1.4 DATA TYPES & STRUCTURES · 1.4.1

Character sets: ASCII & Unicode

How text is stored as binary, and how the two character sets compare. Spec 1.4.1(j).

01 The idea

Char set	Maps each character → a unique numeric code.
Stored as	That code in **binary**.
Why agree	Same set on all devices → text displays the same.
Note	The *character* '5' ≠ the number 5 (its code is 53).

02 ASCII

Bits	**7 bits** → **128** codes (0–127).
Covers	English letters, digits, punctuation, controls.
Digits	'0'–'9' = 48–57.
Capitals	'A'–'Z' = 65–90 (in order → countable).

03 Unicode

Bits	**16+ bits** → **1,000,000+** code points.
Covers	(Almost) all world languages + symbols + emoji.
Overlap	First **128 codes = ASCII**.
Cost	More bits/char → **more storage**.

04 Compare (exam gold)

Same	Both map chars → binary codes; Unicode ⊇ ASCII (first 128).
Differ	Unicode: more chars + more languages, but more memory.
Use Unicode	When you need many languages / emoji.

FINAL PASS BEFORE THE EXAM

Rapid exam tips

Six slips on character-set questions.

01

ASCII = **7 bits**, 128 codes (often stored in a byte, but the code is 7-bit).

02

Unicode exists to represent **many languages** + symbols/ emoji.

03

Similarity: both map chars to **binary codes**; first 128 codes match.

04

Difference/trade-off: Unicode = more chars but **more storage**.

05

Letters are **in order** — find 'D' as 'A'(65) + 3 = 68.

06

The character '5' has code 53 — **not** the value 5.