Chapter 1:

1. Define scalability in the context of a software organization.
2. Give two resources in a software organization that must scale.
3. What is the Churn Rule and why does it scale well?
4. Give an example of a policy that does not scale in large organizations.
5. Explain *Time and Change* in software engineering.
6. Differentiate between *"It works"* and *"It is maintainable"* in software Engineering aspect.
7. Explain with an example a policy that does not scale and a policy that scales well in large organizations.
8. Why do compiler upgrades tend to be expensive in large codebases?
9. Mention two factors that improve codebase flexibility when performing upgrades.
10. Why is expertise considered to scale better than asking every team to do the same task?
11. State one human cost and one compute cost that can limit scalability.
12. Define scalability in the context of a software organization. Explain why human effort, compute resources, and the codebase itself need to scale.