Question 1

What permissions does your app need to connect to the internet?

- android.permission.CONNECTIVITY
- android.permission.INTERNET
- It doesn't need any special permissions, because all Android apps are allowed to connect to the internet.

  Ans : android.permission.INTERNET

Question 2

How does your app check that internet connectivity is available?

In the manifest:

- request ACCESS_NETWORK_STATE permission
- request ALL_NETWORK_STATE permission
- request NETWORK_CONNECT permission

In the code:

- Wrap the code to connect to the internet in a try/catch block, and catch NO_NETWORK errors.
- Use ConnectivityManager to check for an active network before connecting to the network.
- Present a dialog to the user reminding them to make sure that internet connectivity is available before they attempt to connect to the internet.

  Ans :

  - Request ACCESS_NETWORK_STATE permission
  - Use ConnectivityManager to check for an active network before connecting to the network.

Question 3

Where do you implement the loader callback method that's triggered when the loader finishes executing its task?

- In the AsyncTaskLoader subclass. The AsyncTaskLoader must implement LoaderManager.LoaderCallbacks.
- In the Activity that displays the results of the task. The Activity must implement LoaderManager.LoaderCallbacks.
- In a Utility class that extends Object and implements LoaderManager.LoaderCallbacks.

  Ans: In a Utility class that extends Object and implements LoaderManager.LoaderCallbacks.

Question 4

When the user rotates the device, how do AsyncTask and AsyncTaskLoader behave differently if they are in the process of running a task in the background?

- A running AsyncTask becomes disconnected from the activity, but keeps running. A running AsyncTaskLoader becomes disconnected from the activity and stops running, preserving system resources.
- A running AsyncTask becomes disconnected from the activity and stops running, preserving system resources. A running AsyncTaskLoader automatically restarts execution of its task from the beginning. The activity displays the results.
- A running AsyncTask becomes disconnected from the activity, but keeps running. A running AsyncTaskLoader automatically reconnects to the activity after the device rotation. The activity displays the results.

  Ans : A running AsyncTask becomes disconnected from the activity and stops running, preserving system resources. A running AsyncTaskLoader automatically restarts execution of its task from the beginning. The activity displays the results.

Question 5

How do you initialize an AsyncTaskLoader to perform steps, such as initializing variables, that must be done before the loader starts performing its background task?

- In onCreateLoader() in the activity, create an instance of the AsyncTaskLoader subclass. In the loader's constructor, perform initialization tasks.
- In onCreateLoader() in the activity, create an instance of the AsyncTaskLoader subclass. In the loader's init() method, perform initialization tasks.
- In the Activity, implement initLoader() to initialize the loader.
- Perform initialization tasks for the loader at the start of loadInBackgroud() in the Loader.

  Ans : In onCreateLoader() in the activity, create an instance of the AsyncTaskLoader subclass. In the loader's constructor, perform initialization tasks.

Question 6

What methods must an AsyncTaskLoader implement?

Ans:

- onStartLoading
- loadInBackground