**Python variables and functions:**

Python as a language requires less syntax to perform a lot of operations than C. For example, each line in python does not need to end with a semicolon and variable types don’t always need to be expressly declared such as if you assign a string piece of data to a new variable the variable will automatically declare itself as a string.

**Assignment** is done using an = for example:

player\_choice = “rock”

The convention for variable names is to use snake\_case

To write a String, you can use either single or double quotes such as ‘String’ and “String”.

In python, indentation is very important. When defining a function, any code that is indented more than the declaration line is considered inside of the function which is used instead of the { and } used in C to show what code is inside of the function.

**Functions** and **procedures** are declared using the keyword def for example:

def get\_choices():

//code inside the function

//code outside the function

Where parameters can be defined inside the () and the : is used to show that following indented code will belong to this function (and the colon can also be used in things like if statements to show the following indented code will be part of this if statement).

The return keyword is used to return a value from a function for example:

def function():

value = 4

return value

When a return statement is reached, no further lines will be run inside of the function.

To call a subroutine, you write the name of the subroutine with brackets containing any arguments for example:

def the\_function(value):

value = value + 1

return value

the\_function(1)

The final line is where the function is called which will return the number 2.

The print function is used to display text to the console for example

print(“words”)

def word\_function():

return “words”

print(word\_function)

Both print statements will print the word “words” to the console.

**Dictionaries, inputs and random:**

**Dictionaries** store data in key value pairs. This is where keys can be listed alongside values which they are set equal to for example:

dict = {“name”: “beau”, “colour”: “blue”, “age”: 17}

The value in the pair can refer to a variable or other data structure and data can be retrieved from a dictionary such as in the example:

dict[“name”]

this will return the string “beau” as referring to the key returns the value it is tied to.

The **input** function is used to get a string input from the user which they type for example:

player\_input = input(“Please input something here”)

where the parameter of the function is printed onto the screen.

Python has a large quantity of libraries which can be linked using the **import** keyword for example:

import random

which imports the “random” library. The random library can be used to generate random numbers or random indexes.

**Lists and selection:**

A **list** is used to store multiple items of data of the same data type for example

the\_list = [“name1”, “name2”, “name3”]

To generate a random item from a list you would do:

random\_choice = random.choice(the\_list)

**If statements** are a form of selection which checks a given condition, and if the condition is true then it will run a segment of code and if it is false then it will not run the segment. Else statements can also be used to run a section of code if previous if statements are false. Elif statements are used to combine else if together which will run the segment of code if the previous if statements are false and if the new given statement is met. For example:

number1 = 4

number2 = 5

if number1 > number2:

print(number1 + “ > ” + number2)

elif number1 < number2:

print(number1 + “ < ” + number2)

else:

print(number1 + “ = ” + number2)

which will print “4 < 5” as 5 is greater than 4 which is the code inside of the elif.

The **comparison operators** that can be used in python are:

== which is true if both items are the same.

!= which is true if both items are NOT the same.

> which is true if the left item is greater than the right item.

< which is true if the right item is greater than the left item.

>= which is true if the left item is greater than or equal to the right item.

<= which is true if the right item is greater than or equal to the left item.

The **Boolean operators** that can be used in python are:

**and** which is true if both statements are true.

**or** which is true if at least one of the statements are true.

**not** which is true if the single statement is false (and false if the single statement is true).

It is possible to use **nested if statements** to run a section of code only if a condition is met and then if a second condition is also met. For example:

if player == “rock”:

if computer == “paper”:

print(“You lose”)

elif computer == “scissors”:

print(“You win!”)

This means that this entire section of code will only run if the player variable is equal to rock and inside of that, You lose will be printed if computer equals paper whereas you win will be printed if computer equals scissors.

**String handling:**

The + sign can be used to concatenate two strings together which means to combine them into a single string. For example:

new\_string = “This string” + “ That string”

This will make new\_string == “This string That string”

f-strings in python combines text and variables in a single string. For example:

age = 25

print(f“Jim is {age} years old.”)

Where any variables are contained in the {} and there must be a f at the start of the f-strings.