2025 CpSc H Q3

Section: Software Design & Development

Topic: Implementation (Algorithm Specification)

Question Summary

Design an algorithm that counts the number of upper-case and lower-case letters in a given string variable sentence. Use ASCII ranges: upper-case 65–90, lower-case 97–122. For the example input “To be or not to be?”, the expected output is: Upper case = 1, Lower case = 12.

Worked Solution

Idea: Traverse each character of sentence. Use the ASCII code to decide whether to increment the upper-case or lower-case counter; ignore any other character (spaces, punctuation, digits).

Pseudocode (structured English)

```
SET upperCount ← 0
SET lowerCount ← 0
FOR each character ch IN sentence DO
SET code ← ASCII code of ch
IF 65 ≤ code ≤ 90 THEN
SET upperCount ← upperCount + 1
ELSE IF 97 ≤ code ≤ 122 THEN
SET lowerCount ← lowerCount + 1
END IF
END FOR
OUTPUT "Upper case: ", upperCount
OUTPUT "Lower case: ", lowerCount
```

Trace on the example

Input: “To be or not to be?” Letters encountered = T,o,b,e,o,r,n,o,t,t,o,b,e → upperCount = 1 (the ‘T’), lowerCount = 12. Spaces and ‘?’ are ignored.

Notes & Pitfalls

• Only ASCII letters are counted. Symbols, punctuation, spaces and digits must be ignored (no counter change).

• Using ASCII ranges avoids locale issues — stick to 65–90 for ‘A’..‘Z’ and 97–122 for ‘a’..‘z’.

• If your language offers isUpper/isLower, using those is also acceptable — but this solution matches the question’s ASCII guidance.

Final Answer

Algorithm above (structured English). For input “To be or not to be?” the program outputs: Upper case: 1 and Lower case: 12.

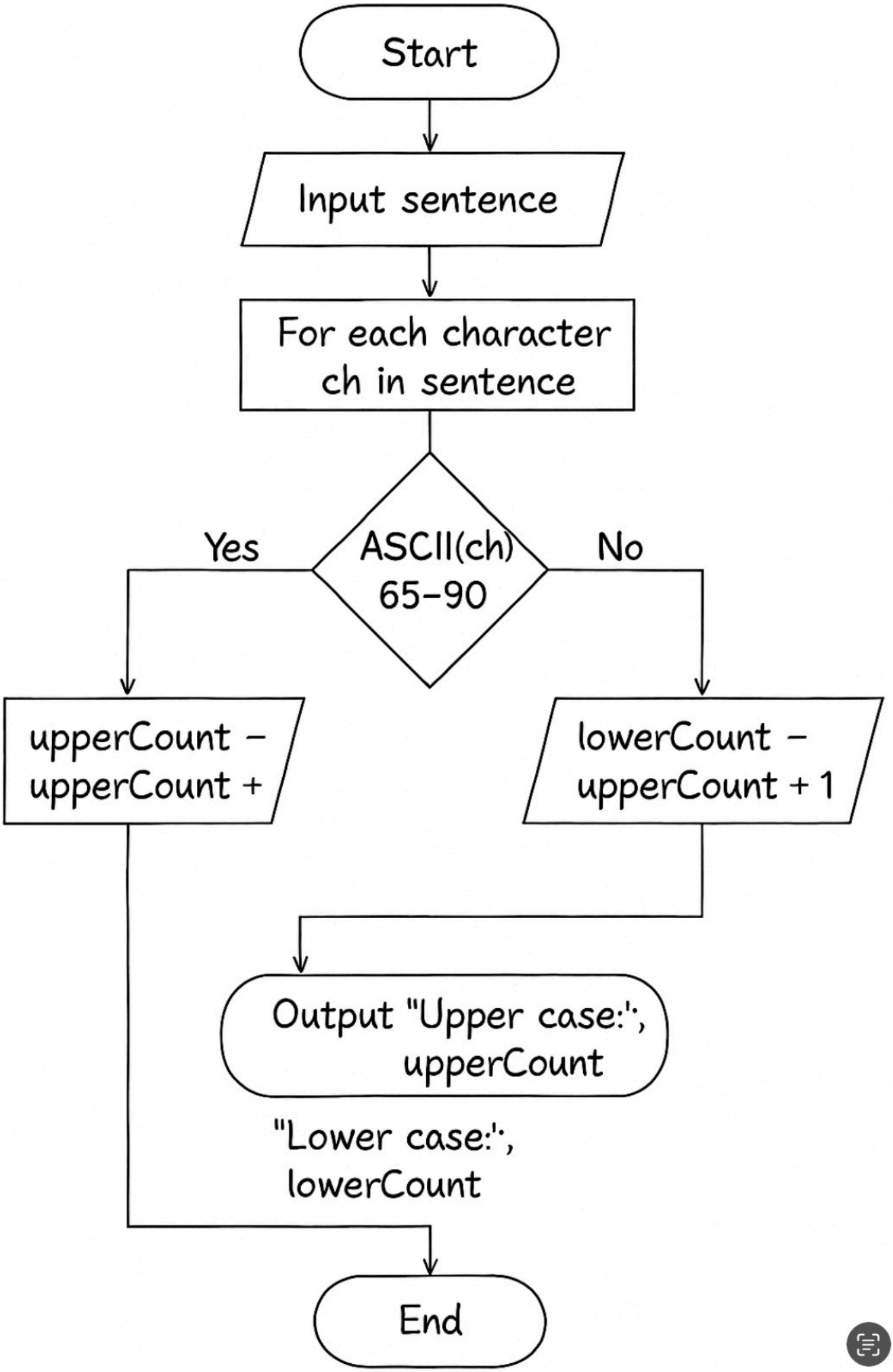
Revision Tips

• ASCII checkpoints: 65–90 = ‘A’..‘Z’; 97–122 = ‘a’..‘z’.

• When a question asks for a design, write clear, language-neutral steps (no specific syntax required).

• Show that you ignore non-letters; an explicit ‘ELSE do nothing’ earns clarity marks.

• Time complexity is O(n) for n characters — one pass is ideal.


Start
Input sentence
For each character
ch in sentence
ASCII(ch)
65-90
Yes
No
upperCount -
upperCount +
lowerCount -
upperCount + 1
Output "Upper case:",
upperCount
"Lower case:",
lowerCount
End