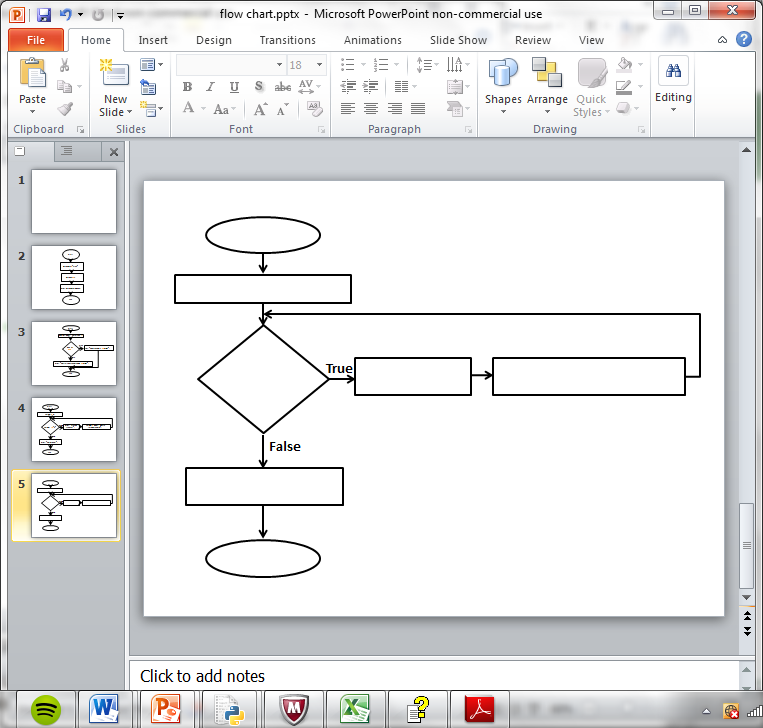
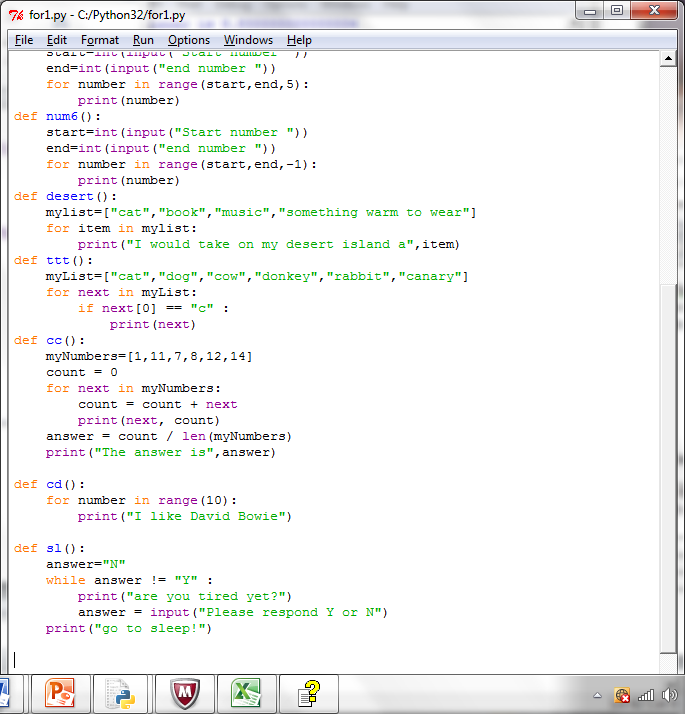


**Activity 13.4**

**Flowcharts and while command**

Flowcharts can be used to represent programs.

Complete the flow chart for this program.



**Lesson 14 activities**

**Activity 14.1**

**Pseudocode**

Pseudocode, or “mock” code, is another way of describing a program. You will need a copy of the pseudocode used in this course (Appendix B of the specification) to answer the following questions.

What does this pseudocode do?

RECEIVE myName FROM (STRING) KEYBOARD

RECEIVE myAge FROM (INTEGER) KEYBOARD

SET AgeInTen = myAge + 10

SEND myName “will be” AgeInTen “in 10 years time”

Write the Python code for this program.

**Activity 14.2**

What does this pseudocode do?

SET score TO 119

IF score < 50 THEN SEND “You have lost” TO DISPLAY

ELSE SEND “You have won” TO DISPLAY

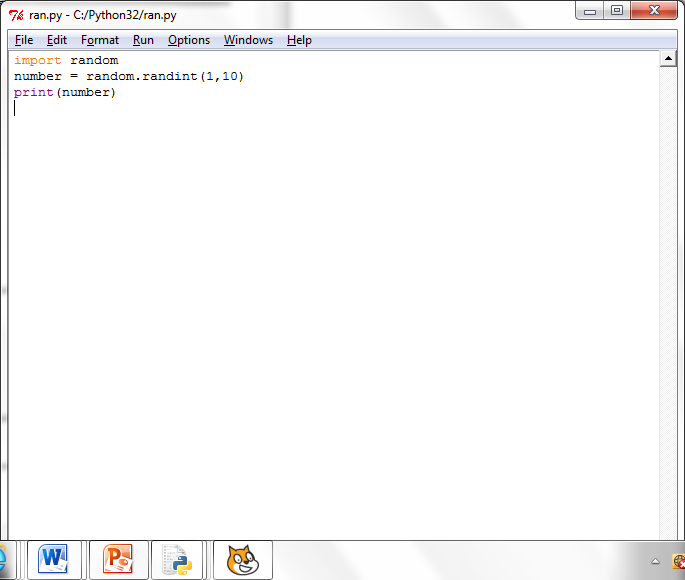
END IF

Write the Python code for this program.

**Activity 14.3**

**random() function**

Copy and run the program code.



Run the program a few times. What does it do?

What is meant by “import random”?

What happens if you alter the values to 1, 100?

**Activity 14.4**

Write a program that acts like a dice. After each “throw” of the dice it should ask if you wish to continue and stop when you enter “Y”.

**Activity 14.5**

Write a program that asks you to guess a number between 1 and 10. It then compares your answer with a randomly generated number and stops the program when the number generated matches the number guessed.

**Lesson 15 activities**

**Activity 15.1**

**Lists and for loops: a review**

Study the command and answer the questions

>>>countries= ["Japan","Germany","USA","China","Austria","Turkey","Mexico"]

* What does this command do?
* How would you display all the values?
* How would you display “China”?
* How would you display “Japan”
* How would we address 3th item in the list?
* Copy and run this program. Explain how it works.

for name in countries:

print(“This is one of my favourite countries “, name)

* Copy and run this program which uses list comprehension. Explain how it works.

length=10

myList = [43 for number in range(length)]

print(myList)

**Activity 15.2**

**Battleships: a game using two-dimensional array addressing (lists)**

**How to set up the game**

Each player decides at which index locations in the two dimensional array [row, column] to place their ships.

They have 5 ships:

* A battle ship that takes up 5 index spaces
* A cruiser that takes up 4 index spaces
* A submarine that takes up 3 index spaces
* A destroyer that takes up 2 index space
* Four spy ships disguised as a fishing boats that each take up 1 index space.

None of the ships may be placed diagonally; they must all be placed in straight lines either horizontally or vertically. It is legal (but not required) for two or more ships to be adjacent to each other. The ships are marked by blocking in the appropriate spaces.

**How to play the game**

Players take turns taking shots at each other’s ships. A shot is taken by calling out the index locations on the 8 x 8 two-dimensional array. The array index locations are given [row, column] e.g. [2, 6].

Each player takes one shot at a time.

If the player calls the coordinates of an index where a ship is located, his opponent tells him so by saying "hit." If he missed, his opponent says "miss."

Players mark the shots they take on their "Opponent" array, and whether each shot was a hit or a miss to keep track of their shots. Players may also mark the "Self" array to show the shots taken by their opponent.

A ship is sunk when all of its index locations have been hit. When this happens, the player whose ship was sunk says, for example, "You sank my spy ship."

The winner is the play who manages to sink all their opponent’s ships.

**Battleships: My battle ships**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| [0,0] | [0,1] | [0,2] | [0,3] | [0,4] | [0,5] | [0,6] | [0,7] |
| [1,0] | [1,1] | [1,2] | [1,3] | [1,4] | [1,5] | [1,6] | [1,7] |
| [2,0] | [2,1] | [2,2] | [2,3] | [2,4] | [2,5] | [2,6] | [2,7] |
| [3,0] | [3,1] | [3,2] | [3,3] | [3,4] | [3,5] | [3,6] | [3,7] |
| [4,0] | [4,1] | [4,2] | [4,3] | [4,4] | [4,5] | [4,6] | [4,7] |
| [5,0] | [5,1] | [5,2] | [5,3] | [5,4] | [4,5] | [5,6] | [5,7] |
| [6,0] | [6,1] | [6,2] | [6,3] | [6,4] | [6,5] | [6,6] | [6,7] |
| [7,0] | [7,1] | [7,2] | [7,3] | [7,4] | [7,5] | [7,6] | [7,7] |
| [8,0] | [8,1] | [8,2] | [8,3] | [8,4] | [8,5] | [8,6] | [8,7] |

**Battleships: Opponent’s battle ships**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| [0,0] | [0,1] | [0,2] | [0,3] | [0,4] | [0,5] | [0,6] | [0,7] |
| [1,0] | [1,1] | [1,2] | [1,3] | [1,4] | [1,5] | [1,6] | [1,7] |
| [2,0] | [2,1] | [2,2] | [2,3] | [2,4] | [2,5] | [2,6] | [2,7] |
| [3,0] | [3,1] | [3,2] | [3,3] | [3,4] | [3,5] | [3,6] | [3,7] |
| [4,0] | [4,1] | [4,2] | [4,3] | [4,4] | [4,5] | [4,6] | [4,7] |
| [5,0] | [5,1] | [5,2] | [5,3] | [5,4] | [4,5] | [5,6] | [5,7] |
| [6,0] | [6,1] | [6,2] | [6,3] | [6,4] | [6,5] | [6,6] | [6,7] |
| [7,0] | [7,1] | [7,2] | [7,3] | [7,4] | [7,5] | [7,6] | [7,7] |
| [8,0] | [8,1] | [8,2] | [8,3] | [8,4] | [8,5] | [8,6] | [8,7] |

**Activity 15.3**

**Two-dimensional arrays (using nested lists)**

In Python, two-dimensional arrays are represented as nested lists (a list of lists) so the addresses are given as [row] [column] rather than [row,column]

Copy and run the program which creates a two-dimensional array (a list of lists in Python) using nested for loops and initialises each element with zero (0). Run the program. What happens and why?

rowLength=4

columnLength=6

myArray=[[0 for row in range(rowLength)] for column in range(columnLength)]

print(myArray)

Change the value of “0” to “86” and run the program again. Run the program. What happens and why?

rowLength=4

columnLength=6

myArray=[[86 for row in range(rowLength)] for column in range(columnLength)]

print(myArray)

Change the initialised value back to 0 and Change rowLength=9 and columnLength=5. Run the program. What happens?

rowLength=9

columnLength=5

myArray=[[0 for column in range(columnLength)] for row in range(rowLength)]

print(myArray)

Amend the program as shown below which assigns values within the array. Run the program. What happens and why?

rowLength=9

columnLength=5

myArray=[[0 for column in range(columnLength)] for row in range(rowLength)]

myArray[0][5] = 99

myArray[2][3] = 74

print(myArray)

Make the following changes to the program which will print out the array a row at a time. Explain how it works.

rowLength=9

columnLength=5

myArray=[[0 for column in range(columnLength)] for row in range(rowLength)]

myArray[0][4] = 99

myArray[2][3] = 74

# print out a row at a time

for row in range(rowLength):

print(myArray[row])

**Activity 15.4**

**Summary sheet for Python and two-dimensional arrays**

|  |  |
| --- | --- |
| **How to use two-dimensional arrays in Python (nested lists which start from zero)** | |
| **Task** | **Example** |
| How to initialise a two-dimensional array |  |
| How to address an array element |  |
| How to assign values in a two-dimensional array. |  |
| How to print a two-dimensional array |  |

**Activity 15.5**

Write a program that fills up a two-dimensional grid with the results of the multiplication table 10 x 10 and prints out the result.

**Activity 15.6**

Write a program to implement a “one player” Battleships game. Use randint() to initialise the location of the ships. The player enters the index locations for their shots. The player scores one point for every direct hit on a ship. The player is allowed 10 “goes” to hit as many ships as possible. The score is displayed at the end of the game.**Lesson 16 activities**

**Activity 16.1**

**Validation**

Validation is the automatic checking of entered data by a computer program. Validation cannot check that the data entered is correct only that it is a reasonable value.

Complete the table which describes some of the types of validation used to reduce errors in input data. Give an example of each.

*presence range length type look-up*

|  |  |  |
| --- | --- | --- |
| **Type of validation** | **Description** | **Example** |
|  | Checks the data entered is not too short or too long |  |
|  | Checks that data has been entered. |  |
|  | Checks that a the value entered falls within a given range |  |
|  | Checks that the value entered is of the expected type |  |
|  | Checks the entered value is a value that is expected. Checks value against a look up list or string. |  |

**Activity 16.2**

**Validation: length check**

Write a program that asks the user to input a password and then uses a length check to make sure the password is at least 8 characters long. If it is shorter than 8 characters the user is asked to enter a different password.

*Hint: Use the len() function.*

**Activity 16.3**

**Validation: presence check**

Write a program that asks the user to enter a name and uses a presence check to make sure that an answer has been entered. If nothing has been entered the user is asked to re-enter.

Extension: Give an appropriate message when the user fails to enter any characters.

**Activity 16.4**

**Validation: type check**

Write a program that asks the user to enter their age and checks that they have entered an integer. It should display a message asking them to enter a number if they have not done so.

*Hint: Use the “try except else” command*

**Activity 16.5**

**Validation**

Write a program that asks the user to enter an email address and then checks the string entered to make sure it contains an “@”. If it does not the user is asked to enter the email address again.

**Activity 16.6**

**Validation: range check**

Write a program that asks the user to enter a % of charge left in their mobile phone. Use a range check to make sure the value is less 100% or more than 0%. The user is asked to re-enter the value if it is outside the range.

Extension: Alter the program so it only allows integer numbers to be entered.

**Activity 16.7**

**Try command: divide by zero error check**

Write a program that asks for two numbers and then divides the numbers and displays the answer. If the program generates a divide by zero error display a message to explain they enter a zero as the second number.

**Activity 16.8**

Write a program that asks the user to enter their name, age and email address using validation to ensure the data contains reasonable values. It should then display the data and ask the user if it is correct. Use validation to make sure the user can reply Y, y or Yes and N,n, or No. The user should be allowed to re-enter the data if it is incorrect.

**Python Command Set**

Remember: Python is case sensitive.

|  |  |
| --- | --- |
| **Built in function** | **Description** |
| len() | Length of a string |
| random() | Returns a random number |
| print() | Displays on the screen |
| type() | Displays the type (int, bool, str or real) of a variable or value |
| int() | Converts a value into a integer number.  number = int(input(“Please enter an integer number :”)) |
| input(“prompt”) | Displays the prompt and gets input from the screen and assigns to a variable.  answer = input(“Please enter your name”) |
| range() | Creates a list of numbers. Often used with the for loop. |

|  |  |  |
| --- | --- | --- |
| **Variables and arrays** |  |  |
| **Syntax** | **Description** | **Example** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **Selection** | | |
| **Syntax** | **Description** | **Example** |
| If <expression>:  <command>  <command> | If <expression> is true then commands are executed | if colour == "green":  print("you are safe to cross") |
| If <expression>:  <commands1>  else:  <commands2> | If <expression> is true then first <commands1> are executed otherwise second <commands2> are executed | if colour == "green":  print("you are safe to cross")  else:  print("STOP! it is not safe to cross") |
| If <expressionA>:  <commands1>  elif<expressionB>::  <commands2>  elif<expressionC>::  <commands3> | If <expressionA> is true then <commands1> are executed else if <expressionA> is true then <commands2> are executed etcetc | if answer == 1:  print("You will meet a stranger")  elif answer == 2:  print("Your favourite colour is blue")  elif answer == 3:  print("You enjoy programming") |
| try :  <commands1>  except:  <commands2>  else:  <commands2> | If <commands1> cause an error then <commands2> will execute. If the commands1 execute successfully then <commands3> will execute. | try:  ans=numOne/numTwo  except ZeroDivisionError:  print("Second number cannot be zero!")  else:  print("The answer is: ", ans) |

|  |  |  |
| --- | --- | --- |
| **Repetition** | | |
| **Syntax** | **Description** | **Example** |
| for variable in <expression>:  <command1>  <command2> | Executes <commands> for a fixed number of times given by <expression> | MyList=["cat","dog","cow","donkey","rabbit","canary"]  for next in myList:  print(next) |
| while <condition> :  <command1>  <command 2> | Executes the commands whilst <condition> is true. This is a pre-condition loop. | answer="N"  counter=0  while answer != "Y" :  print("are you hungry? You have been asked {0} times".format(counter))  answer = input("Please respond Y or N :")  counter=counter + 1  print("Please get something to eat!") |

|  |  |  |
| --- | --- | --- |
| **Input/output** |  |  |
| **Syntax** | **Description** | **Example** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| **File handling** | | |
| **Syntax** | **Description** | **Example** |
| variable = open (“filename”, “character showing way file is to be used” | Opens a file for reading, r, or writing,w. This creates a file object which is assigned to a variable. | myFile=open(“file.txt,r)  myFile=open(file.txt,w) |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| **HOW TO** | **Explanation** |
| Start the debugger | In IDLE shell debug/debugger |
| Save a file | File/save (Hint: type is .py ) |
| How to run a program | Run/run module |
| Display the last entered command in the shell | alt p |
| How to indent and dedent blocks of code | Select and use format/indent and format/dedent |
|  |  |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Data type** | **Python Abbreviation** | **Explanation** | **Example** |
| integer | int | A whole number | 45 |
| string | str | A sequence of characters which can include letters, spaces and other characters | “Have a nice day” |
| float | float | A number with a fractional part | 16.76 |
| boolean | bool | Boolean or logical data can only have one of two values. In Python they are either True or False. | True or False |

|  |  |
| --- | --- |
| **Escape sequence** | **Effect** |
| \t | tab |
| \n | new line |
| \\ | displays \ |
| \’ | displays ‘ |
| \” | displays “ |

|  |  |
| --- | --- |
| **Mathematical operator symbol** | **Operation** |
| / | divide |
| \* | multiple |
| \*\* | exponential |
| + | add |
| - | subtract |
| // | integer division |
| % | modulus (remainder after the division) |

|  |
| --- |
| **Precedence** |
| The parentheses control the order in which the numbers are calculated. Anything in parentheses is evaluated first.  The precedence order is: parenthesis (round brackets), exponential, division and multiplication, subtract and add  B E D M A S |

|  |  |
| --- | --- |
| **Type of Python errors** | **Description** |
| TypeError | When an operation is attempt that is invalid for that type of data |
| RuntimeError | An error occurs when the program is running |
| NameError | When a name is used that is not known about (often a misspelt variable name |
| ZeroDivisionError | Dividing a number by zero |
| KeyBoardInterrupt | When a program is interrupted from the keyboard by pressing control+c |

|  |
| --- |
| **Rules for variable names** |
| Must begin with a letter (upper or lower case) followed by zero or more other letters or numbers |
| Cannot have spaces in the name |
| Can include “\_” egMy\_variable |
| Reserved Python command words cannot be used |

|  |  |  |
| --- | --- | --- |
| **Reserved Python command words** | | |
| and | exec | not |
| assert | finally | or |
| break | for | pass |
| class | from | print |
| continue | global | raise |
| def | if | return |
| del | import | try |
| elif | in | while |
| else | is | with |
| except | lambda | yield |

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator meaning** | **Operator** | **Sample condition** | **Evaluates to** |
| Equal to | == | “fred” == “sid” | false |
| Not equal to | != | 8 != 8 | false |
| Greater than | > | 10 > 2 | True |
| Greater than or equal to | >= | 5 >= 5 | True |
| Less than | < | 40 < 34 | False |
| Less than or equal to | <= | 2 < = 109 | True |

|  |  |
| --- | --- |
| **Symbol** | **Description** |
| AND | Returns true if both conditions are true |
| OR | Returns true if any of the conditions are true |
| NOT | Reverses the outcome of the expression; true becomes false, false becomes true |

|  |  |  |
| --- | --- | --- |
| **Python command colour coding** | **What does it show** | **Example** |
| green | string | “hello” |
| purple | function | print() |
| Black | Variables and data | myName |
| orange | Key commands | if |
| red | Comment | # This is a comment |