№1

Прикладной уровень — верхний уровень модели, обеспечивающий взаимодействие пользовательских приложений с сетью

Уровень представления обеспечивает преобразование протоколов и кодирование/декодирование данных

Сеансовый уровень модели обеспечивает поддержание сеанса связи, позволяя приложениям взаимодействовать между собой длительное время

Транспортный уровень модели предназначен для обеспечения надёжной передачи данных от отправителя к получателю

Сетевой уровень модели предназначен для определения пути передачи данных

Канальный уровень предназначен для обеспечения взаимодействия сетей на физическом уровне и контроля ошибок, которые могут возникнуть

Физический уровень — нижний уровень модели, который определяет метод передачи данных, представленных в двоичном виде, от одного устройства (компьютера) к другому

№2

**TCP (протокол управления передачей)** обеспечивает передачу информации, проверяя дошла ли она, насколько полным является объем полученной информации и т.д. TCP дает возможность двум хостам производить обмен пакетами через установку соединения. Он предоставляет услугу для приложений, повторно запрашивает потерянную информацию, устраняет дублирующие пакеты, регулируя загруженность сети. TCP гарантирует получение и сборку информации у адресата в правильном порядке.

TCP требует явного указания максимального размера сегмента ([MSS](https://ru.wikipedia.org/wiki/MSS)) в случае, если виртуальное соединение осуществляется через сегмент сети, где максимальный размер блока ([MTU](https://ru.wikipedia.org/wiki/Maximum_transmission_unit)) менее, чем стандартный MTU Ethernet (1500 байт).

**UDP (протокол пользовательских датаграмм)** — он занимается передачей автономных датаграмм. UDP не гарантирует, что всех датаграммы дойдут до получателя. Датаграммы уже содержат всю необходимую информацию, чтобы дойти до получателя, но они все равно могут быть потеряны или доставлены в порядке отличном от порядка при отправлении. Позволяет отправлять информацию (датаграммы) по IP-сети без предварительного установления соединения и создания специального виртуального канала или путей данных.

№3

Нужны адреса и протоколы – принятые соглашения для обмена данных.

№4

Активный сокет осуществляет взаимодействие между сервером и пользователями (через него читаем и пишем). Пассивный сокет проверяет запросы на взаимодействие и, если с какого-нибудь адреса (зависит от ситуации) подали запрос на взаимодействие с сервером), передаёт работу с клиентом активному сокету.

№5

Для TCP-сервера:

1. Делает пассивный сокер - acceptor
2. связывает с ним порт – bind(endpoint)
3. Переводит в режим прослушки listen(size)
4. Делает активный сокет - socet
5. Передаёт в активный сокет – accept(socet)

Для TCP-клиента:

1. Делает активный сокет
2. Осуществляет коннект: socet.connect()