Question 1:
Prop drilling is when we need to pass state from the top level of the application through all the intermediary components to the one which needs the data at the bottom, even though the intermediary levels don’t need it.
For example, in the todolist app, if I want to change theme of the app, which will update the text color of the bottom child Todo, I need to pass the new text color through pros from App -> Home -> Todo. In this case, Home component doesn’t need to know this value. Which makes the process unnecessary. When the app has more complicated hierarchy structure, the disadvantage of props drilling is more obvious.

Question 2:
By using context to create provider-consumer pair relationships, we can avoid prop drilling.
The same example in the assignment, I created two theme providers that wrapped the App component. All the children components have direct access to the theme value by switching to this approach.

Question 3:
Using context is preferable for most cases, for example, when some data needs to be accessible by many components at different nesting levels, or a component nested deeply. However, if the the data are only required to pass down for 1-2 levels, it is not recommended to use context.

Question 4:
A giant component makes the code hard to read and maintains.
The advantage of breaking a monolith app into separate components are:

1. Increasing the code sharing and reusability
2. Easier to manage state
3. Easier to do unit test

Question 5:
When writing unit tests for React, shallow rendering lets us render a component independently, which means not to worry about the behavior of child components.
The advantage is easier to write unit tests, without considering about the connection between the component with its children.
The disadvantage is it cannot test the interaction amongst components

Question 6:
The statement is false. A unit test does not verify how the unit works with other parts of the application. For example, in todolist assignment , I wrote a unit test only test if a todo item’s check state can be updated correctly.

Question 7:

Snapshot is used to verify the changes in the code, while unit test verify if the unit works as expected.

Question 8:
The statement is right. For example, in the todolist app, the context is used to hold theme values which are acessed by multiple components like Home and its child Todo.