Flowcharts, pseudocode and program code

1.2.1 · Computational thinking · revision notes

An algorithm is a precise set of steps that solves a problem. You need to follow and write algorithms as flowcharts, pseudocode and Python program code, using sequence, selection, repetition and input/output. Questions ask you to name symbols, work out an output, complete a flowchart or write an algorithm of your own.

Key facts

- Algorithm: a precise sequence of steps that solves a problem
- Process (rectangle): a calculation or an assignment
- Input/output (parallelogram): input data or output results
- Sequence: in order. Selection: choose a path (if). Repetition: loop (for / while)
- Iteration: for item in a list (or string) runs once for every item (or character)
- Terminal (rounded ends): Start / Stop
- Decision (diamond): a question with two labelled exits
- Subprogram (rectangle with double side lines): call a subprogram
- Count-controlled loop (for): a set number of times. Condition-controlled loop (while): repeats while a condition is True
- range(a, b) counts from a up to b − 1

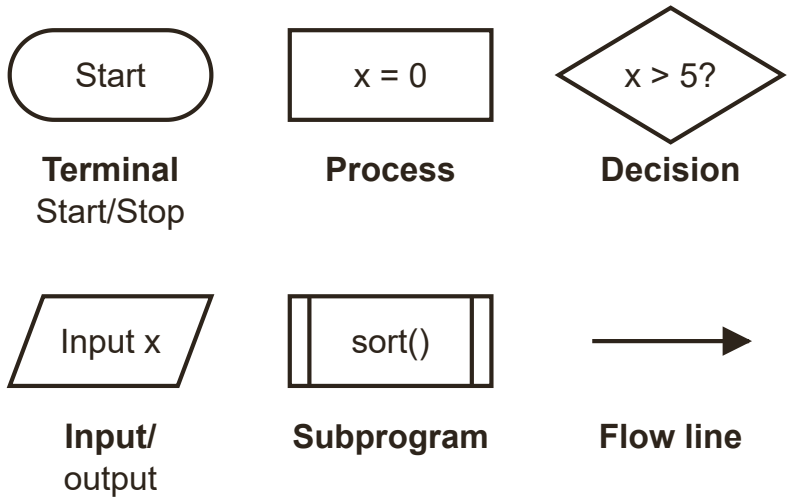
Notes

Three ways to show an algorithm

- An **algorithm** is a precise, ordered set of steps that solves a problem.
- A **flowchart** shows the steps as symbols joined by arrows.
- **Pseudocode** writes the steps as structured, English-like statements. It is not a real programming language, so there is no strict syntax, but it must be clear and unambiguous.
- **Program code** for this course is Python 3.
- When a written question asks you to write an algorithm, you can usually use any of the three, unless the question says which one to use.

Flowchart symbols

- **Terminal** (rectangle with rounded ends): Start and Stop.
- **Process** (rectangle): a calculation or an assignment, e.g. `total = total + price`.
- **Decision** (diamond): a question with two exits, labelled Yes/No or True/False.
- **Input/output** (parallelogram): e.g. Input age, Output total.
- **Subprogram** (rectangle with a double line down each side): calls a subprogram that has its own flowchart.
- **Arrows** (flow lines) show the order of the steps. A loop is an arrow that goes back to an earlier step.



The six flowchart symbols. A decision always has two exits, labelled Yes and No

The building blocks

- **Sequence**: steps carried out one after another, in order.
- **Selection**: a condition decides which steps run next. In Python: `if`, `elif`, `else`.
- **Repetition**: steps are repeated. A **count-controlled** loop repeats a set number of times (`for`). A **condition-controlled** loop repeats while a condition is True (`while`), so the number of repeats may not be known in advance.
- **Iteration** over a data structure: a `for` loop visits every item in turn. `for score in scores:` runs once for each item in the list; `for letter in word:` runs once for each character.
- **Input/output**: `input()` gets data from the user as a string (use `int()` or `float()` to turn it into a number); `print()` displays data.
- This program uses all four: sequence, a count-controlled loop (it runs 3 times), selection, and input/output.

```
total = 0
for count in range(3):
    price = float(input("Enter price: "))
    total = total + price
if total > 20:
    print("Free delivery")
else:
    print("Delivery costs £3")
```

- `range(1, 6)` gives 1, 2, 3, 4, 5: it stops **before** the end value. `range(3)` gives 0, 1, 2.

How to answer each type of question

Identify flowchart symbols 1 mark each **Grade 3**

1. Match the shape to its job: rounded ends = Start/Stop, rectangle = process, diamond = decision, parallelogram = input/output.
2. Give the name or the purpose of the symbol, whichever the question asks for.

Example. State the purpose of each flowchart symbol.



(a) (b) (c)

(a) A diamond
(b) A parallelogram
(c) A rectangle with rounded ends

Model answer.

(a) Decision: a question with two possible outcomes, such as Yes or No (1)
(b) Input or output of data (1)
(c) Terminal: the start or the end of the algorithm (1)

State the output of an algorithm

1 to 3 marks | Grade 4

1. Work through one line (or symbol) at a time, keeping track of each variable.
2. At each decision, check the condition carefully, including the boundary (< or <=).
3. Write the output exactly as it would be displayed.

Example. Here is a program.

```
age = int(input("Enter age: "))
if age < 5:
    price = 0
elif age < 16:
    price = 4
else:
    price = 7
print("Price:", price)
```

State the output when the user enters:
(a) 16
(b) 4
(c) 15

Model answer.

(a) Price: 7 (1), because 16 is not less than 16
(b) Price: 0 (1)
(c) Price: 4 (1)

Explain the difference between the two types of loop

2 marks | Grade 6

1. Describe the count-controlled loop: it repeats a set number of times, known before it starts.
2. Describe the condition-controlled loop: it repeats while a condition is True (or until it becomes True).

Example. Explain the difference between a count-controlled loop and a condition-controlled loop.

Model answer. A count-controlled loop repeats a fixed number of times, which is known before the loop starts (1). A condition-controlled loop keeps repeating while a condition is True, so the number of repeats may not be known in advance (1).

Write an algorithm 4 to 6 marks **Grade 7**

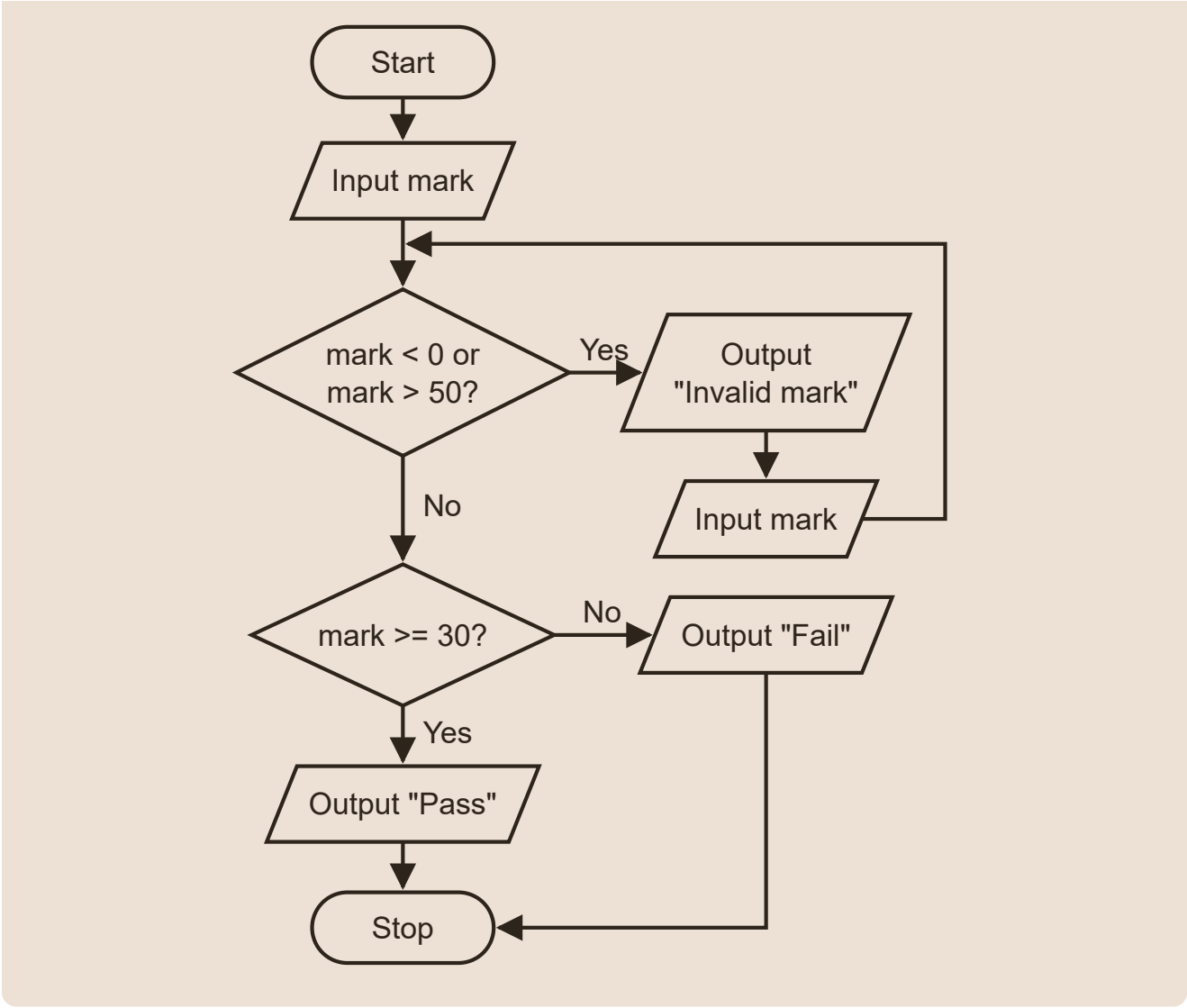
1. List what the algorithm must do: the inputs, the processing and the outputs.
2. Choose the right constructs: a while loop to repeat until an input is valid, a for loop for a fixed number of repeats, if for choices.
3. Use clear names and indent the steps inside each loop and if.
4. Check it with a test value, including a boundary value.

Example. Write an algorithm that:
• asks the user to enter a test mark
• keeps asking until the mark is between 0 and 50 inclusive
• then outputs `Pass` if the mark is 30 or more, otherwise outputs `Fail`.

Model answer.

```
mark = int(input("Enter mark: "))
while mark < 0 or mark > 50:
    print("Invalid mark")
    mark = int(input("Enter mark: "))
if mark >= 30:
    print("Pass")
else:
    print("Fail")
```

Inputs a mark (1); a loop that repeats while the mark is invalid (1); a correct condition for an invalid mark, below 0 or above 50 (1); inputs the mark again inside the loop (1); outputs Pass for 30 or more and Fail otherwise (1).
Pseudocode or a flowchart with the same logic earns the same marks. The same algorithm as a flowchart:



Don’t lose marks

- Using a rectangle for an input or an output, or leaving the exits of a decision unlabelled.
- Forgetting to input again inside a validation loop, so the loop never ends (an infinite loop).
- Getting `range` wrong: `range(1, 10)` stops at 9, not 10.
- Forgetting `int()` or `float()` around `input()`, so the value is a string and cannot be used in a calculation.
- Leaving out Start or Stop in a flowchart, or drawing an arrow that leads nowhere.

More tips

Memory tricks

- Shapes: rounded ends start and stop, diamonds decide, parallelograms talk to the user, rectangles do the work.
- A decision diamond always has two exits. Label both of them.
- `for` = a fixed number of times. `while` = until something changes.
- Validation pattern: input; then while the value is invalid, show an error and input again.

Exam technique

- Read the question to see whether a particular form (flowchart, pseudocode or Python) is asked for.
- Pseudocode has no fixed rules, but indent the steps inside loops and selections so that the structure is clear.
- Before you move on, trace your algorithm with one normal value and one boundary value.
- Use the variable names and messages given in the question.

What each grade needs

What you need to be able to do, from the first marks up to the top grade.

Grade 3 — **Name the standard flowchart symbols**
Terminal, process, decision, input/output and subprogram symbols, joined by arrows.

Grade 4 — **Identify sequence, selection and repetition**
Spot each construct in a flowchart, in pseudocode or in Python code.

Grade 4 — **Follow an algorithm to find its output**
Work through a flowchart or program step by step with the inputs given.

Grade 5 — **Complete a partly drawn flowchart**
Fill in missing decisions, processes or outputs so that it does what is described.

Grade 6 — **Compare count-controlled and condition-controlled loops**
A count-controlled loop repeats a set number of times; a condition-controlled loop repeats while a condition is True.

Grade 7 — **Write an algorithm using selection and repetition**
For example, a loop that keeps asking for an input until the value is valid.

Grade 8 — **Convert between flowcharts, pseudocode and code**
Turn a flowchart into Python, or Python into a flowchart, keeping exactly the same logic.

Free notes, Full packs, practice papers and mock exams at chhetriacademy.com. Written for this site; not taken from past papers. Stuck? 1-to-1 tutoring with a free 30-minute trial lesson.