#1: What does “sufficient condition” guarantee?

If you meet a sufficient condition, you will get what you are looking for (the results you are for are guaranteed).

#2: If a sufficient condition is not satisfied, what conclusion can we draw?

No conclusion (“may or may not”).

#3: What does “necessary condition” guarantee?

If you do not meet a necessary condition, it is guaranteed that you will NOT get what you are looking for.

#4: If a necessary condition is satisfied, what conclusion can we draw?

No conclusion (“may or may not”).

#5: What are the two primary advantages of learning “computer organization and architecture” by CS majors?

#6: What does “efficient” in “efficient computer programs” mean? Mention at least two.

#7: What is “Moore’s Law”?

#8: What does “MIPS” stand for?

#9: What are “super computers” (how fast, what purpose and how large)?

#10: What is “Von-Neumann Architecture”?

#11: What does “high-level” in “high-level programming languages” mean?

#12: What does “low-level” in “low-level programming languages” mean?
#13: Mention typical hardware components in a computer system (at least five examples).

- Processors (GPUs)
- Memory
- Hard drives
- Mother boards
- Power
- Network Interface Card (NIC)

#14: Mention the three different groups of software in a computer system.

- Application programs (any programs human users use)
- Operating systems (Windows, UNIX, MacOS, …)
- Device drivers

#15: What does “performance” in computer system mean (at least four examples)?

- Execution speed (take less time to get some job done)
- Power consumption
- Memory size (memory footprint size)
- Reliability (less program bugs)

#16: Where executable programs exist (before they are executed by a processor) in a computer system)?

Memory

#17: What are the three different types of “program files”? Which format do processors understand?

#18: What are the languages used in (human) programmers’ source code files?

#19: What are the languages used in assembly source code files?

#20: What are the languages used in binary executable files?

Machine language

#21: What is the primary purpose of “(human) high-level programmers’ source code files”?

#22: How will “programmers’ life” be easier if programmers develop programs using high-level programming languages? Mention at least three different ways.

#23: What is the primary purpose of “binary executable files”? 


#24: What is the primary purpose of “assembly source code files”?

#25: What is the software tool that translates “(human) programmers’ source code files” into “assembly source code files”?

#26: What is the software tool that translates “assembly source code files” into “binary executable files”?

#27: What is the relation between “statements” in high-level programming languages and “instructions” in assembly languages? Select the best option that represents their relation in the following options.

(a) one (statement)-to-one (instruction)

(b) one (statement)-to-many (instructions)

(c) many (statements) to-one (instruction)

(d) many (statements)-to-many (instructions)

(e) none of the above

[This list continues to the next page]
#28: What is the relation between “instructions” in assembly languages and “machine codes”? Select the best option that represents their relation in the following options.

(a) one (instruction)-to-one (machine code)
(b) one (instruction)-to-many (machine codes)
(c) many (instructions) to-one (machine code)
(d) many (instructions)-to-many (machine codes)
(e) none of the above