Teaching Input and Output in Python

Introduction – why Input/Output matters

In the A/L ICT syllabus (competency 9.7), learners are expected to write programs that take **input from the keyboard** and produce **output to standard devices** 【721046920033966†L2885-L2929】. Interacting with the user through input and output makes programs meaningful and prepares students for solving real-world problems. In this lesson you will learn how to display information using Python's print() function, how to read data from the user with input(), and how to combine the two to build interactive programs. The approach emphasises clear explanations, live coding demonstrations and progressively harder exercises to develop problem-solving skills.

1. Displaying Output – using print()

Concepts

- **print() basics** – print() displays values to the console. It takes one or more objects separated by commas and automatically adds a space between them and a newline at the end 【696423437970415†L218-L233】. When you pass non-string objects (numbers, lists, dictionaries, functions) to print(), Python converts them to a string representation before displaying them 【696423437970415†L233-L255】.
- **Separators and terminators** – The keyword argument sep changes the string inserted between printed objects, while end changes the string appended at the end 【696423437970415†L529-L609】. For example, print("input", "output", sep="/") prints input/output 【696423437970415†L537-L547】. You can suppress the newline by setting end="" or customise it with any string 【696423437970415†L594-L607】.
- **Printing without newline** – Inside loops you can display values on the same line by using end=" " and printing a final newline when the loop finishes 【696423437970415†L630-L649】.
- **Redirecting output** – The file argument of print() (not examined often) sends output to a file object instead of the screen 【696423437970415†L529-L609】.
- **Formatted output (f-strings)** – Python's f-strings allow you to embed expressions inside string literals, controlling numeric precision and alignment. For example: print(f"Area = {area:.2f}") prints the area rounded to two decimals.

Coding examples

```
# Basic printing
print("Welcome to A/L ICT!")
print("Name:", "Dinuka", "Fernando")

# Custom separator and terminator
print("ICT", "Python", sep=" – ", end=" ***\n")  # prints: ICT – Python ***
print("First", "Second", "Third", sep=" | ")      # prints: First | Second | Third
```

```python
# Printing a list and a dictionary
numbers = [1, 2, 3]
print("Numbers:", numbers)  # Python converts the list to its string representation
student = {"name": "Saman", "grade": 13}
print("Student details:", student)

# Using f-strings to format output
length = 5
width = 3
area = length * width
print(f"Rectangle {length}×{width} has area {area}")
```

Model questions (easy → hard)

1. **Easy** – Write a Python statement that prints Hello ICT Students. (Expected output: Hello ICT Students)
2. **Medium** – Given the variables a = 7 and b = 3, write a single print() statement using an f-string to display 7 + 3 = 10.
3. **Hard** – Write Python code to display the following three values on one line separated by tabs (\t): the string Date, the integer 25, and the float 3.14159 rounded to two decimal places. Then ensure the cursor moves to a new line afterwards.

Past paper-style MCQ (output)

1. **Which call to print() will output exactly:** Hello:World!

   A. print("Hello", "World", sep="-")
   B. print("Hello", "World", sep=":", end="!")
   C. print("Hello:" + "World!")
   D. print("Hello", "World!", sep="::")

2. **What does the call print("input", "output", sep="...", end="!!!") produce?**

   A. inputoutput!!!
   B. input output!!!
   C. input...output!!!
   D. input...output!!!\n

3. **Which keyword argument of print() controls what is inserted between objects?**

   A. file
   B. sep
   C. end
   D. flush

Past paper-style structured questions (output)

1. *Structured (filling blanks)* – Consider the following code:

```
length = 8
width = 5
________ # (i) print the text "Area = " without newline
________ # (ii) print the product of length and width
```

   (i) and (ii) represent two separate print() calls. Fill in the blanks so that the program outputs Area = 40 on one line.

2. *Short program* – Write a Python program that prints the first and last characters of the string course = "Information" separated by a hyphen, using a single print() call.

3. *Output formatting* – Complete the code below so that it prints a dictionary's key–value pairs in the format key -> value (one pair per line). Use appropriate print() arguments.

```
person = {"first_name": "Chamari", "last_name": "Perera"}
for key, value in person.items():
    ________
```

Past paper-style essay questions (output)

1. *Essay* – Describe with examples how the print() function separates multiple objects and how the sep and end keyword arguments can be used to control the formatting of output【696423437970415†L529-L609】. Write a Python program that prints the numbers 1 to 10 on a single line separated by commas and ending with an exclamation mark.
2. *Essay* – Explain the importance of readable output in interactive programs. Write a program that displays a formatted summary of a student's name, school, and G.C.E. A/L index number using one print() call and an f-string.
3. *Essay* – In your own words discuss why Python automatically converts non-string objects (such as lists and dictionaries) to strings before printing them 【696423437970415†L233-L255】. Provide examples of printing a list and a dictionary, and comment on how the output differs.

2. Formatting Output

Formatting output improves readability and aligns text or numbers. Although the A/L ICT syllabus does not explicitly mention advanced formatting, exposing students to f-strings helps them produce neat output and prepares them for later studies.

Concepts

- **f-strings** – Strings prefixed with f allow you to embed expressions in curly braces. You can apply format specifications inside the braces to control number of decimal places (:.2f), width and alignment. E.g., print(f"Total marks: {total:05d}") prints the integer total padded with leading zeros to five digits.

- **format() method** – Another way to format strings (older than f-strings). For instance, "{:.2f}".format(pi) prints pi rounded to two decimals.
- **Escape sequences** – \n for newline, \t for tab. Use them to structure output lines or columns.

Coding examples

```
# Using f-strings to display two decimal places
pi = 3.14159265
print(f"The value of π rounded to three decimals: {pi:.3f}")

# Aligning numbers in a table
print("x\t x^2\t x^3")
for x in [1, 2, 3]:
   print(f"{x}\t {x**2}\t {x**3}")

# Padding with zeros and aligning text
num = 42
print(f"Padded number: {num:05d}")      # prints 00042
name = "Apsara"
print(f"Centered name: {name:^10}")      # centres within 10-character width
```

Model questions (easy → hard)

1. **Easy** – Write a Python statement using an f-string to display the price 250.0 with two decimal places.
2. **Medium** – Given radius = 7.5, write code to compute the area of a circle (πr^2) and print the result in the form Area = 176.71 m^2 with two decimals. Use pi = 3.14159.
3. **Hard** – The following list contains the names of three students: students = ["Kamal", "Sithara", "Bindu"] and their marks marks = [85, 92, 78]. Write Python code that prints a two-column table showing each student's name left-aligned in a 10-character field and their mark right-aligned in a 3-character field.

3. Reading Input – using input()

Concepts

- **input() basics** – input() pauses program execution and reads a line from the keyboard. When the user presses Enter, the function returns the characters typed **as a string** 【696423437970415†L136-L139】. You can provide a prompt message inside the parentheses to guide the user 【696423437970415†L136-L143】.
- **Always returns a string** – Regardless of what the user types, input() returns a string 【696423437970415†L155-L165】. Combining a string with a number using + causes a TypeError 【696423437970415†L166-L176】. To perform arithmetic, convert the string to the appropriate numeric type using int() or float() 【696423437970415†L174-L196】.
- **Prompting the user** – Passing a meaningful prompt argument improves user experience 【696423437970415†L141-L153】.

- **Type conversion** – Use int(input()) or float(input()) to read numeric values. Handle potential conversion errors in real-world programs.
- **Reading multiple values** – You can read a single line and split it into multiple parts using str.split(), then map each part to the desired type. Example: a, b = map(int, input("Enter two numbers separated by space: ").split()).

Coding examples

```
# Reading a string
name = input("Please enter your name: ")  # returns a string 【696423437970415†L136-L139】
print("Hello", name, "and welcome!")

# Reading an integer and converting
age_str = input("Enter your age in years: ")
age = int(age_str)  # convert string to int 【696423437970415†L174-L196】
print("You are", age * 12, "months old.")

# Reading two floats in one line
length, width = map(float, input("Enter length and width separated by space: ").split())
print(f"Area = {length * width}")

# Reading multiple values into a list of integers
numbers = list(map(int, input("Enter five numbers separated by commas: ").split(',')))
print("Numbers entered:", numbers)
```

Model questions (easy → hard)

1. **Easy** – Write code to prompt the user for their first name and then display a greeting in the form Hello, <name>!.
2. **Medium** – Read two integers from the user (prompt separately) and print their product. Ensure you convert the inputs to integers before multiplication.
3. **Hard** – Write a Python program that reads three floating-point numbers representing the marks of a student in ICT, Mathematics and Physics. Calculate the total and the average and display them using f-strings (two decimal places). Your program should label each output clearly.

4. Combining Input and Output

This section shows how to build small interactive programs by combining input and output operations.

Concepts

- **Prompt–process–display** – The typical pattern for interactive programs: prompt the user with input(), process or compute using the received values, and display results using print().

- **Multiple inputs** – Use split() and map() to read multiple values in one line when appropriate.
- **User-friendly messages** – Use informative prompts and clearly labelled output to enhance usability.

Coding examples

```
# Simple greeting program
name = input("Enter your name: ")
print("Hello", name, "and welcome!")【696423437970415†L263-L289】

# Calculator: sums two integers
x = int(input("Enter first integer: "))
y = int(input("Enter second integer: "))
print(f"{x} + {y} = {x + y}")
print(f"{x} × {y} = {x * y}")

# Currency converter (LKR to USD)
rupees = float(input("Enter amount in LKR: "))
rate = 300  # fixed rate for example
usd = rupees / rate
print(f"{rupees:.2f} LKR is equal to {usd:.2f} USD")

# Electricity bill calculator with fixed slabs
units = float(input("Enter units consumed: "))
if units <= 64:
   cost = units * 5.0
else:
   cost = 64 * 5.0 + (units - 64) * 10.0
print(f"Your bill amount is Rs.{cost:.2f}")
```

Model questions (easy → hard)

1. **Easy** – Ask the user for their favourite colour and then print a sentence stating Your favourite colour is <colour>. Use a single print() call.
2. **Medium** – Prompt for the length and width of a rectangle (floats). Compute the perimeter (2×(l+w)) and area (l×w) and print both results in labelled format using one f-string.
3. **Hard** – Create a program that asks the user for the distance travelled (in kilometres) and the time taken (in hours). Calculate the average speed and display it with one decimal place. Use descriptive prompts and labelled output.

5. Past examination questions (input/output)

Below are typical questions from past A/L ICT papers focusing on sequence and selection with input/output. Use them for revision and exam preparation.

Multiple Choice (3 questions)

1. **A student writes the following Python code:**

```
x = int(input("Enter a number: "))
y = 5
print(x + y)
```

If the user inputs 3, what is printed?

A. 8 B. 35 C. 3 + 5 D. Error because input() returns a string

2. Which statement about the input() function in Python is **true**?

A. It always returns an integer.
B. It pauses execution and returns the typed characters as a string 【696423437970415†L136-L139】.
C. It displays its argument after reading the user's input.
D. It cannot accept a prompt message.

3. What is the output of the following code?

```
a = 10
b = "5"
print(a * b)
```

A. 50 B. 105 C. 5555555555 D. TypeError because multiplication is not defined

Structured questions (3 questions)

1. **Program completion** – Complete the following program to read two integers and print their sum and difference using descriptive labels. You should use int() to convert the input strings to integers.

```
# Reads two numbers and prints their sum and difference
________ = int(input("Enter first number: "))
________ = int(input("Enter second number: "))
sum_val = ________ + ________
diff_val = ________ - ________
print(f"Sum = {sum_val}, Difference = {diff_val}")
```

2. **Currency conversion** – Write a Python program that prompts the user to enter an amount in Sri Lankan Rupees. Assume 1 USD = 300 LKR. Convert the amount to U.S. dollars and display the result rounded to two decimal places. Your output should identify both the original amount and the converted amount.

3. **Grade report** – The variables ICT, Maths, and English hold the marks of a student. Without using conditionals, write a Python script that calculates the total and average

marks, then uses print() and an f-string to display ICT = 85, Mathematics = 90, English = 78, Total = 253, Average = 84.33.

Essay questions (3 questions)

1. **Interaction and data types** – Explain why input() always returns a string and discuss the importance of converting inputs to numeric types when doing arithmetic 【696423437970415†L155-L196】. Write a program that asks the user for the base and height of a triangle, converts them to floats, computes the area, and displays the result with units.

2. **Real-world problem** – Describe a scenario in which a small business needs to calculate the cost of electricity consumed by a customer. Write a Python program that asks the user for the number of units consumed and then computes the bill using the following rates: the first 64 units at Rs. 5.00 per unit and any additional units at Rs. 10.00 per unit. The program should display a neatly formatted bill.

3. **Analytical question** – Discuss how input and output functions support problem solving and human–computer interaction. Then write a Python program that prompts the user for their name, age and monthly salary. The program should calculate their yearly salary and display a summary message like Priya (19 years) earns Rs.<yearly salary> per year. Explain how you would adapt the program to handle invalid numeric input (without actually writing exception handling code).