The University Of Sheffield.

The University of Sheffield CiCS Research IT

INTRODUCTION TO PROGRAMMING USING PYTHON

SETUP THE PYTHON ENVIRONMENT

If the Anaconda Python is installed on your desktop.
Please start Spyder.

If it is not installed, please go to the Software Centre and install it.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

SESSION 1

The University Of Sheffield.

The University of Sheffield CiCS Research IT

INTRODUCTION

The University Of Sheffield.

The University of Sheffield CiCS Research IT

WHAT IS PYTHON?

- We learn Python 3 !
- Well-designed language with a reasonably clean syntax.
- Supporting multiple platform: Linux, macOS, Win, etc…
- Python is so popular! Excellent included and third party documentation.
- Python is a high-level programming language for general-purpose programming.

HIGH VS LOW - LEVEL PROGRAMMING LANGUAGE

High-level programming	Low-level programming
Python, Matlab, IDL, etc…	C/C++, Java, PHP, etc…
Portable applications, platform dependent	Non - Portable applications, platform independent
Easy to read, understand and write the code	Hard to learn and understand the code
Not fast, Python have a lot of abstractions and layers before it reaches the hardware.	In low-level the machine is nearer, fast to process and return the output

The University Of Sheffield.

The University of Sheffield CiCS Research IT

THE HARD WAY IS EASIER.

You will do the incredibly simple things that all programmers do to learn a programming language:

1. Go through each exercise.

2. Type in each sample exactly. Do not copy-paste!

3. Make it run.

We start to write programs now. This will be very difficult at first but it teaches reading and writing, attention to detail, and spotting differences.

Finally, the purpose of this setup is so you can do four things very reliably while you work on the exercises:

1. Write exercises using your text editor
   We use Spyder but you can use something else.

2. Run the exercises you wrote.
   Again, we use Spyder for running but a simple terminal is good as well.

3. Fix them when they are broken.
   Try to solve the issue. The complier tells you what is the problem. Google the error. Trust me, somebody else had the same issue. The solution is there.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

HELLO, WORLD!

Write your first code and run it! Use the apostrophe.

Save your code and start a new file.

```
print('Hello World!')
```

Study Drill

Make your script print another line: “How are you?”
And one more: “Meh.”

The University Of Sheffield.

The University of Sheffield CiCS Research IT

SOLUTION

```
print('Hello World!')
print('How are you?')
print('Meh.')
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

COMMENTS

Really important. It tells you what something does in English! Good program = 'n' line code + 'n' line comment

```
#This line is your fist working code.
print('Hello World!')
```

Study Drill

Use the previous code and comment every line.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

SOLUTION

```
#This line is your fist working code.
print (‘Hello World!’)

#This line is really polite.
print ('How are you?’)

#This line is not.
print ('Meh.’)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

MATH - CHEATSHEET

Let's name the symbols:

+ plus

- minus

/ slash

* asterisk

% percent

< less-than

> greater-than

<= less-than-equal

>= greater-than-equal

```
#Plus
print(4 + 5)

#Divide
print(5 / 4)

#Mix
print((5 - 7) * 3)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

VARIABLES

Python variables do not need explicit declaration to reserve memory space.The declaration happens automatically when you assign a value to a variable.The equal sign (=) is used to assign values to variables.

```
#My variables
counter = 100                # An integer assignment
pi      = 3.14               # A floating point
name    = 'John'             # A string

#Print them
print(counter)
print(pi)
print(name)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

Study Drill

Write a simple calculator.
Define two variables ‘a = 12’ and ‘b = 3’.
Print the basic arithmetics (+, -, *, /).
The output should be 15, 9, 36 and 4.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

SOLUTION

```
#Define the variables
a = 12
b = 3

#Print the basic arithmetics.
print(a + b)
print(a - b)
print(a * b)
print(a / b)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

PROMPTING PEOPLE

If you want to read the value of a new variable from the keyboard you can use the input() function.

```
#Variables from keyboard
x = input('Type something here: ')

#Print it
print(x)
```

Warning! ‘x’ will be a string! But you can convert it.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

TYPE CONVERSION

Sometimes it's necessary to perform conversions between the built-in types. To convert between types you simply use the type name as a function.

```
#Integer to float
counter = 100           #An integer assignment
string = '53'           #String can be anything, even a “number”

#conversion int to float
counter = float(counter)

#conversion string to int
number = int(string)

#Print them
print(counter)
print(number)
```

TYPE CONVERSION

- If the string is a number you can convert it to an integer or float. If the string is not a number the conversion will fail.
- You can always convert integer to float.
- You can convert float and integer to a string.

We learned two data types: a **string** and **number**.
Number can be **integer** or a **float**.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

Study Drill

Modify your calculator. Let the user define the input numbers and print the basic arithmetics.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

SOLUTION

```
#Define the variables
a = input('First? ')
b = input('Next? ')

#Type conversion.
a = float(a)
b = float(b)

#Print the basic arithmetics.
print(a + b)
print(a - b)
print(a * b)
print(a / b)
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

PRACTICE SESSION 1

The University Of Sheffield.

The University of Sheffield CiCS Research IT

LOOPS AND IF STATEMENT

IF STATEMENT (IF)

The general syntax for a simple if statement is:

if condition ***then***

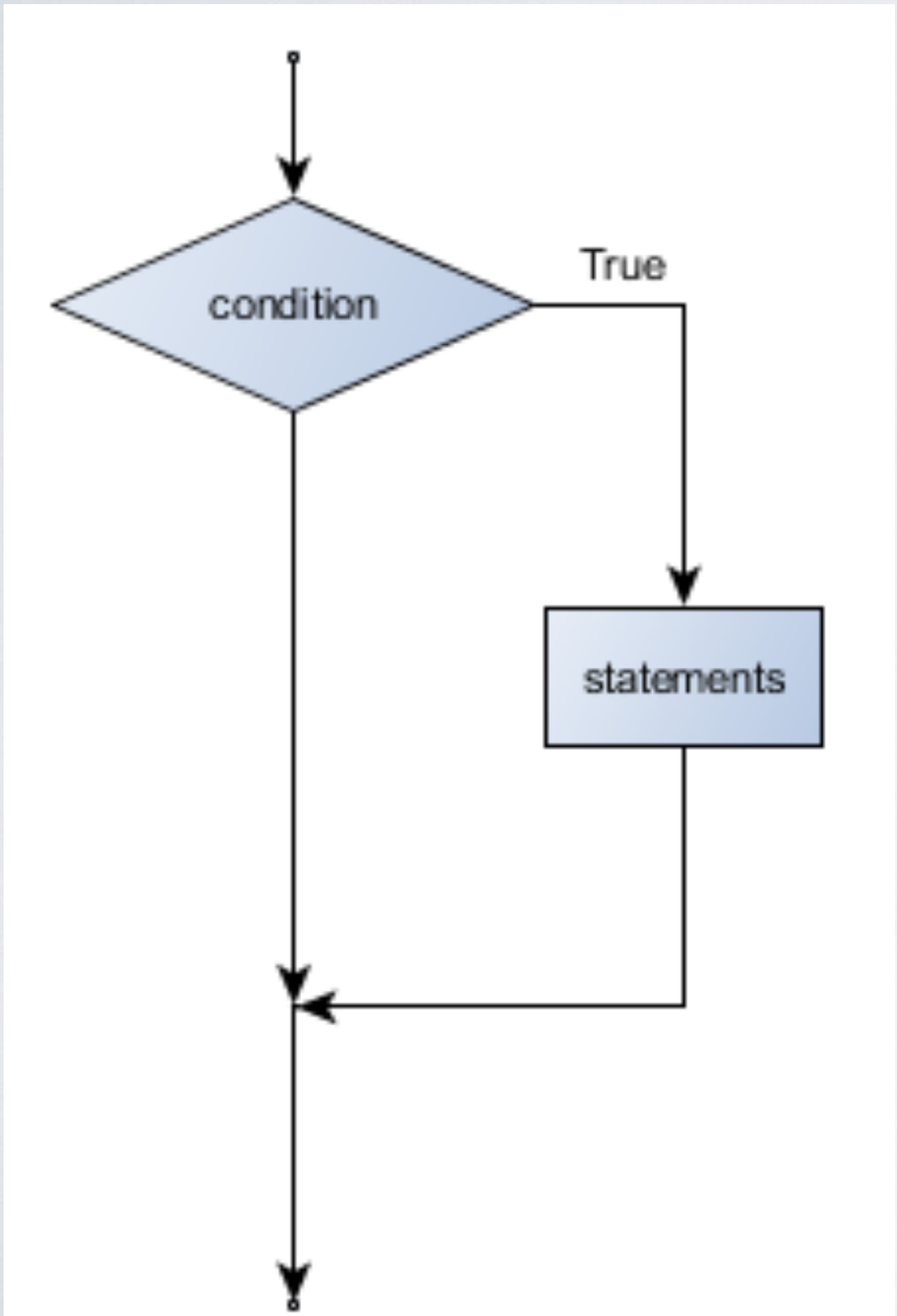
indentedStatementBlock

If the condition is true, then do the indented statements. If the condition is not true, then skip the indented statements.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

IF STATEMENT




```
#Define a number
number = 4

#Mind the indentation
if number > 0:
    print('Positive')
```

IF STATEMENT (IF - ELSE)

if condition ***then***

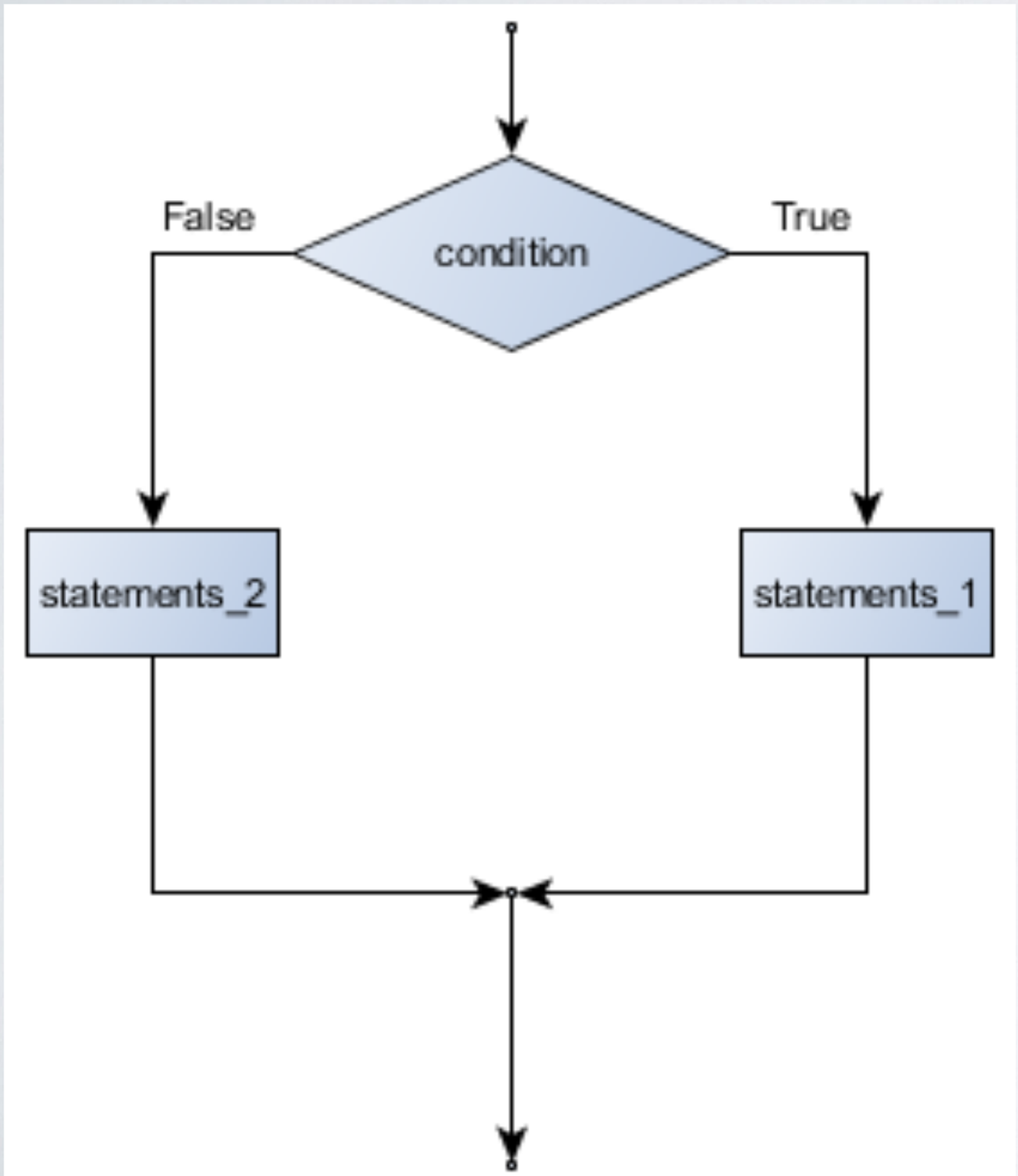
    indentedStatementBlock

else

    indentedStatementBlock

If the condition is true, then do the indented statements. If the condition is not true, then do the other indented statements.

IF STATEMENT (IF - ELSE)

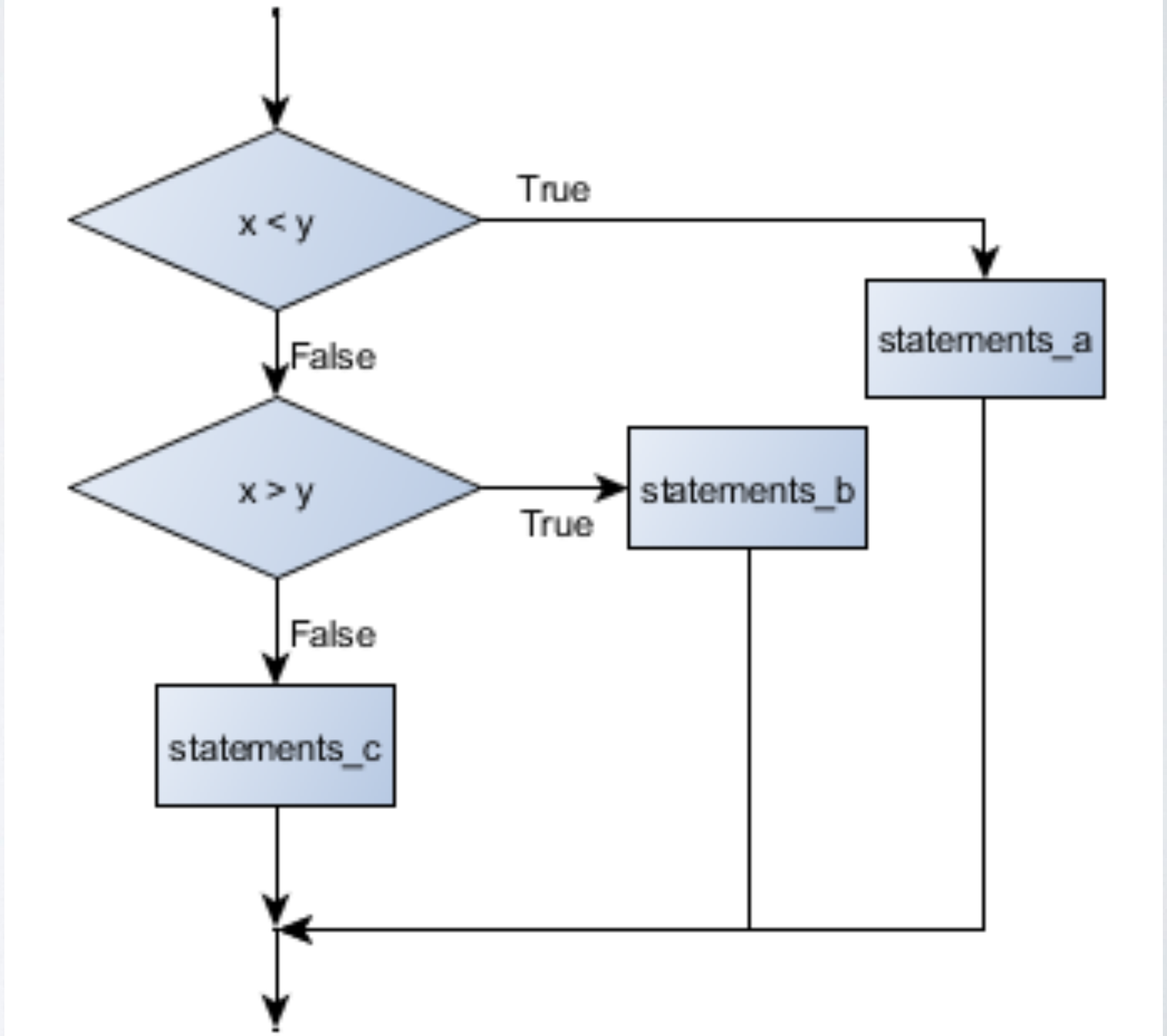



```
#Define a number
number = 4

#Mind the indentation
if number > 0:
    print('Positive')
else:
    print('Negative')
```

MORE CONDITION? (ELIF)

The **elif** statement allows you to check multiple expressions for TRUE and execute a block of code as soon as one of the conditions evaluates to TRUE. Again, indentation is really important!



The University Of Sheffield.

The University of Sheffield CiCS Research IT

MORE CONDITION? (ELIF)

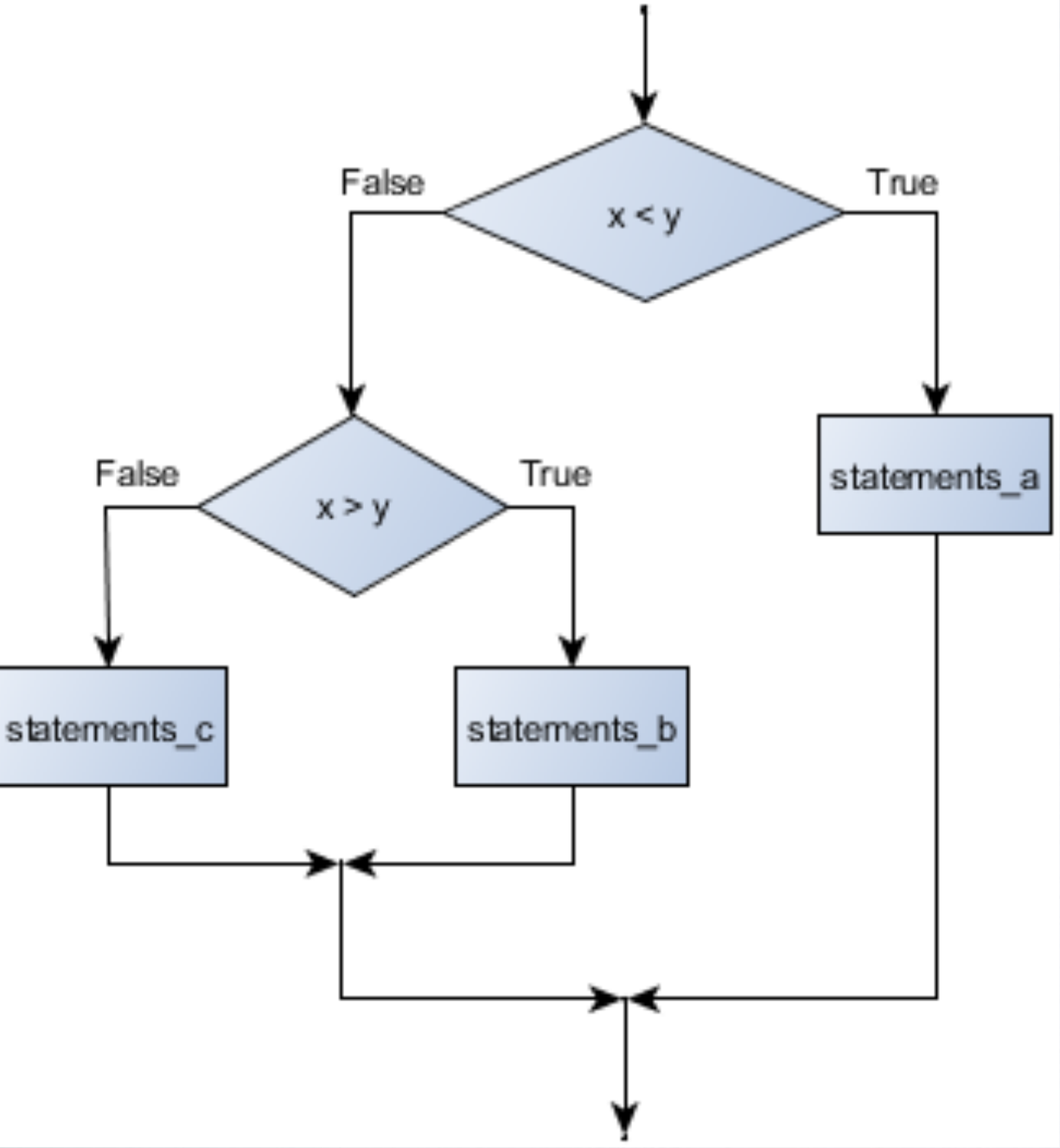
```
#Define a number
number = 4

#Mind the indentation
if number > 0:
    print('Positive')
elif number == 0:
    print('The number is zero.')
else:
    print('Negative')
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

NESTED CONDITION




```
x = 3
y = 7
if x < y:
   print('y is larger')
else:
   print('x is larger')
   if x > 0:
      print('x is pos.')
   else:
      print('x is neg.')
```

MULTIPLE CONDITION

```
a = 3
b = 5
c = 2

if a > 0 and b > 0 and c > 0:
    print('All positive!')
else:
    print('Negative number(s)')
```

Study Drill

The program reads three inputs (angles). Convert to variables to float! Your program has to check wether the tree given angles are able to create a triangle or not. Three given angles form a non-degenerate triangle (and indeed an infinitude of them) if and only if both of these conditions hold: (a) each of the angles is positive, and (b) the angles sum to 180°. If degenerate triangles are permitted, angles of 0° are permitted.

The University Of Sheffield.

The University of Sheffield CiCS Research IT

SOLUTION

```
#Define the variables
a = input(“1st angle? “)
b = input(“2nd angle? “)
c = input(“3rd angle? “)

#Convert the variables
a = float(a)
b = float(b)
c = float(c)

#Print the basic arithmetics.
if a > 0 and b > 0 and c > 0:
   if a + b + c == 180:
      print('Possible triangle')
   else:
      print('The sum of the angles is not equals to 180.')
else:
   print('One of the angle (or more then one) is zero.')
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

FOR LOOP

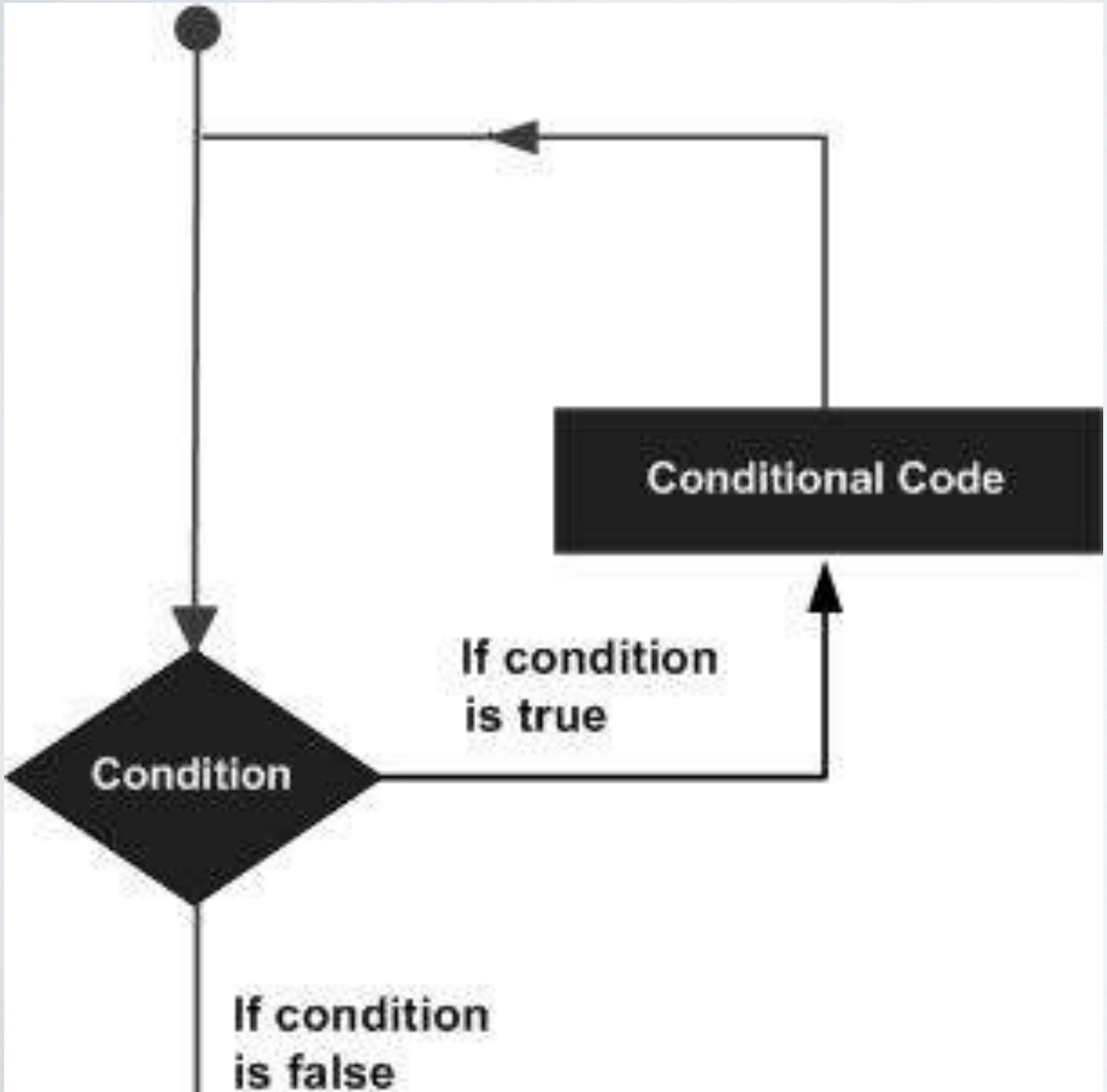
In general, statements are executed sequentially: The first statement in a function is executed first, followed by the second, and so on. There may be a situation when you need to execute a block of code several number of times.

```
for LOOP_VARIABLE in SEQUENCE:
    STATEMENTS
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

FOR LOOP




```
for i in range(5):
    print(i)
```

```
Output

0
1
2
3
4
```

Ok, but what is range()?

THE RANGE FUNCTION

The range() function is used to indicate how many times the loop will be repeated.

The structure of the function is range(start, upto, step) in which the arguments of range are used as follows:

- start and step are both optional.
- upto must always be there
- start, upto, and step must all be integers

The University Of Sheffield.

The University of Sheffield CiCS Research IT

THE RANGE FUNCTION

range(10) produces: 0,1,2,3,4,5,6,7,8,9

range(1, 7) produces: 1,2,3,4,5,6

range(0, 30, 5) produces: 0,5,10,15,20,25

Study Drill

1. Write a program that will ask the user for a message (string) and the number of times they want that message displayed. Then output the message that number of times.

2. Write a program that will calculate the average (mean) of a set of numbers. This time, the user is to be asked how many numbers are to be averaged, they must then enter this number of numbers. Your program will calculate and display the average of those numbers.

   Hint #1: Use input() function in the for loop.
   Hint #2: You can summarise the numbers in the for loop as well:
   for example: sum = sum + number

SOLUTION

```
#Define the variables
msg = input('Please write the message here: ')
n = input('How many times would you like to repeat?: ')

#Conversion
n = int(n)

#For loop for printing multiple lines
for i in range(n):
   print(msg)
```

SOLUTION

```
#Define the variables
sum = 0     #For storing the summarised numbers
n = input('How many number would you like to enter? ')

#Conversion
n = int(n)

#For loop
for i in range(n):
    number = input('Enter a number: ')
    number = float(number)
    sum = sum + number

#Calculate the average
print(sum / float(n))   #inline conversion
```

The University Of Sheffield.

The University of Sheffield CiCS Research IT

PRACTISE SESSION 2