Sequence, selection and iteration

Algorithms and programming (Paper 2) · revision notes

Every program is built from three constructs: sequence, selection and iteration. You need to recognise them in code, know the difference between count-controlled and condition-controlled loops, and use if, switch, for, while and do … until in your own code.

Key facts

- Sequence: instructions run in order, one after another
- Count-controlled: for i = 1 to 10 … next i (a fixed number of times)
- while … endwhile: condition checked at the start (may run 0 times)
- Nesting: one construct inside another
- Selection: if … then … elseif … else … endif, or switch … case … default … endswitch
- for … step 2 counts in 2s; step -1 counts down; the end value is included
- do … until: condition checked at the end (runs at least once)

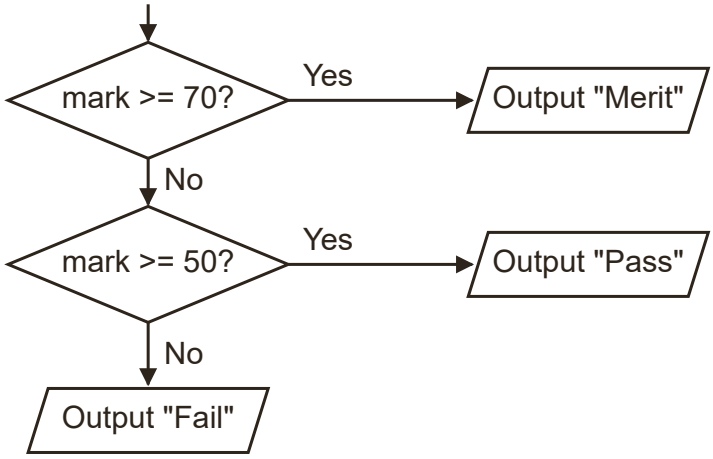
Notes

Sequence and selection

- **Sequence**: instructions are carried out one after another, in the order they are written.
- **Selection**: a condition decides which instructions run next.

```
if mark >= 70 then
   print("Merit")
elseif mark >= 50 then
   print("Pass")
else
   print("Fail")
endif
```

- Only the first true condition runs, so the order of the conditions matters.



Selection as a flowchart: only the first true condition's branch runs

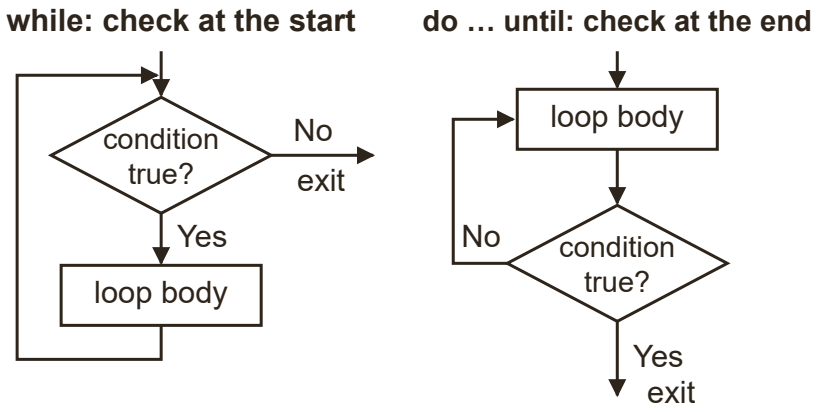
- A **switch** (case) statement selects based on one variable's value:

```
switch choice:
   case "P":
      print("Play")
   case "Q":
      print("Quit")
   default:
      print("Unknown option")
endswitch
```

Iteration (loops)

- **Count-controlled**: repeats a set number of times. `for i = 1 to 5 … next i` runs 5 times. `for i = 10 to 0 step -2` counts down in 2s: 10, 8, 6, 4, 2, 0.
- **Condition-controlled**: repeats until a condition changes, so the number of repeats is not known in advance.
- `while … endwhile` checks the condition at the **start**, so the code inside may never run.
- `do … until` checks the condition at the **end**, so the code inside always runs at least once.

```
do
   pin = input("Enter a 4-digit PIN")
until pin.length == 4
```



while checks before the loop body; do … until checks after it

Nesting

- Constructs can be put inside each other, for example an if statement inside a loop. This is called **nesting**.
- Indent each level so it is clear which lines belong to which loop or if statement.

How to answer each type of question

Identify the constructs in code

1 to 3 marks **Grade 4**

1. Selection: look for if or switch.
2. Count-controlled iteration: for … next.
3. Condition-controlled iteration: while or do … until.
4. Give the line number(s) the question asks for.

Example. Look at the code.

```
01 total = 0
02 for day = 1 to 7
03    steps = int(input("Steps today?"))
04    total = total + steps
05 next day
06 if total >= 70000 then
07    print("Target met")
08 endif
```

(a) Identify the line where selection starts. [1]
(b) State the type of iteration used and explain how you know. [2]

Model answer.

(a) Line 06 (1)
(b) Count-controlled iteration (1): it is a for loop that runs a fixed number of times, 7 (1).

State the output of a loop

1 to 2 marks **Grade 5**

1. Write down the value of the loop variable each time round.
2. For a for loop, remember the end value is included if it is reached exactly.
3. For a while loop, check the condition before every pass.

Example. State the output of this code.

```
for i = 1 to 10 step 3
   print(i)
next i
```

[2]

Model answer. 1, 4, 7 (1) and 10 (1).

Compare types of iteration

2 marks | Grade 5

1. Count-controlled: repeats a fixed, known number of times.
2. Condition-controlled: repeats until (or while) a condition is met, so the number of repeats is not known in advance.

Example. Describe the difference between count-controlled and condition-controlled iteration. [2]

Model answer. Count-controlled iteration repeats a set number of times (1). Condition-controlled iteration repeats until a condition is met, so the number of repetitions does not need to be known in advance (1).

Write code with a condition-controlled loop

3 marks | Grade 7

1. Pick do … until if the code must run at least once (for example asking for input).
2. Put the input inside the loop.
3. Write the condition for when the loop should stop.

Example. Write an algorithm that keeps asking the user to enter a whole number from 1 to 10 until a valid number is entered. [3]

Model answer.

```
do
   num = int(input("Enter a number from 1 to 10"))
until num >= 1 AND num <= 10
```

Condition-controlled loop (1); input inside the loop (1); correct condition for 1 to 10 inclusive (1).

Don’t lose marks

- Putting conditions in the wrong order, such as checking mark >= 50 before mark >= 70, so a Merit is never given.
- Forgetting that a for loop includes its end value.
- Writing a while loop whose condition never becomes false (an infinite loop), for example forgetting to input a new value inside the loop.
- Saying a while loop always runs at least once: that is do … until.
- Missing endif, endwhile or next, or indenting so that a line is in the wrong loop.

More tips

Memory tricks

- SSI: Sequence = in order, Selection = choose, Iteration = repeat.
- while checks first ('look before you leap'); do … until checks last ('leap, then look').
- In an if … elseif chain, put the most demanding condition first (for example 70 before 50).

Exam technique

- When a loop must repeat a known number of times, use a for loop; examiners look for the right type of loop.
- Close every construct (endif, next i, endwhile, until) so your structure is clear.
- Trace a loop for its first two and its last time round before writing the output.

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 three programming constructs**
Sequence (in order), selection (decisions) and iteration (repetition).

Grade 4 — **Identify each construct in given code**
Point to the lines that use if or switch, and the lines that use a loop.

Grade 5 — **Compare count- and condition-controlled loops**
A for loop repeats a set number of times; while and do … until repeat depending on a condition.

Grade 5 — **Work out the output of a loop**
Follow for loops with step values and while loops, including the last time round.

Grade 6 — **Write code using if, elseif and else**
Choose the right order of conditions so that each case gives the correct result.

Grade 7 — **Choose and write the most suitable loop**
Use for when the number of repeats is known and do … until when it must run at least once.

Grade 8 — **Use nested selection and iteration**
Put loops and if statements inside each other, with correct indentation and 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.