1. Что такое оркестрация контейнеров в наиболее общем смысле?

2. Какая из перечисленных проблем НЕ решается с помощью оркестратора?

3. Оркестрация контейнеров является ключевым элементом в практике DevOps, потому что она:

4. Что является главной философской основой работы таких оркестраторов, как Kubernetes?

5. Что из перечисленного является прямым следствием использования оркестратора?

6. Контейнеризация (например, Docker) решает проблему переносимости приложений, а оркестрация решает проблему:

7. Какая из перечисленных практик DevOps напрямую зависит от возможностей оркестратора?

8. По аналогии с дирижером оркестра, оркестратор контейнеров:

9. Что из перечисленного НЕ является преимуществом перехода на оркестрацию контейнеров?

10. Как оркестратор способствует практике «Infrastructure as Code» (IaC)?

11. Основное различие между виртуальными машинами и контейнерами заключается в том, что контейнеры:

12. Зачем в DevOps нужна оркестрация контейнеров?

13. Что означает принцип «самовосстановления» (self-healing) в контексте оркестрации?

14. Какая из перечисленных технологий является стандартом де-факто для оркестрации контейнеров?

15. Что такое «кластер» в контексте оркестрации?

16. Как оркестратор помогает командам разработки?

17. Что из перечисленного является вызовом, который призвана решить оркестрация?

18. Какое утверждение о связи CI/CD и оркестрации является верным?

19. Что такое «нода» (узел) в кластере?

20. Какой из перечисленных пунктов лучше всего описывает основное назначение оркестратора?

21. Что такое «декларативный подход» в оркестрации?

22. Какой компонент оркестратора отвечает за принятие решения о том, на каком узле запустить контейнер?

23. Что такое «манифест» в Kubernetes?

24. В чем разница между горизонтальным (scaling out) и вертикальным (scaling up) масштабированием?

25. Что такое Horizontal Pod Autoscaler (HPA) в Kubernetes?

26. Для чего нужна Liveness Probe?

27. Для чего нужна Readiness Probe?

28. Что произойдет, если контейнер fail Liveness Probe?

29. Что произойдет, если контейнер fail Readiness Probe?

30. Что такое стратегия Rolling Update?

31. Какова главная цель использования стратегии Rolling Update?

32. Что позволяет сделать механизм «отката» (rollback)?

33. Что такое «ресурсные запросы» (requests) в конфигурации контейнера?

34. Что такое «лимиты ресурсов» (limits) в конфигурации контейнера?

35. Что произойдет, если контейнер превысит свой лимит по памяти?

36. Как оркестратор обеспечивает отказоустойчивость (self-healing)?

37. Что такое «сервис» (Service) в Kubernetes?

38. Как оркестратор обычно обеспечивает обнаружение сервисов (service discovery)?

39. Какая из перечисленных задач НЕ входит в основные функции оркестратора?

40. Что главным образом описывается в манифесте развертывания (Deployment)?

41. Какая из перечисленных систем оркестрации является самой распространенной и имеет наибольшее сообщество?

42. Какое утверждение о Docker Swarm является верным?

43. Apache Mesos является уникальным по сравнению с другими оркестраторами, потому что он:

44. Какой оркестратор разработан компанией HashiCorp и известен своей простотой и гибкостью в работе с различными типами задач (контейнеры, виртуальные машины, приложения)?

45. Что изначально было ключевым компонентом для управления ресурсами в экосистеме Apache Mesos?

46. Какая система оркестрации часто характеризуется как «легковесная» и имеющая минимальные требования к ресурсам для управления кластером?

47. Какой оркестратор использует архитектуру «менеджер-воркер» (manager-worker) и встроен прямо в Docker?

48. Какая система изначально создавалась внутри Google и черпала идеи из их внутренней системы Borg?

49. Какой оркестратор позиционируется как «универсальный планировщик» и может работать вместе с другими фреймворками, например, с Marathon или Chronos?

50. Для какого оркестратора типично использование сложной терминологии с такими понятиями, как Pods, Services, Deployments, DaemonSets?