

Python Workbook

## This Workbook

This workbook is very important, look after it carefully. When you do your controlled assessments you will be able to refer to it to help you.

## Help

**Ask** for help from **others**

Complete them in **sequence** and do not move on to the next challenge until you have successfully got the program running.

If possible try the extension tasks.

Make sure that you save your solutions to your Python Practice folder using the challenge title/topic as the file name**. py**

Look at the programs that you have completed in previous lessons for help if you struggle

## Get Organised

Life will be much harder if you don’t save your work carefully.

Create the following folders:







## Inputting information and stringsC:\Users\jwardell\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\92R9WSNQ\MC900331536[1].wmf

**print() function**

The print function prints information to the screen. Type this in:

print (“Hello World”)

**input() function**

The input function allows the user to enter information into the computer which the computer will save. Type this in:

name = input("Hi. What's your name? ")

print ("Hello, ", name)

**Variables**

A variable is what is used to store information. In the program above the variable is **name.** Because the variable is in quotes “”, the variable type is called a **string**.

**String**

A string can contain letters, characters and numbers.

### Challenge 1 - Write a program that….





Strings can be joined together:

### Challenge 2 - Write a program that joins two strings together …



**Task:** Joining strings together is possible. Find out what this is called and write the answer in your word document.



## Working with INT and FLOAT to handle numbers in Python

We have used the input command to ask the user to enter text. We’re going to use this again but with numbers. What do you Think what this code will do?

### Challenge 3 – Programming with numbers

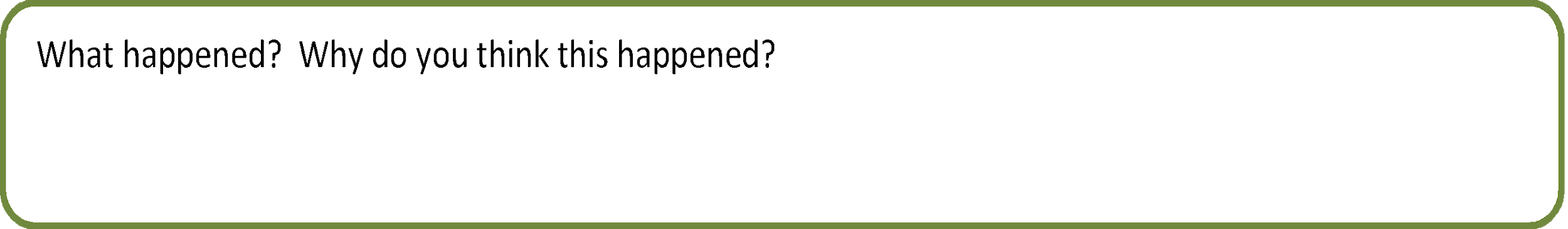












### Challenge 5 – I’m floating man….







### Challenge 6 – Calorie Counter

Now let’s build a calorie counter. The NHS recommends that an adult male takes on board 2,500 calories per-day and an adult woman takes on 2,000 calories per-day. Build your program for a woman or a man.

Save your program as **CalorieCounter**





Now try these challenges. Think carefully – what type of variable will you need to use?

Remember that by *default*  **variables** are strings.

### Challenge 7 – Area Calculator

Jennifer wants to carpet her new room with pink carpet. Create a program that will ask the user for the two dimensions of a room (length and width) and then calculate the area of the room, (length x width), before finally displaying the answer.

*Save as* ***Area***

### Challenge 8 – User Alive Calculator

Write a program to work out how long a user has been alive for to the nearest year for the moment - there are 365 days in a year



Get the program to ask for the user’s name and age.

Use the age variable value to work out how many hours that is – there is 24 hours per day

Develop the program further so that it can work out how many minutes and seconds the user has lived for – 60 minutes per hour / 60 seconds per minute.

Make sure all the information is clearly displayed on the screen.

*Save as Age*

Project 1 – Going Shopping

Write a program that monitors a user’s shopping trip and calculates how much the spent



Project 2 – Pizza

Pizzas come in various shapes and sizes. We can use python to calculate which pizza is the best in terms of size and price

The mathematical formula for area is 3.142 times radius squared if it’s a circle and length times width if it’s a rectangle.





## Some more about Strings

Strings are variables that contain letters, numbers and symbols. There are lots of things that can be done with strings.

### Challenge 9 – I am an excellent programmer



What is shown on the screen should be the same as you typed in.

Now try changing your programme to this:



This is known as a string **method**

|  |  |
| --- | --- |
| **String method** |  |
| lower() |  |
| upper() |  |
| title() |  |
| swapcase() |  |
| capitalize() |  |

Then instead of the word lower use the following methods, write down what each one does in your word document.

### Challenge 10 – Concatenation

Strings can be joined together – which is called concatenation.



Save as **Christmas Gifts**

### Challenge – If

Let’s write a programme that compares two numbers. 



This tests each combination and does something (prints a line out) if the test is **TRUE**

Python can do other things if the test is **FALSE**:

* *Do another test* – If the test is False we can get Python to another test: we use **elif** (short for “else if”)
* *Do something else* – If all the other tests come out false do something else: we use **else**

We can re-write the code to use these features:





Indenting is very important in Python, it’s a necessary part of how you write the code. Indenting tells Python where blocks of code start and where they end. The Python IDE will do this for you – just be aware that it is doing it.

For testing whether two things are equal Python uses a double equal sign (==) .



## How to use Python for basic math

Let’s look at how python calculates using numbers.

### Challenge 11 – Multiplication

Let’s see how much a student spends on food at school. Do you remember why we have to use int or float?



Save as Food spending

**Challenge 12 – Dividing**

An Aunt wins on the lottery. She gives £1000 to you. You are thinking of sharing it with some of your family. How much would each person get (try different numbers of people to share it with). To get you started:



If you want an answer to be a decimal, instead of typing 1000, try entering 1000.0. Try it to see what happens.

Save as: sharing my money out

### Challenge 13 – Modulus

If we divide two numbers and they don’t divide evenly we get a remainder. Modulus gives us the remainder of a sum.

For example 7/2 = 3 with remainder 1. Try this:



You should get the answer 1. Try some other numbers to see what you get.

### Challenge 14 – Addition

Addition is easy. Try this:



What’s the value of the variable length:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Try it out to test it.



Another way of writing this in python is:

Test it out. It’s quicker

## Operands

The +, -, /, \* symbols are called operators. That’s because they “operate on” the numbers we put around them.

|  |  |  |
| --- | --- | --- |
| **Operator** | **Example** | **Is Equivalent to** |
| \*= | K\*=5 | K = K \* 5 |
| /= | k/=5 |  |
| %= | K%=5 |  |
| += | K+=5 |  |
| -= | K-=5 |  |

### Challenge 15 - Restaurant Tip

Write a program that calculates the answers to these questions and displays the results



1. Two people eat dinner at a restaurant and want to split the bill.
2. The total comes to £100 and they want to leave a 15% tip. How much should each person pay?
3. Make the program ask how many people there are, what percentage the tip should be and how much the bill comes to.

Save this as Restaurant tip.

**Comments**

Comments are added to a programme to explain what the programme does and what the section of code is doing. Programmers do it so that if the code is looked at in the future (either by themselves or others) they can work out what the code is doing.

Comments can easily be added using # at the start of the line of code. Python ignores anything on a line that begins with a #.

Example

|  |
| --- |
| *# this programme creates a random number between 1 and 10*  *# the user guesses the number. It ends when the user guesses the number correctly*  import random |
| guess ="" |
|  |
| print ("I've thought of a number between 1 and 10. Try to guess it.") |
|  |
| randomNumber = random.randrange (1,10) |
| *# this loop runs until the number guessed equals the randomNumber* |
| while guess != randomNumber: |
| guess = int(input("Guess the number")) |
| print("You got it wrong") |
|  |
| input ("Well done. Press any key to exit.") |

## Making decisions in programmes

Python can make decisions based on the input. To make decisions, programs check to see **if** a condition is true or not.

Python has a few ways to test something, and there are only two possible answers for each test: ***true* or *false***

### Challenge 16 – Magic 8 Ball

The code below will use Python’s **random number generator** to make a magic 8 ball game.

 Make this game and save it as **Magic8Ball.**





Write down in your word document what each of the below lines are doing, for example:

|  |  |
| --- | --- |
| **Programme** | **What it does…** |
| import random  answer1=(“Absolutely!”)  answer2=(“No way Pedro!”)  answer3=(“Go for it tiger.”)  print(“Welcome to the Magic 8 Ball game—use it to answer your questions...”)  question = input(“Ask me for any advice and I’ll help you out. Type in your question and then press Enter for an answer.”)  print(“shaking.... \n” \* 4)  choice=random.randint(1,3)  if choice == 1:  answer=answer1  elif choice == 2:  answer=answer2  else:  answer=answer3  print(answer) |  |

|  |  |  |
| --- | --- | --- |
| **Comparison Operators** | **What the symbols mean** | **Example** |
| == | are two things equal? | firstName == “Gary” |
| != | are two things not equal? | firstName != “Gary” |
| < | less than | income < 300 |
| > | greater than | income > 300 |
| >= | greater than OR equal to | income >=300 |
| <= | less than OR equal to | income <= 300 |

### Challenge 17 – Mega Sale



A local shop is having a promotion. If you spend over £10 you will get a £1 voucher to spend next time you come in the store. If you spend over £20 you get a £3 voucher.

Write a programme to tell the sales assistant which voucher to give the customer.

Save this as: megasale

### Challenge 18 – Happy Message



Write a programme that gives users a message depending upon how happy they say they are.

You could get the user to rate how happy they feel on a scale between 1 and 10. If the reply is 3 or less it gives one message.

Between 4 and 7 (including these numbers) they get another message.

8 and above they get a different message.

Try to make the messages ones to make them happy all day long!

Save this as: Happy message

### Challenge 19 – Mobile Phone Costs



You want to see how much a mobile phone will cost. There are charges for sending pictures (£0.35), for texts (£0.10) and for data (£2.50 for 500MB).

1. Write a program that asks the user for how many pictures, texts and data they would use each month. It should then calculate a total bill for the month.
2. If the total comes to more than £10 they would be better on a contract. Get the programme to give them this advice.

Save this as

### Challenge 20 – Secret password



Make a program where the user has to enter a secret password to use the programme.

The programme could be one you’ve already written, or make it display a message when they have got the password correct.

## Boolean or logical expression

Boolean is a type of arithmetic that only uses two values: true or false, yes or no, 1 or 0.

It was invented by an English mathematician George Boole.

We use Boolean expressions and can combine them with **and**, **or** and **not** to make decisions in our programmes.

### Challenge 21 – For loops

Sometimes we want to do something a number of times.

We may know how many times we want to do it – and we use a **counting loop** OR

We may want to do it until something happens – and then we can use a **conditional loop.**

## Conditional Loops

This is the simplest loop. Try these out and make a note in your word document of what they do:









To get out of a loop you can use the command break e.g. if k == 7:  
print 'breaking out of the loop!'  
break

### Challenge 22 – For loops

Write a loop that displays numbers 10 to 100 in steps of 5.

### Challenge 23 – For loops

Write a loop that displays the 5 times table

Save as 5 timestable.

### Challenge 24 – While loops

All of this programme should be familiar to you apart from the while loop. Write down what each line does and then type in the code.

|  |  |
| --- | --- |
| import random  guess =""  print ("I've thought of a number between 1 and 10. Try to guess it.")  randomNumber = random.randrange (1,10)  while guess != randomNumber:  guess = int(input("Guess the number"))  print("You got it wrong")  input ("Well done. Press any key to exit.") |  |

**Save as**: guess a number 1 to 10





The **while** loop repeats whist the conditions is **TRUE**.

In this case this is when guess is **NOT** equal to randomNumber.



**Another example for you to try:**

import random

print ("you will be given something that will help you out of here but it’s not as easy as you think")

selection = random.choice(["key","sword","rope"])

if selection == ("key"):

print ("here is your key use it wisely as it only opens one door")

elif selection == ("sword"):

print ("here is your sword incase you run into some trouble use it wisley as it might break")

elif selection == ("rope"):

print ("here is a rope find where to use as it will come in use one time")

else:

print ("bye")Some further challenges for you to have a go at. Use this workbook and the programmes you have already used to help you solve the problems.

### Challenge 25 – Rock, Paper, Scissors Game



We’ve all played the rock, paper, scissors game. The computer will play against you.

Get the computer to ask for the player’s name.

The game rules are simple: rock beats scissors, scissors beat paper, paper beats rock. Two of the same gives a draw.

An idea to get you started

You could use pseudo code to help you identify the steps needed.

Add **comments** to the game so it’s clear what the code in the game is going.

**Extension:** Ask the user how many rounds they want to play, between 3 and 10.

Keep score and show this at the end of the game.

**Further extension:** Check to make sure the user can only enter a number between 3 and 10 and give them an error message.

### **Using AND and OR**

You can also represent more complicated options using AND and OR

For example:

x=45 x=45 x=45 x=45

while X= <50: while x >45 and x<47 while x<50 and x<47 while x<50 or x<47

print (x) print (x) print (x) print (x)

x +=1 x +=1 x +=1 x +=1

would return 45,46,47,48,49 would return nothing would return 45,46 would return 45,46,47,48,49

### Challenge 26 – Making a times table (using nested if)

With a times table we start with a number and then multiply this by a series of numbers.

For the 2 times table we multiply it by 1 to 12, before moving on to do the 3 times table, etc.

Computers are very good at this sort of task and we can do it using loops within loops to consider every combination – these are called **NESTED LOOPS.**

Type this code in (try to predict what you will get before you do it). 

**for i in range(1,13):** # i is the first number we are going to multiply by

# print a title at the top of the times table to show which times table

# we are working on

**print (i, "Times table\n")**

**for j in range (1,13):** # loop through the numbers we are multiplying

# i by and then print the results

**print (i, "times", j, " = ", i\*j)**

## Extension:

Maths Teachers will be impressed if you’ve worked out the 1345 x 1 to 12 times table - change the programme to work out your own different times tables.

### Challenge 27 – Chicken Wraps (using nested if)http://www.oldelpaso.co.uk/~/media/Images/OEP_UK/Recipes/Original-smoky-bbq-fajitas2.ashx

Students can choose to have any combination of ingredients in their chicken wrap.

Write a programme to calculate the price of each combination of Chicken Wraps.

|  |  |  |  |
| --- | --- | --- | --- |
| **Tortilla**  **http://0.tqn.com/d/mexicanfood/1/0/l/1/tortilla.jpg** | **Chicken**  http://t1.gstatic.com/images?q=tbn:ANd9GcQIbU1LLt2Qw1fLxQd_nbg9Gj8qS6X1hMK0-SzT1yV8T-TLylVFjCuHKkuv | **Salad**  http://t2.gstatic.com/images?q=tbn:ANd9GcSisgWZnXEfBwFIxrLai4mnGoqvB7xwENcoUohyBqP3By_OCVnNz0Xd4Rx-fw | **Chilli Sauce**  http://t1.gstatic.com/images?q=tbn:ANd9GcTS66-BN_GijYualEBhJgEf-UgEd5cc8rOri7b9Kd4rz9gt4-JXvhI-Bq67bQ |
| **£0.25** | **£0.55** | **£0.12** | **£0.24** |

To start with see what this code does. It uses loops within loops to consider every combination – these are called **NESTED LOOPS.**

print("\tTortilla \tChicken \tSalad \tChilli Sauce")

count = 1

for tortilla in (0,1):

for chicken in (0,1):

print("#", count)

print ("\t\t",tortilla, "\t\t", chicken)

count =count+1

Note: ***The \t is an instruction to Python to tab the data – that is put it into columns***

Develop this to that it creates all combinations for Salad and Chilli.

Test the programme to see if the results are as you would expect. You will have to do some manual calculations and work out the costs ar.

## Extension

Add another column labelled cost. Calculate the total cost of the ingredients in each option.

Think carefully before you do this – there is a very simple way, and much longer ways.

Test it. You will have to do some manual calculations and work out what the costs should be.

## Functions

Programming languages have pre-made functions for us to use. Python has some such as print() or random.

But we can make our own functions, these are called user-defined functions:

A function is a block of organised, reusable code that is used to perform a single action.

By using functions we can make our programs modular (made up of separate parts) and simpler, because we don’t have to write the same instructions many time but reuse the same code.

### Challenge 27 – Area or Perimeter

Let’s make a programme to calculate the area **OR** the perimeter of a rectangle.







We can define a function to calculate area when we need it and one to calculate the perimeter when we need it:



def area():

shapeArea = length \* width

print("Area = ",shapeArea)

def perimeter():

shapePerimeter = length\*2 + width\*2

print ("Perimeter = ", shapePerimeter)

Look at the code and think what it is doing. Then type it into Python and run the code. Write your own explanation.

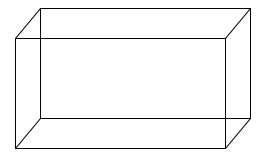
|  |  |
| --- | --- |
| **Code** | **Explanation of the code** |
| length = 200  width = 100  def area():  shapeArea = length \* width  print("Area = ",shapeArea)  def perimeter():  shapePerimeter = length\*2 + width\*2  print ("Perimeter = ", shapePerimeter)  while response not in ("a","p"):  response = input("Do you want to calculate area or perimeter? Enter a or p" )  if response == "a" or "A":  area()  elif response == "p" or "P":  perimeter() |  |

Add your own comments to the programme to explain what the code is doing.

## 

**Extension**

Have the programme ask the user to input their own measurements.

Add a function to calculate the volume.

To work out the volume of a regular cuboid shape is:

length \* width \* height

Add an option to the programme to calculate the volume defining a new function called volume()

**Working with lists**

A list is different to a string or an array in that it allows you to identify and manipulate individual elements of a variable.

A list is denoted using the [ ] brackets.

Example fruit=[“apple”, “banana”, “pear”]

The numbering system used by lists is also a bit odd in that it starts counting at 0.

In the example above apple would be fruit[0] – this is the 1st item in the list

pear would be fruit[2] - this is the 3rd item in the list

In python, lists have some built in functions

**Len**

In the above example len(fruit) would return a value of 3 because there are 3 items in the list

**Max**

Example:

numbers[5, 4, 10, 30, 22]

print(max(numbers)) would return the value 30 as it is the biggest number in the list

**Min**

Example:

numbers[5, 4, 10, 30, 22]

print(min(numbers)) would return the value 4 as it is the smallest number in the list

**Change**

Example:

numbers[2]=77 would alter the value of the 3rd item on the list from 10 to 77

**Delete**

Example:

del numbers[2] would delete the 3rd item from the list

**Replace**

Example:

example=list(“sandwich”)

example[4:]=list(“baby”) would return the string “sandbaby” as the instruction is to replace from

item 5 to the end of the list, with the word baby

**Clear**

Example:

numbers[5, 4, 10, 30, 22]

numbers.clear()

print (numbers) would return nothing as the instruction was to clear off the whole list

**Copy**

Example:

tuna=ages.copy() copies the contents of the ages list into another called tuna

**Append**

Example:

numbers[5, 4, 10, 30, 22]

numbers.append(45) would add the value 45 onto the end of the list

print(numbers) would return 5,4,10,30,22,45

**Count**

Example:

numbers.count would return the value 6, as there are 6 items in the list

**Extend**

Example:

one = [1,2,3]

two =[4,5,6]

one.extend(two) adds all the items from the list two onto the end of the list one

print(one) 1,2,3,4,5,6

**Index**

Example:

a=[“hey”,”now”,”brown”,”cow”]

a.index(“brown”) would return 2, as brown is the 3rd item on the list

**Insert**

Example:

a.insert(2, “horse”) inserts the item horse into the list

print(a) “hey”,”now”,”horse”,”brown”,”cow”

**Pop**

Example:

a=pop(1) takes now, the 2nd item, out of the list

print(a) “hey”, “horse”,”brown”,”cow”

**Reverse**

Example:

a.reverse swaps the contents of the list around

print(a) “cow”,”brown”,”horse”,”hey”

**Remove**

Example:

a=remove(“brown”) takes the item “brown” out of the list

print(a) “cow”,” horse”,”hey”

**Making use of the list index**

Lists can sometimes be very long so there are a number of index features to make lists more helpful

Remember the index counter in lists starts at 0 so the third item in a list is indexed as [2]

Example:

names=[“Alf”,”Betty”,”Charlie”,”David”]

print (names[2]) returns “Charlie” as it is the 3rd item on the list

Notice: the close variable (round) bracket on the print statement is after the close

Example:

print (names[2:3]) returns “Charlie” because the instructions is to start at the 3rd item and stop before the 4th item

print (names[2:4]) returns “Charlie”,” David” because the instruction is to start at the 3rd item and stop before the 5th item, even though there isn’t actually a 5th item

print(names[-3:-1]) returns “Betty”,” Charlie” because the instruction is to start at the 3rd item from the end (-) rather than the beginning and end before the last item

print (names[-3:]) returns “Betty”,” Charlie”,” David” because the instruction is to start at the 3rd item from the end (-) and continue indefinitely and past the last item on the list

print (names[:2]) returns “Alf”,”Betty”, because the instruction is to start at the start of the list and stop before the 3rd item

print (names[:]) returns all the items on the list

print (names[0:3:2]) returns “Alf”,”Charlie”, because the instruction is to start at the 1st item on the list and stop before the 4th item, displaying :2 only every other result i.e. items 1 and 3 in this example

You can also calculate items on lists

Example:

listOne =[128,64,32,16,8,4,2,1]

listTwo =[1,0,1]

result =listOne[0]\*listTwo[0] In this example the instructions are to multiply the contents of item 1 from listOne by the contents of item 1 from listTwo