9. \*\*Интернет\*\* - Всемирная компьютерная сеть, построенная на основе стека протоколов TCP/IP. Количество пользователей: около 3 млрд.

10. \*\*Служба Интернет\*\* - это виды услуг, оказываемых серверными службами сети Интернет.

11. \*\*Узел сети Интернет\*\* - это устройство, подключенное к сети Интернет, способное обмениваться данными с другими устройствами в этой сети.

12. \*\*Клиент-серверное приложение\*\* - это архитектурная модель сетевого приложения, в которой клиентские устройства отправляют запросы на сервер, а серверы обрабатывают эти запросы и возвращают клиентам результаты.

13. \*\*Сетевой протокол\*\* - это набор правил и соглашений, определяющих формат и порядок обмена данными между устройствами в сети.

14. \*\*Основные свойства протокола HTTP\*\*:

- Простота и читаемость.

- Базируется на клиент-серверной модели.

- Без состояния (stateless).

- Поддерживает кеширование.

15. \*\*Состав информации, пересылаемой в HTTP-запросе\*\*:

метод;

URI;

версия протокола (HTTP/1.1);

заголовки (пары: имя/заголовок);

параметры (пары: имя/заголовок);

расширение

16. \*\*Состав информации, пересылаемой в HTTP-ответе\*\*:

версия протокола (HTTP/1.1);

код состояния (1xx, 2xx, 3xx, 4xx, 5xx);

пояснение к коду состояния;

заголовки (пары: имя/заголовок);

расширение

17. \*\*Группы заголовков HTTP\*\*:

- Общие заголовки.

- Заголовки запроса.

- Заголовки ответа.

- Заголовки сущности.

18. \*\*Web-приложение\*\* - это программное обеспечение, которое работает на удаленном сервере и доступно через интернет посредством веб-браузера. Оно обычно состоит из клиентской (frontend) и серверной (backend) частей.

19. \*\*Frontend\*\* - это часть веб-приложения, которая выполняется на стороне клиента (обычно в браузере) и отвечает за визуальную часть и взаимодействие с пользователем. \*\*Backend\*\* - это часть веб-приложения, которая выполняется на сервере и отвечает за обработку запросов от клиентов, доступ к базам данных и другие серверные операции.

20. \*\*Кроссплатформенное приложение\*\* - это приложение, способное работать на различных операционных системах без изменения исходного кода.

21. \*\*Общая схема web-приложения\*\*:

- Клиентский (frontend) и серверный (backend) компоненты.

- Клиент взаимодействует с сервером через интерфейс (обычно браузер).

- Сервер обрабатывает запросы клиента и возвращает данные или результаты обработки.

22. \*\*Основные технологии разработки серверных кроссплатформенных приложений\*\*:

- Node.js

- Python (с использованием фреймворков Django, Flask)

- Ruby (с использованием фреймворков Ruby on Rails)

- Java (с использованием фреймворков Spring, Jakarta EE)

23. \*\*Асинхронная операция\*\* - это операция, которая выполняется независимо от основного потока выполнения программы, позволяя продолжать выполнение других операций параллельно.

24. \*\*Основное назначение NODE.JS\*\* - это выполнение JavaScript-кода на сервере, что позволяет разработчикам создавать масштабируемые и эффективные сетевые приложения.

25. \*\*Основные свойства NODE.JS\*\*:

- Однопоточность событийного цикла.

- Асинхронный ввод/вывод.

- Модульность и расширяемость.

- Высокая производительность.

- Кроссплатформенность.