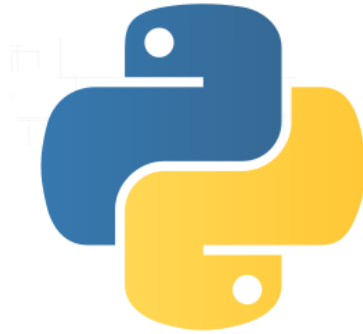


Python 2



KS3 Programming Workbook

"Taking you Parseltongue further"

Name _____

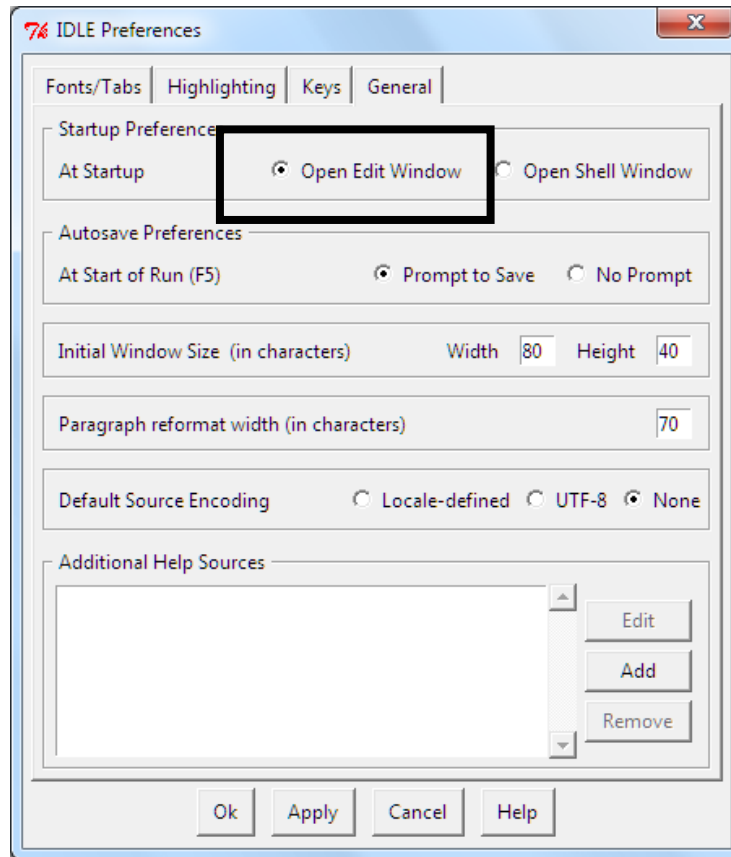
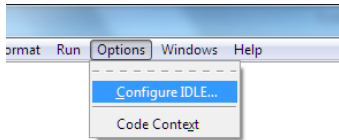
ICT Teacher _____ Form _____

To Execute the program code press F5

Welcome to Python

The python software has two windows that we will use. The main window is called the Python Shell and allows you to directly write in program lines and then a return will execute them. This is good for testing one line of code. The second window is the “Edit Window” which allows you to enter several lines of code, or a create a program, save it and then execute the code.

Open the python software



Python and the Raspberry Pi

To create a new python program on the Pi, you can use a text editor called “Joe’s Text Editor”

Type: `joe (the name of your program).py`

To run or execute the program type: `python (the name of your program).py`

KEYWORDS to complete

Time.sleep		
\		

Making the Program Exe.

In the exercises in this workbook you may wish to run your code outside the Python shell. This is a simple process. Save your program with the file extension `.py`. When you run the file it will execute the program in a command box style window. Once the program has executed it will close the window. You may wish to build a delay feature into the program to allow time to view the output results of the program. Use the **`import time;`** function and the **`time.sleep(5)`**.

- Create a simple program and save as an executable file, then run, Save as *simple*
- Create a program, add the time function to delay the closing of the window. Save as *simple2*

Time Functions.

Python can do more with the inbuilt time functions. The time function needs to be imported first and then coded to print or work with the data. The import function is as simple as `import time;` (note the semi colon).

ENTER THE FOLLOWING CODE:

```
import time;
x = time.time()
print x
```

What do you notice about the result, is it correct?

To get the current time you need to use the code, `localtime`

TRY THE CODE BELOW:

```
import time;
x = time.localtime(time.time())
print x
```

Yet we still do not have the time in the standard format. You can format any time as per your requirement, but simple method to get time in readable format is **`asctime()`**:

TRY THE CODE BELOW:

```
import time;
localtime = time.asctime( time.localtime(time.time()) )
print "Local current time :", localtime
```

A mouthful to remember but it shows you the programming required to display the time in a suitable format.

Calendar Functions.

Python also has a calendar function, it operates in a similar way to the time function. First import the calendar function using `import calendar`. A lot simpler to remember than the time function.

TRY THE CODE BELOW

```
import calendar
cal = calendar.month(2008, 1)
print cal;
```

Describe the results

Can you work the code for the current month? Write your code below.

What happens if you add another number after the month, for example `cal = calendar.month(2008, 1, 6)`

Printing onto a new line

Sometimes you will want to create text or objects on a new line, this is simple, just use the `\n` before the text and the text will appear on a new line.

```
print "hello"
print "\nHello world"
```

Definitions - Creating your own functions.

Definitions allow you to create a set of instructions and store them as a single word. This is useful if the instruction requires several lines of code. In the example below, the definition is called `greeting`, the second line tells python what the `greeting` definition does, it returns the phrase 'Hello and the name that you enter.

TRY THE CODE BELOW:

```
def greeting(name):
    return 'Hello, ' + name + '!'
print greeting('world')
print greeting('Gumby')
```

- Create a program that uses a definition but allows the user to enter any word and it returns the defined phrase. Save as *Def*

Rounding up Numbers.

Ever wanted to round up a number? Probably not!
Use **round** if you want to round a number up for example,

round(3020.2323)

Pass Blocks.

In simple terms a pass block passes or misses out an instruction if a condition is met. Basically it does nothing.

TRY THE CODE BELOW:

```
month = 3
if month == 1:
    pass
elif month < 6:
    print "early"
else:
    print "late"
```

Try changing the month value to 10, what happens.

Try changing the month value to 1 what happens.

- Can you create a program for all the numbers 1 - 1000 that looks to see if they are divisible by 6, and then prints them if not pass. (Use the range function from book one)
- Create one of your own

List Functions and Appending Strings

An *append* is an adding to a list. Append will add the data to the end of the list, append means the data or text will be added to the end of the list. This useful if you are creating one list with a number of inputs.

TRY THE CODE BELOW:

```
list = ['a', 'b', 'c', 'd']
li.append(33333)
print list
```

- Create a list called **num** with the numbers 1 to 10, then append with the numbers 11 to 20. Write your code below.

- Now try your code...

Save as MyList

Inserting

You may need to insert the data or the elements in a string for example a,b,c,d needs to be inserted with a,b,c,d, example. To do this there are several methods,

TRY THE CODE BELOW:

```
li = ['a', 'b', 'c', 'd']
li.insert(4, "example",)
print li
```

You can also use the simple code of a[*position of the data*] = [*data to be entered*]

TRY THE CODE BELOW:

```
a = [1,2,3,4,]
a[1:1] = ['eggs', 'ham']
print a
```

(remember text needs to be entered using the ' ' whilst the numbers can just be added)

This inserts the word or data after the position ion the list. It places the word example after the fourth item in the list

- Can you create a program that will add two more items to the list, *SaveasInsert*
- Create a program that will add two more items to the list and onto each line *SaveasLine*

Extending a list

A list can be very easily extended. This means data is added to the list. In the example below the list consist of 1 and 2. The extend adds 3, 4 and 5 to the list.

```
L = [1, 2]
L.extend([3,4,5])
print L
```

Removing Data From A List

Sometimes you may be required to remove data from a list. For example, if you want to remove spam and eggs from the list use the = [] function. Basically [] is empty and therefore the data is equal to nothing.

TRY THE CODE BELOW:

```
a = ['spam', 'eggs', 100, 1234]
a[0:2] = []
print a
```

The line **a[0:2] = []** will remove the data in positions 0,1, and 2 for example the “spam” and “eggs”

Replacing data in a list

This function is the same as above accept that the final [] has the data included. This means that the data spam and eggs is replaced with the numbers 1 and 12.

TRY THE CODE BELOW:

```
a = ['spam', 'eggs', 100, 1234]
a[0:2] = [1, 12]
print a
```

- Can you create a program that first prints the list of four items and then allows the user to add two items of their choice to the list.
- The new list is then printed out? *SaveasLister*

What have you enjoyed so far?

What have you found difficult?

What would help?

What do you still need to improve?

What will you do next?

HANGMAN (from David Bau, <http://davidbau.com/python/learning.html>)

In this section we will look at how to create a simple hangman game. Thinking about the game we need to consider some of the features required. For example importing random words, picking one word, displaying a dash for each letter to hide it, allowing the user to guess a letter, if the letter is correct then display the letter, if not, then displaying a part of the hangman. Finally, only allowing the user so many turns say maybe five?

```
import random

print 'time to play hangman'
animals = (list the animals in the game for example, dog, cat, frog, mouse)
secret = random.choice(animals)
guesses = 'aeiou'
turns = 5

while turns > 0:
    missed = 0
    for letter in secret:
        if letter in guesses:
            print letter,
        else:
            print '_',
            missed += 1

    print

    if missed == 0:
        print 'You win!'
        break

    guess = raw_input('guess a letter: ')
    guesses += guess

    if guess not in secret:
        turns -= 1
        print 'Nope.'
        print turns, 'more turns'
        if turns < 5: print ' O '
        if turns < 4: print '\_|_/'
        if turns < 3: print ' | '
        if turns < 2: print ' / \ '
        if turns < 1: print ' d b '
    if turns == 0:
        print 'The answer is', secret
```

BONUS

Internet Access:

A Python program can be used to access the internet and search a website for data or a sentence. In the example below you can search for the word 'school' in your school website.

The first part of the program is to import the URL library 2 module with the command **import urllib2**. The next step is to search for the data,

```
for line in urllib2.urlopen("http://fullwebaddress of site"):
    if "word you are searching for" in line:
        print line
```

What does the program do?