1.

Question 1

Imagine you have an array with one object that represents a dessert. You would like to apply some transformation to the item to output a different structure using the map function as per the code below. What would be the value of the newDesserts variable?

const desserts = [

{

title: 'Chocolate Cake',

description: 'Chocolate cake is a cake flavored with melted chocolate',

calories: 500,

}

];

const newDesserts = desserts.map((dessert) => {

return {

[

{

title: 'CHOCOLATE CAKE',

description: 'Chocolate cake is a cake flavored with melted chocolate',

kCal: 0.5,

}

]

[

{

title: 'Chocolate Cake',

description: 'Chocolate cake is a cake flavored with melted chocolate',

calories: 500,

kCal: 0.5,

}

]

[

{

title: 'CHOCOLATE CAKE',

description: 'Chocolate cake is a cake flavored with melted chocolate',

calories: 500,

kCal: 0.5,

}

]

2.

Question 2

How do you access dynamic data inside the JSX from the render function?

Using local state in the component.

Using component props.

Wrapping the variable in question with curly braces.

3.

Question 3

What could be a potential problem of using a randomiser function that generates an integer number from 0 to 10 as a key for your list items, having a list of only eight items? Select all that apply

The randomiser function does not entirely guarantee that the keys it generates will be different per item and a collision could happen, having two items with the same integer as keys.

There is no persistence of the keys generated since the moment the component re-renders the keys will vary and that could cause unexpected UI changes.

The randomiser function is a potential performance bottleneck since it has to run every re-render and it’s an unnecessary computation.

4.

Question 4

The todos array contains a list of todo objects, where each object has an id property that is unique. Which of the following code snippets will throw a React warning when opening up the browser console? Select all that apply

{todos.map((todo, index) => (

<ToDo id={todo.id} />

))}

{todos.map((todo, index) => (

<ToDo key={index} id={todo.id} />

))}

{todos.map((todo, index) => (

<ToDo key={todo.id} id={todo.id} />

))}

{todos.map((todo, index) => (

<ToDo key=”myKey” id={todo.id} />

))}

5.

Question 5

What are the potential problems of using indexes as keys?

The index is not persisted and will change the moment the component re-renders.

If the order of items may change, that can negatively impact performance and may cause issues with component state.

An index is not guaranteed to be unique.