Redux 1

1. Can you provide a brief summary of what is happening in this function code?

- This code defines a Redux reducer function called ‘countReducer’. A reducer in Redux is responsible for specifying how the application’s state should change in response to different actions. In this code, the reducer checks if ‘action.type’ is ‘increment’. If the ‘action.type’ is ‘increment’, the reducer returns a new state object. It takes the current ‘state.value’ and adds 1 to it. If the ‘action.type’ is not ‘increment’, the state does not change.

2. Add one action that tells the reducer to reduce the state value by 1

```
function countReducer(state = initialState, action) {
  switch (action.type) {
    case 'increment':
      return {
        value: state.value + 1
      };
    case 'decrement':
      return {
        value: state.value - 1
      };
    default:
      return state;
  }
}
```

3. Add one action that tells the reducer to reset the state

```
function countReducer(state = initialState, action) {
  switch (action.type) {
    case 'increment':
      return {
        value: state.value + 1
```

```
        };
      case 'decrement':
        return {
          value: state.value - 1
        };
      case 'reset':
        return initialState;
      default:
        return state;
    }
}
```

Redux 2

1. Can you provide a brief summary on what is happening on line 34, 39?

On line 2, the useState hook is being used. In React, the useState hook allows you to add state to functional components. It returns a pair of values, the current state value and a function to update that value. ‘studentsCount’ is set to 0, meaning that initially there are no students. When you call ‘setStudentsCount’ with a new value, React will update the state.

On line 7, a button element is created. ‘onClick’ specifies an event handler that will be executed when the button is clicked.

2. When a user clicks on the “Add student” button update the state (studentsCount) to include only the total number of students who are present.

a. Write a pseudocode of how your function would look.

```
Function addStudentOnClick:
  Increment studentsCount by 1
  Update the state with the new studentsCount value
Const students
  Array of students

Function classInfo:
```

Initialize studentsCount using useState with initial value 0

Function addStudent:

Increment studentsCount by 1

Update the state with the new studentsCount value

Render:

Display "Number of students in class room: " followed by studentsCount

Render a button with text "Add Student" and onClick event set to addStudent function

b. How do you ensure that the function is triggered when the button is clicked?

<button onClick={addStudent}>Add Student</button>

'addStudent' is the function from the pseudocode that would be executed on click.

c. How will you update the state with the result of your function?

```
const addStudent = () => {
    const presentStudentsCount = students.filter(student => student.present).length;
    setStudentsCount(presentStudentsCount);
};
```

The function filters the students array to count the students who have 'present' set to true. It then sets the students count to the count of the present students.

Redux 3

1. A change of code was made on line 174 (figure 4), can you briefly explain what that would do?

The 'action.payload' is a property of the 'action' object that carries additional data. It can be used to specify the amount that 'increment' increases the state by. When defining a function, like 'incrementAction', you would specify the 'type' as 'increment' and then 'payload' as whatever increment is necessary.

2. Let's say we don't want to set the state locally anymore and want to use dispatch. How would you ensure that an "increment" action that also contains the result of the studentCount is dispatched on button click? According to your answer in part 2.2b what would need to be changed?

Define the action type for the increment action;

Create a function that generates the ‘increment’ action with the payload (count);

Modify the ‘addStudent’ function to calculate the count of present students and dispatch the calculated count as the payload;

In the ‘countReducer’ function, handle the state based on the payload

3. Which code do you think is best suited to ensure that the “increment” action updates the state with the correct total number of students who are present. Is it Figure 4? Or Figure 5? Explain the code difference and your reasoning

Figure 5 – because it ensures that the state is updated correctly based on the action and payload. It is more reusable and keeps the reducer focused on its task of updating the state.