COMPUTE SCIENCE

EDEXCEL INTERNATIONAL GCSE
O LEVEL

PYTHON ACTIVITIES

CHAPTER 5 / CHAPTER 6 /
CHAPTER 7/ CHAPTER 9 /
CHAPTER 10 / CHAPTER 11

Dr. Saw Myat Sandar

STUDENT WORK BOOK

Chapter 5
Activity 1 Data Type

1 Investigate what data types are available in the high-level language you are studying.

High-level programming languages often provide more than the four basic data types.

Data Type	Description	Example	Example of Use
Integer	Use to store whole number	30	Age = 30
Real or Float	User to store numbers with a decimal place	25.5	Weight = 25.5
Boolean	Two possible values: True or False	False	Correct=False
String	A set of characters which can include spaces and numbers	'John'	Firstname='john'
Character	A single letter	'f'	Gender='f'

*** Python does not have a character type.

DATA TYPE	DESCRIPTION	EXAMPLE	EXAMPLES OF USE
integer	Used to store whole numbers without a fractional part	30	age = 30 number = 5
real or float	Used to store numbers with a fractional part (decimal place). Real numbers are sometimes referred to as floats (short for floating point)	25.5	weight = 25.5 price = 12.55
Boolean	Only has two possible values: True or False	False	correct = False lightOn = True
character*	A character can be a single letter, a symbol, a number or even a space. It is one of the four basic data types	'm'	gender = 'm' char = ':'
string	A set of characters which can include spaces and numbers and are treated as text rather than numbers	'the computer'	name = 'Catherine' type = 'liquid'

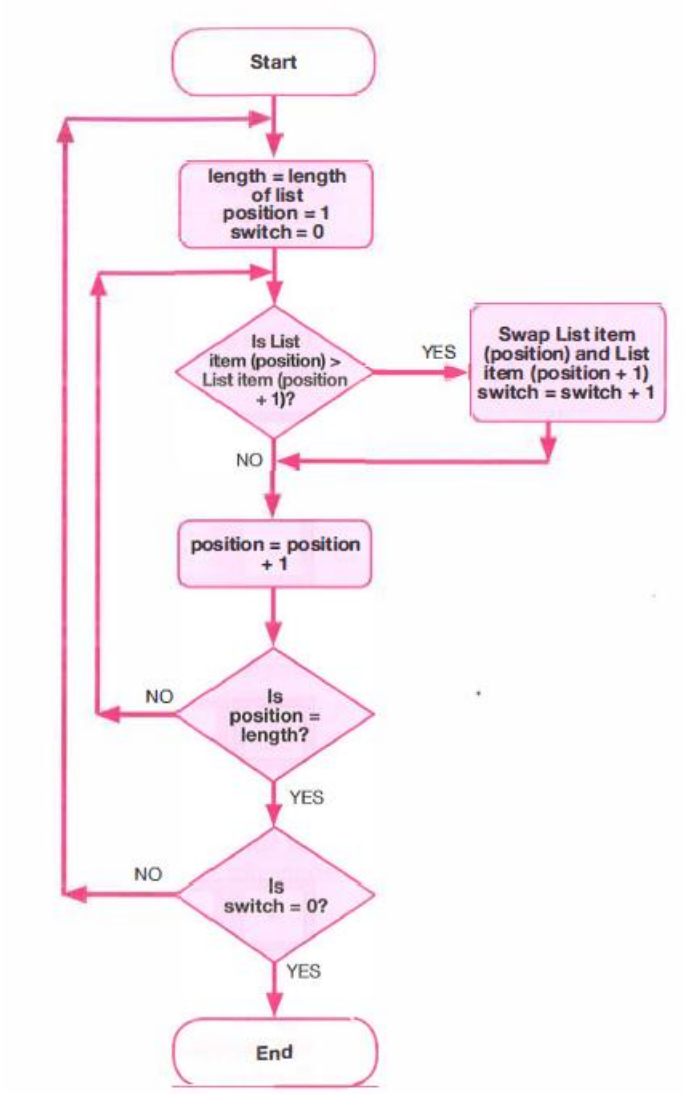
*Python does not have a character data type.

▲ Table 2.1 Common data types

2 What do you think is an appropriate data type for each of these items?
a the test score of an individual learner
b the average score for a group of learners
c whether or not the pass mark for the test has been achieved.

a Integer
b Real
c Boolean

3 Look back over the algorithms you wrote in Unit 1 and find instances of variable initialisation.



Variable initialization – position=1, switch=0

Activity 2 Monitoring Visitor Numbers

A theme park uses a program to monitor the number of people entering and exiting the park.
The maximum number of visitors at any one time must not exceed 10 000.
When the number of people in the park reaches the maximum, a 'Park Full' message is displayed at the entrance gate.
Children can visit the park free of charge.
Adults must pay AED 125 admission.
The program records the amount of money collected at the gate.

1 What are the variables needed in the program?
2 Select an appropriate data type for each variable and constant.

VARIABLE	PURPOSE	DATA TYPE
totalVisitors	Keeps track of the number of people in the park at any one time. (It is incremented by one each time someone enters and decremented by 1 each time someone leaves.)	integer
visitorType	Set to 'a' if the visitor is an adult and 'c' if the visitor is a child.	character
Taking	ning total of the amount of money collected at the gate r adult; no charge for children).	real
parkFull	Set to 'False' initially but changes to 'True' when the number of visitors reaches 10,000.	Boolean

string

amount

*** *while parkfull==False:*

```
# Activity2
max_capacity = 10000
admission_fee = 125
visitors_count = 0
money_collected = 0

# Simulate visitors entering the park
while visitors_count < max_capacity:
    visitor_type = input("Enter 'A' for adult, 'C' for child, or
'Q' to quit: ").upper()

    if visitor_type == 'Q':
        break

    if visitor_type == 'A':
        visitors_count += 1
        money_collected += admission_fee
        print("Adult entered the park.")
    elif visitor_type == 'C':
        visitors_count += 1
        print("Child entered the park (free).")
    else:
        print("Invalid input. Please enter 'A', 'C', or 'Q'.")

    if visitors_count >= max_capacity:
        print("Park Full! Cannot admit more visitors at the
moment.")
        break

# Display park statistics
print(f"Current Visitors Count: {visitors_count}")
print(f"Money Collected: AED {money_collected}")
```

Activity 3 Understanding Algorithms

Read the following algorithm written in pseudocode and then answer the questions below.

```
RECEIVE number1 FROM (INTEGER) KEYBOARD
RECEIVE number2 FROM (INTEGER) KEYBOARD
SET resultl TO numberl / number2
SEND resultl TO DISPLAY
SET result2 TO numberl MOD number2
SEND result2 TO DISPLAY
SET result3 TO numberl DIV number2
SEND result3 TO DISPLAY
```

1 What does this algorithm do?
2 What is the output of the algorithm, given the following inputs:
a 4, 2
b 10,3
C 20, 6?
Implement this algorithm in the high-level language you are studying.

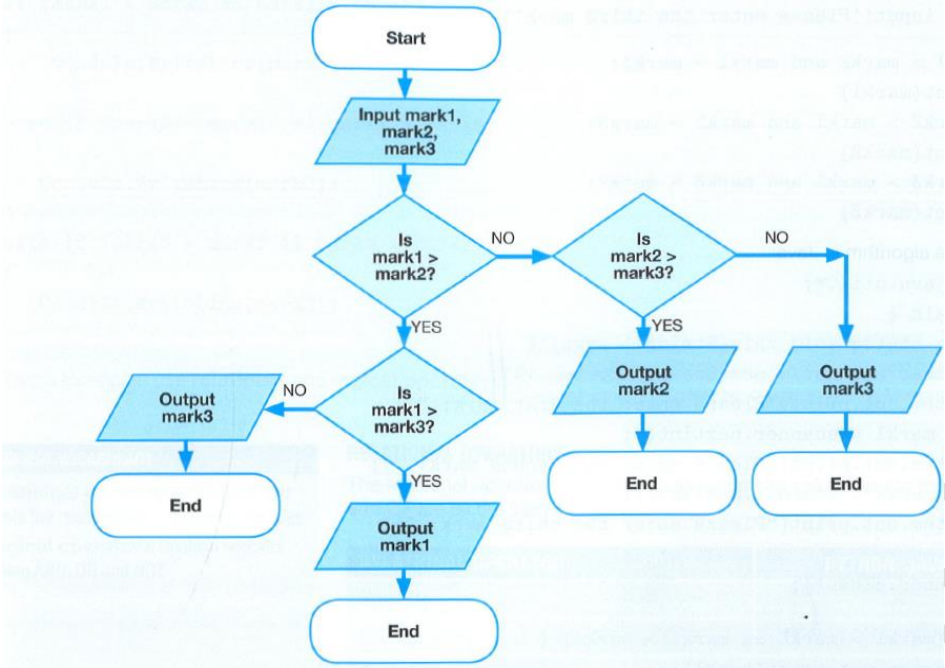
1 The algorithm takes in two integer numbers entered from the keyboard and displays:

- the result of dividing the first number by the second number
- the number remaining after dividing the first number by the second number
- the integer division.

2
a With 4 and 2 as inputs, the outputs would be: 2.0, 0, 2
b With 10 and 3 as inputs, the outputs would be: 3.3333, 1, 3
c With 20 and 6 as inputs, the outputs would be: 3.3333, 2, 3

```python
#activity3
number1=int(input("Enter number 1:"))
number2=int(input("Enter number 2:"))

result1=number1/number2
print(result1)

result2=number1%number2
print(result2)

result3=number1//number2
print(result3)
```

Worked Example



▲ Figure 2.1 Flowchart of an algorithm to print out the highest homework mark

```
RECEIVE mark1 FROM KEYBOARD
RECEIVE mark2 FROM KEYBOARD
RECEIVE mark3 FROM KEYBOARD
IF mark1 > mark2 AND mark1 > mark3 THEN
    SEND mark1 TO DISPLAY
ELSE
    IF mark2 > mark1 AND mark2 > mark3 THEN
        SEND mark2 TO DISPLAY
    ELSE
        IF mark3 > mark1 AND mark3 > mark2 THEN
            Send mark3 TO DISPLAY
        END IF
    END IF
END IF
```

```python
#ch5_workedexample
mark1=int(input("Enter mark 1:"))
mark2=int(input("Enter mark 2:"))
mark3=int(input("Enter mark 3:"))
if mark1>mark2 and mark1>mark3:
    print(mark1)
elif mark2>mark1 and mark2>mark3:
    print(mark2)
elif mark3>mark1 and mark3>mark2:
    print(mark3)
```

Activity 4

Look at the following algorithm and answer the questions.

```
IF score <= highScore THEN
    SEND 'You haven't beaten your high score.' TO DISPLAY
ELSE
    SEND 'You've exceeded your high score!' TO DISPLAY
END IF
```

What is the output of the algorithm when

- score = 5 and highScore = 10?
- score = 20 and highScore = 10?
- score = 15 and highScore = 15?

ACTIVITY 4

SCORE	HIGHSCORE	OUTPUT
5	10	You haven't beaten your high score.
20	10	You've exceeded your high score!
15	15	You haven't beaten your high score.

```python
anotherGo="y"
while anotherGo=="Y" or anotherGo=="y":
    score=int(input("Enter scored:"))
    highscore=int(input("Enter high scored:"))
    if score<=highscore:
        print("You have not beaten your high score")
    else:
        print("You have exceeded your high score")
    anotherGo=input("Do you want another go (y or n)")
```

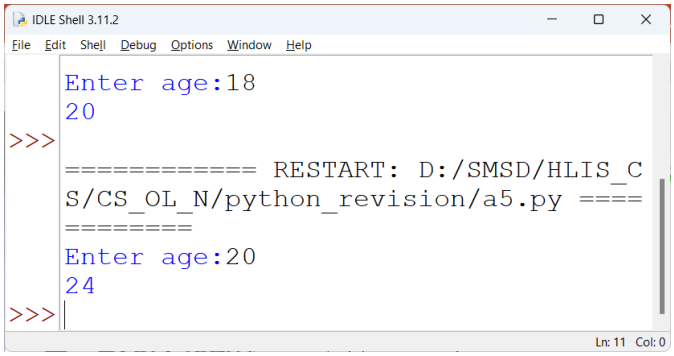
Activity 5

A driving school uses this rule to estimate how many lessons a learner will require.

- Every learner requires at least 20 lessons.
- Learners over the age of 18 require more lessons (two additional lessons for each year over 18).

Create a program in a high-level language that inputs a learner's age and calculates the number of driving lessons they will need.

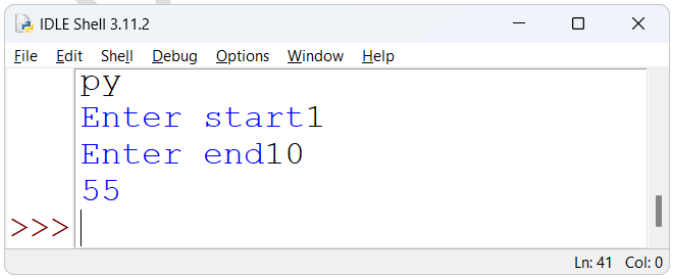
```
#activity5
age=int(input("Enter your age:"))
if age<=18:
    numberLesson=20
else:
    numberLesson=20+(age-18)*2
print("You need",numberLesson)
```



Activity 6

Produce a program in a high-level language that asks a user to enter a start number and an end number and then outputs the total of all the numbers in the range. For example, if the start number was 1 and the end number was 10, the total would be 55 (10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1).

```
#activity6
start=int(input("Enter start number:"))
end=int(input("Enter end number:"))
total=0
for count in range(start,end+1):
    total=total+count
print(total)
```



Activity 7

1 Python

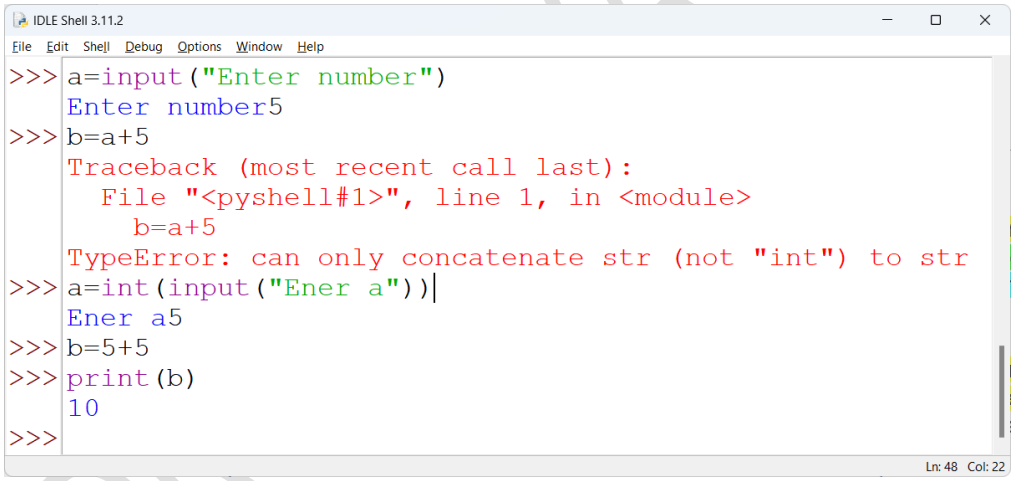
Look at the following program and then answer the questions below.

```
for student in range(1, 21):
    sum = 0
    for mark in range(1, 6):
        nextMark = int(input('Please enter a mark'))
        sum = sum + nextMark
    averageMark = sum/5
    print(averageMark)
```

a What is the purpose of this program?

b Why is `int` used in the line `nextMark = int(input('Please enter a mark'))`?

a It uses nested loops to calculate the average mark for 5 pieces of work from each of twenty students.
b To ensure that the marks entered by the user are recognised as integers and not strings.



Activity 8

Produce an algorithm that will print out the times tables (up to 12 times) for the numbers 2 to 12.

```
#activity8
for table in range(2,13):
    for times in range(2,13):
        print(str(table)+"x"+str(times)+"="+str(table*times))
```

```
IDLE Shell 3.11.2
File Edit Shell Debug Options Window Help
N/python_revision/a8.py
2x2=4
2x3=6
2x4=8
2x5=10
2x6=12
2x7=14
2x8=16
Ln: 299 Col: 0
```

Activity 9

What do these two algorithms do? Implement them in the high-level programming language you are studying.

a Algorithm A

```
FOR index FROM 1 TO 10 DO
    SEND index * index * index TO DISPLAY
END FOR
```

b Algorithm B

```
SET counter TO 10
WHILE counter > 0 DO
    SEND counter TO DISPLAY
    SET counter TO counter - 1
END WHILE
```

1
a Algorithm A displays the numbers 1 to 10 cubed, i.e. 1–1000.
b Algorithm B displays a countdown from 10 to 1.

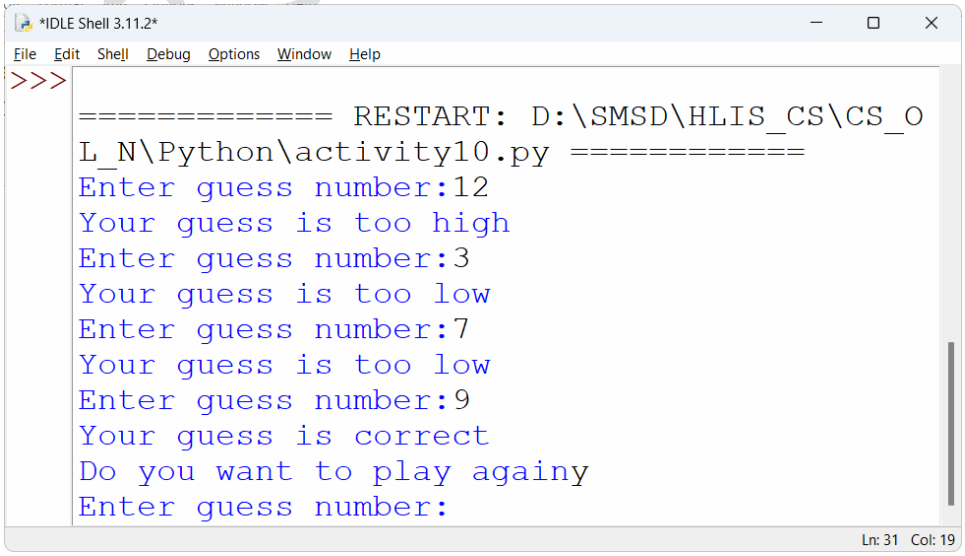
```python
#activity9
#Algorithm A
for index in range(1,11):
    print(index*index*index)

#Algorithm B
counter=10
while counter>0:
    print(counter)
    counter=counter-1
```

Activity 10

Create a guessing-game program with the following specification.

- The computer generates a random number between 1 and 20.
- The user is asked to enter a number until they enter this random number.
- If their guess is too low or too high they are told.
- They are told when their guess is correct.
- They are asked if they want to play another game until their answer is 'NO'.

```
#activity10
import random
play=True
correct=False
while play==True:
    randomNumber=random.randint(1,20)
    while correct==False:
        guess=int(input("Enter a number between 1 and 20"))
        if guess==randomNumber:
            print("Your guess is correct.")
            correct=True
        elif guess<randomNumber:
            print("Your guess is too low")
        elif guess>randomNumber:
            print("Your guess is too high")
    reply=input("Do you want to play again? (y or n)")
    reply=reply.upper()
    if reply=='Y':
        correct=False
        play=True
    else:
        play=False
```



Chapter5 Checkpoint

S1 Why are variables needed?
Variable is used to store value.

S2 Provide examples of the four data types.
S2 Ideally, students should give examples of different data types from their own programs.
Data Type
1. Integer – eg.age = 10
2. Real/Float – eg. temperature=98.7
3. String – eg. first_name="John"
4. Boolean – eg. found=False

S3 How are selection and iteration implemented in the high-level language you are studying?
S3
Selection
if Statements: These are used for simple conditional branching.
Example

```
if score>=40:
     print("Pass")
else:
     print("Fail")
```

Iteration: Looping
Two types of looping.
for Loops: Used for iterating over a sequence.
fruits=["apple", "banana", "cherry"]
Example

```
for x in fruits:
     print(x)
```

while loop: Used for repeating a block of code while a certain condition is true.
Example

```
password=input("Enter password")
while password=='123':
    print("Valid")
    break
else:
    print("Invalid")
```

C1 Outline the following structural components of a program: variable and type declarations, command sequences, selection and iteration constructs.
C1 Command sequences, selection and iteration were described in detail in Unit 1.
They are revisited in this unit on pages 38–49.
Variable initialisation is covered on pages 35–36.
As well as being able to describe these constructs, students should also be able to recognise them in an algorithm.

Chapter 6
Activity 11 Rewriting and Implementing Algorithms

1 Program the following algorithm in a high-level language and make it readable by adding comments and indentation.

```
SET x TO 10
WHILE x >= 0 DO
  IF x > 0
    SEND x TO DISPLAY
  ELSE
    SEND 'Blast Off' TO DISPLAY
    END IF
  SET x TO x -1
END WHILE
```

2 Develop an algorithm for a simple calculator and code it in a high-level language that:

a allows the user to choose from these options: addition, subtraction, division and multiplication
b prompts the user to input two numbers
c performs the calculation and displays the result
d offers the user the option of performing another calculation.

```
SET countDown TO 10
WHILE countDown >=0 DO
      IF countDown >0 THEN
            SEND countDown TO DISPLAY
      ELSE
            SEND 'Blast Off' TO DISPLAY
      END IF
      SET countDown TO countDown-1
END WHILE
```

```
countDown=10
while countDown>=0:
    if countDown>0:
        print(countDown)
    else:
        print("Blast off")
    countDown-=1
```

2 Here is the calculator algorithm expressed as source code:

```
REPEAT
REPEAT
      SEND ‘Select an option: a - addition, s - subtraction, d -
division or m - multiplication’ TO DISPLAY
      RECEIVE choice FROM KEYBOARD
UNTIL choice = ‘a’ OR choice = ‘s’ OR choice = ‘d’ OR choice = ‘m’
SEND ‘Enter the first number’ TO DISPLAY
RECEIVE number1 FROM KEYBOARD
SEND ‘Enter the second number’ TO DISPLAY
RECEIVE number2 FROM KEYBOARD
IF choice = ‘a’ THEN
      SEND number1 + number2 TO DISPLAY
ELSE
IF choice = ‘s’ THEN
      SEND number1 - number2 TO DISPLAY
ELSE
IF choice = ‘d’ THEN
      SEND number1 / number2 TO DISPLAY
ELSE
      SEND number1 * number2 TO DISPLAY
END IF
      END IF
END IF
SEND ‘Another go?’ TO DISPLAY
RECEIVE anotherGo FROM KEYBOARD
UNTIL anotherGO <> ‘y’ OR anotherGo <> ‘Y’
```

```python
#chapter6_activity11_2
anotherGo="y"
while anotherGo=="y" or anotherGo=="Y":
    while True:
        choice=input("Select an option:a-addition,s-subtraction,m-
multiplication,d-division:")
        if choice=='a' or choice=='s' or choice=='m' or
choice=='d':
            break
        else:
            print("Invalid")
    firstNumber=int(input("Enter first number:"))
    secondNumber=int(input("Enter second number:"))
    if choice=='a':
        print(firstNumber+secondNumber)
    elif choice=='s':
        print(firstNumber-secondNumber)
    elif choice=='d':
        print(firstNumber/secondNumber)
    elif choice=='m':
        print(firstNumber*secondNumber)
    anotherGo=input("Do you want another go (y or n)?")
```

```
*IDLE Shell 3.11.2*
File Edit Shell Debug Options Window Help
Enter a,s,d,m
= RESTART: D:/SMSD/HLIS_CS/CS_OL_N/python_revision/a11.py
Select an option:a-addition,s-subtraction,m-multiplication,d-division:a
Ener first number:10
Ener second number:0
10
Do you want another go?y
Select an option:a-addition,s-subtraction,m-multiplication,d-division:d
Ener first number:8
Ener second number:2
4.0
Do you want another go?
Ln: 44 Col: 23
```

Checkpoint

S1 Why it is important to make your code easy to read?

Programming is **not a solo activity**. Programmers usually **work in teams**, with **each programmer** developing a **different part of the program**. This only works if they all adopt a standard approach to writing readable code.

Code you write is **easy to read and understand**. We refer to this as 'readability'. This benefits you and anyone else who needs to understand how your programs work.

S2 Outline the four techniques that a programmer should use to make code easy to read.

TECHNIQUE	DESCRIPTION
Comments	Comments should be used to explain what each part of the program does.
Descriptive names	Using descriptive identifiers for variables, constants and subprograms helps to make their purpose clear.
Indentation	Indentation makes it easier to see where each block of code starts and finishes. Getting the indentation wrong in Python will result in the program not running or not producing the expected outcomes.
White space	Adding blank lines between different blocks of code makes them stand out.

▲ **Table 2.5** Techniques for program clarity

Chapter 7
Activity 12

CHECKING PASSWORD LENGTH

Create and write a program to check the length of a password. If the password entered is less than six characters, the program should output 'The password you have entered is not long enough'; otherwise it should output 'Length of password OK'.

```
SEND 'Enter your password.' TO DISPLAY
RECEIVE password FROM KEYBOARD
IF LENGTH(password) < 6 THEN
     SEND 'The password you have entered is not long enough.' TO
DISPLAY
ELSE
     SEND 'Length of password OK.' TO DISPLAY
END IF
```

```
#activity12
password = input("Enter your password.")
if len(password)< 6:
    print("The password you have entered is not long enough.")
else:
    print("Length of password OK.")
```

Activity 13

Write a program that will check if a make of car entered by the user is in the string **'The cars present included Ford, Mercedes, Toyota, BMW, Audi and Renault.'**
If the car entered by the user is present, then 'It is present' should be returned or 'It is not present', if not.
It should not matter which case the car name is entered by the user.

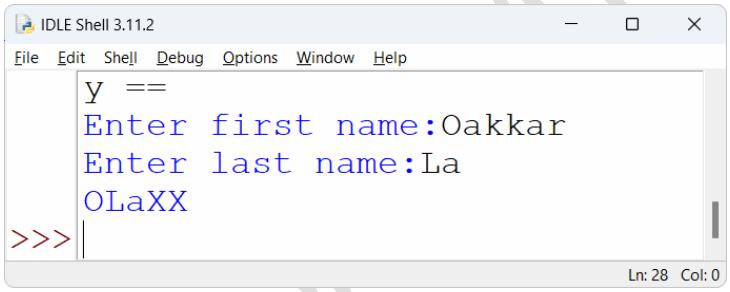
```
#activity13
string = "The cars present include Ford, Mercedes, Toyota, BMW,
Audi and Renault"
make = input("Please enter a make to search for.")
make = make.upper()
string = string.upper()
if make in string:
    print("It is present")
else:
    print("It is not present")
```

Activity 14 Concatenating and Slicing Strings

CONCATENATING AND SLICING STRINGS

A company wants a program to generate usernames for new employees. Each username consists of the first four letters of the employee's last name and the first letter of their first name joined together. If the employee's last name is less than four characters in length a letter 'X' is used to fill in for each of the missing characters. Develop a program that asks the user to input their first and last names and outputs their username.

```
#activity14
firstName=input("Enter first name:")
lastName=input("Enter last name:")
length=len(lastName)
if length==1:
    lastName=lastName+"XXX"
elif length==2:
    lastName=lastName+"XX"
elif length==3:
    lastName=lastName+"X"
#first4letter=lastName[0:4]
username=firstName[0]+lastName[0:4]
print(username)
```



Checkpoint

S1 How are individual characters in a string referenced?

STRING INDEXING

GENERAL VOCABULARY

reference to refer to something

Each character in a string has an index number, with the first character at position 0. You can use the index to **reference** individual characters in a string.

Therefore the index position of the letter 'm' is 2, even though it is the third character, as shown in Table 2.6.

INDEX	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
String	C	o	m	p	u	t	e	r		S	c	i	e	n	c	e

▲ Table 2.6 An example string index

S2 Develop an algorithm that uses a loop to traverse a string.

S2 The program students developed in Activity 13 uses a loop to traverse a string.

S3 How are a string and a non-string concatenated?
S3 Concatenation and slicing strings are explained on pages 59–60.
A string and a non-string can be concatenated by using str().

Example

```
name='John'
age=20
print(name+str(age))
```

Concatenation involves joining two or more items of information together. Concatenating two strings produces a new string object. It is very useful when displaying text on screen.

In Pearson Edexcel pseudocode concatenation is done in the following way.

```
RECEIVE userName FROM (STRING) KEYBOARD
SEND 'Hello' & userName TO DISPLAY
```

Note that literal text is enclosed in speech marks but the variable name is not. In Python, this would be:

```
userName = input('Please enter your username')
print('Hello' + userName)
```

The '+' character is used for concatenation.

C1 Split Sentence

C1 Develop a program that asks the user to input a sentence and then splits it up wherever a space occurs.
Each word should then be displayed on a separate line.

```
#chapter7_c1
sentence=input("Enter a sentence:")
separateSentence=sentence.split(" ")
for index in separateSentence:
    print(index)
```