

LEARN COMPUTING · GCSE COMPUTER SCIENCE

30 Python Programming Challenges

Free Sampler

The first five Bronze challenges from the full pack of thirty



Bronze · Silver · Gold

Ten graded challenges per tier, each with a worked example, success criteria, a hint and an extension task. Written for OCR J277 and AQA 8525 and suitable for every GCSE Computer Science specification.

learncomputing.org

© 2026 Learn Computing. Licensed for use within the purchasing school or by the purchasing individual.

How to use this booklet

Each challenge asks you to write a complete Python program. Work through them in order within a tier - the skills build on each other - and move up a tier when you are getting full marks.

- **Bronze** - variables, input and output, selection and loops. Everyone starts here.
- **Silver** - strings, lists, functions, validation and random numbers.
- **Gold** - searching and sorting algorithms, files, 2D lists and bigger programs that bring everything together.

On every page you will find

- **The task** - what your program must do, and a numbered list of the steps it needs.
- **Example run** - exactly what the finished program looks like when it runs. Match the wording and layout: it is how your work will be checked.
- **Success criteria** - tick each one off as you meet it. The marks add up to 6 (Bronze), 8 (Silver) or 10 (Gold).
- **Hint** - read it only if you are stuck for more than five minutes.
- **Extension** - finished early? Make the program better. Extensions are not marked but they are where the real learning happens.

Rules of the game

- Plan before you type: write the steps in plain English or draw a flowchart first.
- Test with normal, boundary and erroneous data - not just the example shown.
- Use sensible variable names and add a comment where a line is not obvious.
- If it crashes, read the error message from the bottom up. The last line tells you what went wrong; the line above tells you where.

Greeting Generator

Ask for a name and age, then print a personalised message.

Skills: variables · input/output · casting · string formatting

THE TASK

Write a program that asks the user for their name and their age. It should then greet them by name and tell them how old they will be next year.

Your program should:

1. Ask the user for their name and store it in a variable.
2. Ask the user for their age and store it as a whole number (integer).
3. Work out their age next year.
4. Print a single sentence in exactly the format shown in the example.

EXAMPLE RUN

```
What is your name? Priya
How old are you? 14
Hello, Priya! Next year you will be 15.
```

SUCCESS CRITERIA

	Success criteria	Marks
☐	Name is read from the user and stored in a variable	1
☐	Age is read and converted to an integer	1
☐	Age next year is calculated correctly	1
☐	Output matches the required format exactly	2
☐	Sensible variable names used	1
	Total	6

Hint

input() always gives you a string. Use int() to turn the age into a number before adding 1.

Extension

Also tell the user how many years it is until they are 18, and roughly how many days old they are (age × 365).

Cinema Tickets

Calculate the cost of a cinema trip.

Skills: arithmetic · constants · real numbers · formatting to 2 d.p.

THE TASK

A cinema charges £9.50 for an adult ticket and £6.25 for a child ticket. Write a program that asks how many adult and child tickets are needed and prints the total cost.

Your program should:

1. Store the two prices in clearly named constants at the top of the program.
2. Ask for the number of adult tickets and the number of child tickets.
3. Calculate the total cost.
4. Print the total with a £ sign and exactly two decimal places.

EXAMPLE RUN

```
Number of adult tickets: 2
Number of child tickets: 2
Total cost: £31.50
```

SUCCESS CRITERIA

	Success criteria	Marks
☐	Prices stored as named constants	1
☐	Both quantities read and converted to integers	1
☐	Total calculated correctly	2
☐	Output shows £ and exactly two decimal places	2
	Total	6

Hint

`f"{total:.2f}"` prints a number with two decimal places.

Extension

Give a 10% discount when five or more tickets are bought in total. Print the discount and the new total.

Odd or Even

Decide whether a number is odd or even.

Skills: selection · MOD operator · comparison

THE TASK

Write a program that asks for a whole number and reports whether it is odd or even.

Your program should:

1. Ask the user for a whole number.
2. Use the MOD (remainder) operator to test whether it divides exactly by 2.
3. Print the number followed by 'is even' or 'is odd'.

EXAMPLE RUN

```
Enter a whole number: 7
7 is odd
```

SUCCESS CRITERIA

	Success criteria	Marks
☐	Number is read and converted to an integer	1
☐	MOD operator (%) used to find the remainder	2
☐	IF / ELSE selection used correctly	2
☐	Output matches the required format	1
	Total	6

Hint

`number % 2` gives the remainder when number is divided by 2. What is the remainder for an even number?

Extension

Also report whether the number is positive, negative or zero, and whether it is a multiple of 5.

Temperature Converter

Convert between Celsius and Fahrenheit using a simple menu.

Skills: menus · selection (IF / ELIF / ELSE) · real arithmetic

THE TASK

Write a program that shows a two-option menu: convert Celsius to Fahrenheit, or Fahrenheit to Celsius. The user picks an option, enters a temperature, and the program prints the converted value to one decimal place.

Formulae: $F = C \times 9 \div 5 + 32$ and $C = (F - 32) \times 5 \div 9$.

Your program should:

1. Display the menu options numbered 1 and 2.
2. Read the user's choice.
3. For a valid choice, ask for the temperature and perform the correct conversion.
4. Print 'Invalid choice' if the user enters anything else.

EXAMPLE RUN

```
1. Celsius to Fahrenheit
2. Fahrenheit to Celsius
Choose 1 or 2: 1
Temperature in Celsius: 100
100.0°C is 212.0°F
```

SUCCESS CRITERIA

	Success criteria	Marks
☐	Menu displayed and choice read	1
☐	Correct Celsius to Fahrenheit conversion	1
☐	Correct Fahrenheit to Celsius conversion	1
☐	Invalid choice handled with a message	1
☐	Output to one decimal place in the required format	2
	Total	6

Hint

Keep the choice as a string and compare it with "1" and "2" - then a letter typed by mistake will not crash the program.

Extension

Add a third option that converts Celsius to Kelvin ($K = C + 273.15$), and loop the menu until the user chooses to quit.

Times Table

Print any times table from 1 to 12.

Skills: count-controlled loops · FOR · range()

THE TASK

Write a program that asks which times table the user wants and prints it from $1 \times n$ up to $12 \times n$, one line per row.

Your program should:

1. Ask the user for a number.
2. Use a FOR loop that runs exactly 12 times.
3. Print each line in the format ' $3 \times 7 = 21$ '.

EXAMPLE RUN

```
Which times table? 7
1 x 7 = 7
2 x 7 = 14
...
12 x 7 = 84
```

SUCCESS CRITERIA

	Success criteria	Marks
☐	Number read and converted to an integer	1
☐	FOR loop with range() used to repeat exactly 12 times	2
☐	Loop counter used in the calculation	1
☐	Output in the required format for every line	2
	Total	6

Hint

range(1, 13) counts from 1 up to 12 - the end value is not included.

Extension

Let the user choose the start and end of the range, and line the columns up neatly using string formatting.

Want the other 25?

You have just done the first five Bronze challenges. The full pack has thirty:

- **Bronze 1-10** - variables, selection, loops and Boolean logic
- **Silver 11-20** - strings, lists, functions, validation, random numbers and a Caesar cipher
- **Gold 21-30** - bubble sort, binary search, file handling, 2D lists, a login system, a binary converter and a capstone class report

Every challenge comes with a success-criteria checklist and an extension task. Teachers get a mark scheme, a tested model solution and teaching notes for all thirty, plus editable Word files and the solutions as .py files.

Get the full pack at learncomputing.org

Search for "30 Python Programming Challenges" on Tes, or find it under Products on the site.