Q. What is functional programming ?

Ans - **Functional programming** (often abbreviated FP) is the process of building software by composing **pure functions**, avoiding **shared state,** **mutable data,**and **side-effects**. Functional programming is **declarative** rather than **imperative**, and application state flows through pure functions. Contrast with object oriented programming, where application state is usually shared and colocated with methods in objects.

Functional programming is a **programming paradigm**, meaning that it is a way of thinking about software construction based on some fundamental, defining principles (listed above). Other examples of programming paradigms include object oriented programming and procedural programming.

Link - <https://medium.com/javascript-scene/master-the-javascript-interview-what-is-functional-programming-7f218c68b3a0>

Q. What is pure function ?

Ans - A **pure function** is a function which:

* Given the same inputs, always returns the same output, and
* Has no side-effects - Pure functions are completely independent of outside state, and as such, they are immune to entire classes of bugs that have to do with shared mutable state. Their independent nature also makes them great candidates for parallel processing across many CPUs, and across entire distributed computing clusters, which makes them essential for many types of scientific and resource-intensive computing tasks.
* A pure function must not rely on any external mutable state, because it would no longer be deterministic or referentially transparent
* A pure function produces no side effects, which means that it can’t alter any external state.

Unpure function example - Math.random();