Computing @ LMS

OCR Computing

Log Book

2014-16

Name: Click here to enter text.

Computing @ LMS

Table of Contents

Contents

Computing @ LMS

Computing @ LMS

Input

What do we use the input command for?

Click here to enter text.

Copy and paste an **example** of Python code below. Show getting the user to input a string, a float and an int:

string:

Click here to enter text.

float:

Click here to enter text.

int:

Click here to enter text.

Write into the table what these terms mean.

Term	What it means	Example
string	Click here to enter text.	Click here to enter text.
float	Click here to enter text.	Click here to enter text.
int	Click here to enter text.	Click here to enter text.

Variables and Constants

This website has some good definitions on – don't copy and paste write in your own words.
http://www.computerhope.com/jargon.htm

Term	What it means	Explain how and why you might use this in a program
variable	Click here to enter text.	Click here to enter text.
constant	Click here to enter text.	Click here to enter text.
boolean	Click here to enter text.	Click here to enter text.

Strings

Joining strings together is known as:

Click here to enter text.

Copy and paste an **example** of Python code below showing **joining two strings together**:

Click here to enter text.

What do these **string methods** do?

String method	Description of what it does
lower()	Click here to enter text.
upper()	Click here to enter text.
title()	Click here to enter text.
swapcase()	Click here to enter text.
capitalize()	Click here to enter text.

Python - basic math

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	Click here to enter text.
%=	K%=5	Click here to enter text.
+=	K+=5	Click here to enter text.
-=	K-=5	Click here to enter text.

Write in what these are equivalent to

Formatting how numbers prints out

Sometimes we want to show the user a number printed to a certain number of decimal places. We can increase or decrease the number using this method. **Note:** the number stored isn't changing, just how it is shown on the screen.

Command	What this does

`nOne = 3.123456789`
`nTwo = 9.87654321`
`ans =(nOne*nTwo)`
`print (ans[:5])`
	Click here to enter text.

`n3 = 1.1` `print ("%0.2f"% (n3))`	Click here to enter text.

Random module

Sometimes we want Python to make a random choice. We import the Random module first and then can use commands to use this module.

import random is placed at the start of the program.

Program	Write down precisely what this code does
`choice=random.randint(1,3)`	Click here to enter text.

Towards the end this shows how to select a random entry from a list:

Program	Write down precisely what this code does

`letter = ['a', 'b', 'c', 'd', 'e']`
`from random import choice`
`print (choice(letter))`
	Click here to enter text.

Conditional Statements – if / elif / else

A Conditional Statement is one where the program can branch off to perform one of a number of different tasks depending upon a Boolean test. You can specify that the test is true / not true and use and / or.

Program	Explanation
Paste in the code from one of your programs that uses an if / elif / else statement below and then explain what it is doing	
Click here to enter text.	Click here to enter text.

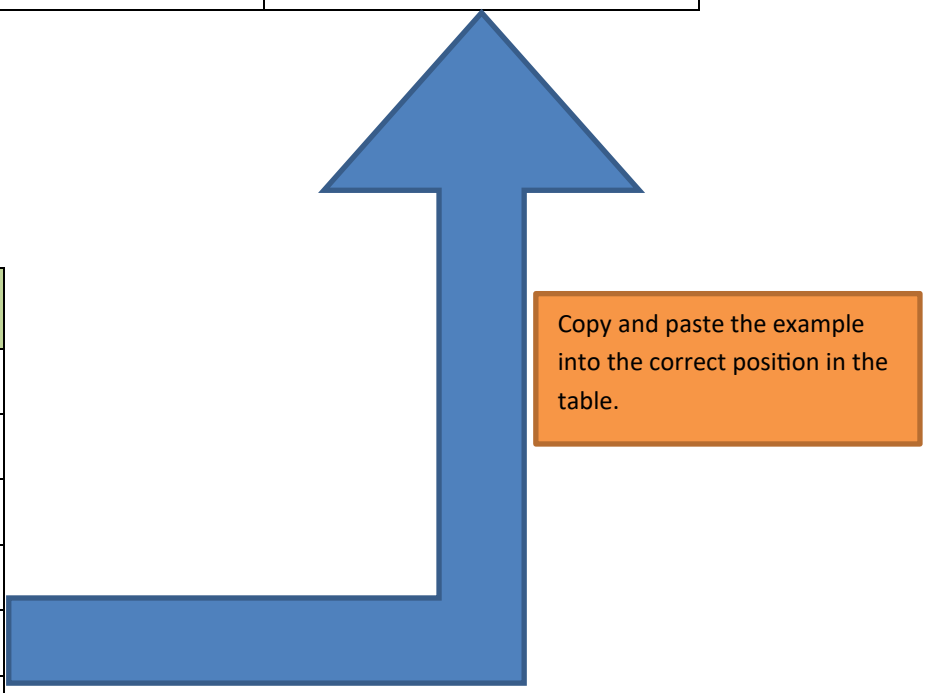
Computing @ LMS

Comparison Operators

A comparison operator allows us to compare two values. We can compare variables, constants, numbers or strings. It is one of the more often completed tasks.

Comparison Operators	What the symbols mean	Example
==	are two things equal?	Click here to enter text.
!=	are two things not equal?	Click here to enter text.
<	less than	Click here to enter text.
>	greater than	Click here to enter text.
>=	greater than OR equal to	Click here to enter text.
<=	less than OR equal to	Click here to enter text.

Example
income <= 300
firstName != “Gary”
income > 300
income >=300
firstName == “Gary”
income < 300



Computing @ LMS
Indenting

Indentation is very important in Python. Indention is where a piece of code is written in from the left hand side. It tells Python which code is part of the same: if statement / while loop / for loop / function. Indenting tells Python where blocks of code start and where they end.

```
#ROUNDS CHECKER:
check = 1
while check == 1:
    rounds = int(input("How many rounds do you wish to play (3-10)? "))
    if rounds <= 10:
        if rounds >= 3:
            time.sleep(1)
            print("You are playing: ",rounds," Rounds")
            time.sleep(2)
            os.system("CLS")
            check = 0
        else:
            print("ERROR: Choose between 3 and 10")      #ERROR number sm
            time.sleep(1)
    else:
        print("ERROR: Choose between 3 and 10")          #ERROR number la
        time.sleep(1)
```

These are in-line. As a block of code they will run when if rounds >=3 is **TRUE**

These are in-line. The if / else are a pair. They tell Python that if:

rounds >=3 is **FALSE**

then the code underneath the else command will run.

Now on the next page copy in your own program and put vertical coloured lines in where you can see indentation in use by Python. You can do a screen print and copy in the code.

Indenting

This is my program that uses indentation to identify the start and end of sections of Python code.

You can print screen the code and add lines to the image rather than copying and pasting the code.

Click here to enter text.

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.

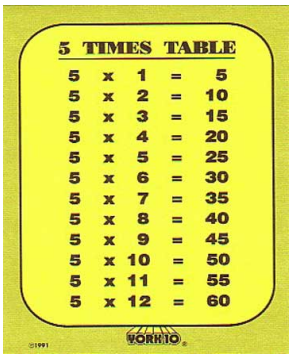
We use Boolean expressions and can combine them with **and**, **or** and **not** to make decisions in our programmes.

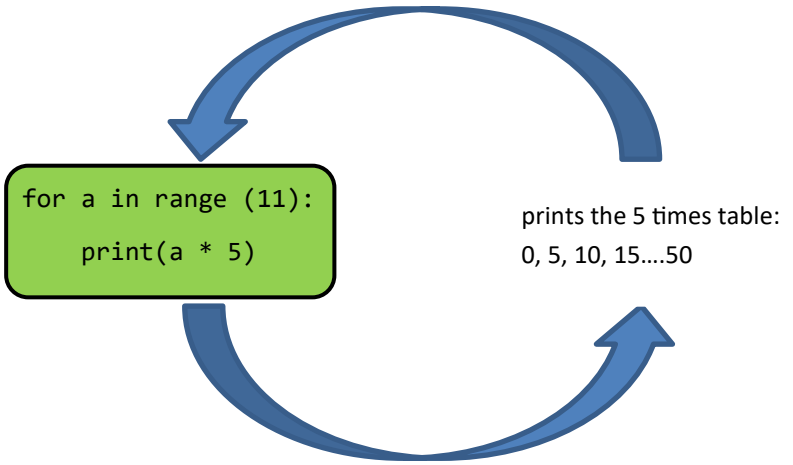
Computing @ LMS

For loops (counting loops)

Sometimes we want to do something a number of times. This process of doing code repeatedly is called **iteration**.

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





Write down what these bits of code do….

```
for a in range (1, 10):
    print(a)
```

Click here to enter text.

```
for a in range (1, 11):
    print(a)
```

Click here to enter text.

```
for a in range (1, 11,2):
    print(a)
```

Click here to enter text.

for a in range (1, 11,2):

The 1 tells the loop:

Click here to enter text.

The 1 tells the loop:

Click here to enter text.

The 1 tells the loop:

Click here to enter text.

Clever for loops

```
sentence = "Here is a sentence. How many letter 'e''s are there in it?"
numberE = 0


for letter in sentence:
    if letter =="e":
        numberE +=1


print ("The number of e's in the sentence is:", numberE)
```

Write down what this for loop is doing:

Click here to enter text.

Conditional Loops - While Loop

We may want to do some task **until** something happens – and then we can use a **conditional loop**. These are really useful for checking what is being typed into our program by the user.

Here is a practical example of using a conditional loop you may be familiar with...

```
while items left to wash up:
    select item
    wash it
    place on drainer
```

or in other words wash everything **until** nothing is left..

Put in here a print screen (or a copy of the code) here of challenge 29 that uses a while loop (if you need to crop it do it before adding it here):

Click here to enter text.

Explain what the **while loop** is doing in the program (remember a while loop only progresses to the next code when it is true):

Click here to enter text.

The **while loop** is validating the entry – in this program it is a range check. Explain what **validation** is:
(hint: use the *Computer Jargon Buster* on Myvle)

Click here to enter text.

Explain the while loop in this program and what it is doing:

Code	What it does

`# Enter a number between 1 and 10`
 	
`number = 40`
 	
`while number not in range (1,11):`
 	
`    try:`
 	
`        number = int(input("Please enter a number between 1 and 10"))`
 	
`    except:`
 	
`        print("ERROR invalid input. Out of range or wrong type of data.")`
 	
`print("Thank you I have recorded your entry as :", number)`
	Click here to enter text.

Nested for

We can use a loop within another loop to consider every combination – these are called **NESTED LOOPS.**

```
for i in range(1,13):
    print (i, "Times table\n")
```

This is the first **for** loop

```
    for j in range (1,13):
        print (i, "times", j, " = ", i*j)
```

This is the second for loop.

It runs from inside the other for loop

So we say they are nested together.

A bit like school chairs that sit one inside the other they are **nested** together.

Put in a print screen of your own program that shows nested statements..

Click here to enter text.

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.

```
def area():
    shapeArea = length * width
    print("Area = ",shapeArea)
```

Print screen evidence of one of your programs that uses a function:

Click here to enter text.

Code	Explanation of the code
`length = 200`	Click here to enter text.
`width = 100`	Click here to enter text.
`response = None`	Click here to enter text.

`# function to calculate area`
`def area():`
`    shapeArea = length * width`
`    print("Area = ",shapeArea)`
	Click here to enter text.

`def perimeter():`
`    shapePerimeter = length*2 + width*2`
`    print ("Perimeter = ", shapePerimeter)`
	Click here to enter text.

`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()`
	Click here to enter text.

Function Parameters (or arguments)

We have learnt how useful making our own functions can be. We can also pass values of variables around using them, which is a very useful thing to be able to do.

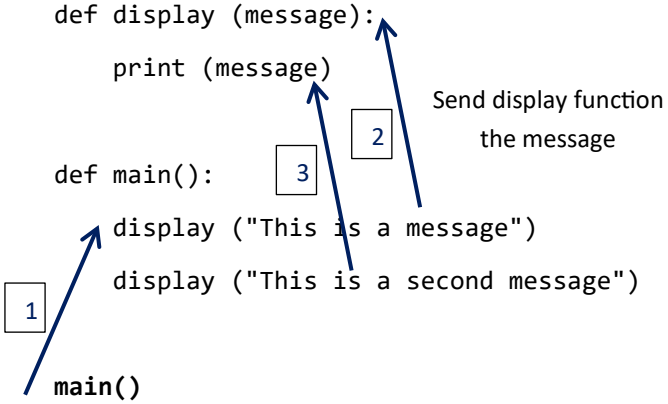
Functions in other programming languages are also known as sub-routines.

Method 1

```
def display (message):
    print (message)


def main():
    display ("This is a message")
    display ("This is a second message")


main()
```



Arrows and numbers have been added to show the sequence of operations.

This one has been done for you.

Do the same thing for the next function methods.

Method 2

```
def give_me_a_number():
    gmaNumber = 78
    return gmaNumber


def main():
    number = give_me_a_number()
    print ("Here's the number I got from the give_me_a_number_function: ", number)


main()
```

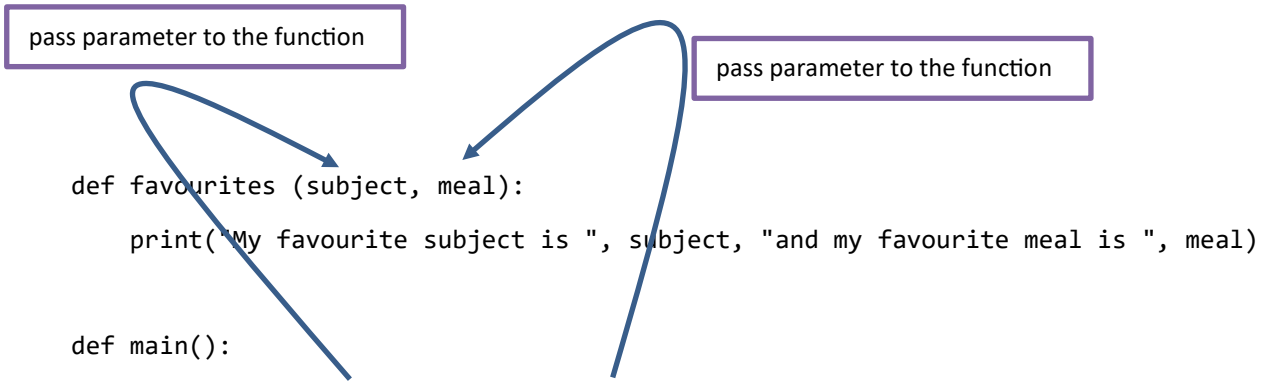
Method 3

```
def ask_yes_no(question):
    response = None
    while response not in ("y","n"):
        response = input(question)
    return response

def main():
    answer = ask_yes_no("\nPlease enter y or n: ")
    print ("Thanks for entering:", answer)
    input ("\nPress the key to exit.")

main()
```

Method 4




```
def favourites (subject, meal):
    print("My favourite subject is ", subject, "and my favourite meal is ", meal)

def main():
    favourites ("Computing", "Burgers and Chips\n")
    favourites  (meal = "Chicken wraps", subject="Computing")

main()
```

Computing @ LMS

Print screen evidence of one of your programs that uses a parameter within a function:

Click here to enter text.

Lists

Lists are known as arrays in other languages.

Here is an example of a list:

mondayTT = ["French", "English", "PSHE", "ICT", "Maths"]

mondayTT is the list name and the contents of the list are in the following positions:

MondayTT =[	"French"	"English"	"PSHE"	"ICT	"Maths"
Position	0	1	2	3	4

Note: Lists always start with 0.

Challenge 35

Type this code in. Run it and then explain what it does.

Code	Description
print ("The list monday contains the following information:")	
print (Monday[2]);	Click here to enter text.
print ("We can also just ask to print out the list. So here is monday:") print(monday)	Click here to enter text.
print ("We can also print them out one at a time") for i in monday: print (i)	Click here to enter text.

We can also add to lists, very easily in fact!

Add this line after the first line that defines Monday: monday.append ("Geography")

What position is geography in now? Click here to enter text.

Computing @ LMS

Print screen a program of yours that uses a list:

Click here to enter text.

More lists

Slicing lists

Try typing these in and write down in your log book what Python is doing:

Instruction	What it does
print (bars[:2];	Click here to enter text.
print (bars[2:]);	Click here to enter text.
print (bars[:]);	Click here to enter text.

Adding to a list:

Instruction	What it does
bars.append(“Snickers”); print (bars)	Click here to enter text.
bars.insert(1,”Yorkie”); print (bars)	Click here to enter text.
bars.extend(["Picnic","Twirl"]) print (bars)	Click here to enter text.

Note: If you want to add more than one item to a list use extend.

Deleting from a list:

Let's start with a list. Enter this into a new programme: bars = ["Mars" , "Bounty" , "Twix", "Yorkie", "KitKat"]

Then add the following instructions and see what they do (the **print(bars)** option is there just so you can see what has happened). You can add each command after the previous ones:

Instruction	What it does
bars.remove("Twix") print (bars)	Click here to enter text.
del bars [0] print (bars)	Click here to enter text.
myBar = bars.pop() print (bars) print (myBar)	Click here to enter text.
myBar1 = bars.pop(0) print (bars) print (myBar1)	Click here to enter text.