1.1

const Header = (*props*) =>{

  return (

    <div>

      <h1>

        {props.course}

      </h1>

    </div>

  )

}

const Content = (*props*) =>{

  return(

    <div>

      <p>

        {props.part} {props.exercise}

      </p>

    </div>

  )

}

const Total = (*props*) =>{

  return(

    <div>

      <p>

        Number of exercises {props.exercises}

      </p>

    </div>

  )

}

const App = () => {

  const course = 'Half Stack application development'

  const part1 = 'Fundamentals of React'

  const exercises1 = 10

  const part2 = 'Using props to pass data'

  const exercises2 = 7

  const part3 = 'State of a component'

  const exercises3 = 14

  return (

    <div>

      <Header *course* = {course}/>

      <Content *part* = {part1} *exercise* = {exercises1}/>

      <Content *part* = {part2} *exercise* = {exercises2}/>

      <Content *part* = {part3} *exercise* = {exercises3}/>

      <Total *exercises* = {exercises1 + exercises2 + exercises3}/>

    </div>

  )

}

export default App

1.2

const Header = (props) => {

return (

<div>

<h1>{props.course}</h1>

</div>

);

};

const Part = (props) => {

return (

<div>

<p>

{props.part} {props.exercises}

</p>

</div>

);

};

const Content = () => {

return (

<div>

<Part />

<Part />

<Part />

</div>

);

};

const Total = (props) => {

return (

<div>

<p>Number of exercises {props.exercises}</p>

</div>

);

};

const App = () => {

const course = "Half Stack application development";

const part1 = "Fundamentals of React";

const exercises1 = 10;

const part2 = "Using props to pass data";

const exercises2 = 7;

const part3 = "State of a component";

const exercises3 = 14;

return (

<div>

<Header course={course} />

<Part part={part1} exercises={exercises1} />

<Part part={part2} exercises={exercises2} />

<Part part={part3} exercises={exercises3} />

<Content />

<Total exercises={exercises1 + exercises2 + exercises3} />

</div>

);

};

export default App;

1.4

const Header = (props) => {

return (

<div>

<h1>{props.course}</h1>

</div>

);

};

const Part = (props) => {

return (

<div>

<p>

{props.part} {props.exercises}

</p>

</div>

);

};

const Content = () => {

return (

<div>

<Part />

<Part />

<Part />

</div>

);

};

const Total = (props) => {

return (

<div>

<p>Number of exercises {props.exercises}</p>

</div>

);

};

const App = () => {

const course = "Half Stack application development";

const parts = [

{

name: "Fundamentals of React",

exercises: 10

},

{

name: "Using props to pass data",

exercises: 7

},

{

name: "State of a component",

exercises: 14

}

];

return (

<div>

<Header course={course} />

<Part part={parts[0].name} exercises={parts[0].exercises} />

<Part part={parts[1].name} exercises={parts[1].exercises} />

<Part part={parts[2].name} exercises={parts[2].exercises} />

<Content />

<Total

exercises={parts[0].exercises + parts[1].exercises + parts[2].exercises}

/>

</div>

);

};

export default App;

1.5

const Header = (props) => {

return (

<div>

<h1>{props.course}</h1>

</div>

);

};

const Part = (props) => {

return (

<div>

<p>

{props.part} {props.exercises}

</p>

</div>

);

};

const Content = (props) => {

return (

<div>

<Part part={props.part[0].name} exercises={props.part[0].exercises} />

<Part part={props.part[1].name} exercises={props.part[1].exercises} />

<Part part={props.part[2].name} exercises={props.part[2].exercises} />

</div>

);

};

const Total = (props) => {

return (

<div>

<p>Number of exercises {props.exercises}</p>

</div>

);

};

const App = () => {

const course = {

name: "Half Stack application development",

parts: [

{

name: "Fundamentals of React",

exercises: 10

},

{

name: "Using props to pass data",

exercises: 7

},

{

name: "State of a component",

exercises: 14

}

]

};

return (

<div>

<Header course={course.name} />

<Content part={course.parts} />

<Total

exercises={

course.parts[0].exercises +

course.parts[1].exercises +

course.parts[2].exercises

}

/>

</div>

);

};

export default App;

1.6

import { useState } from "react"

const Stats = ({ good, neutral, bad }) => {

return (

<div>

<p>good: {good}</p>

<p>neutral: {neutral}</p>

<p>bad: {bad}</p>

</div>

)

}

const Button = ({ handleClick, text }) => (

<button onClick={handleClick}>{text}</button>

)

const Header = (props) => {

return (

<div>

<h1>{props.title}</h1>

</div>

)

}

const App = () => {

*// save clicks of each button to its own state*

const [good, setGood] = useState(0)

const [neutral, setNeutral] = useState(0)

const [bad, setBad] = useState(0)

const title = "give feedback"

const stats = "statistics"

const handleGoodClick = () => {

setGood(good + 1)

}

const handleNeutralClick = () => {

setNeutral(neutral + 1)

}

const handleBadClick = () => {

setBad(bad + 1)

}

return (

<div>

<Header title={title} />

<Button handleClick={handleGoodClick} text="good" />

<Button handleClick={handleNeutralClick} text="neutral" />

<Button handleClick={handleBadClick} text="bad" />

<Header title={stats} />

<Stats good={good} neutral={neutral} bad={bad} />

</div>

)

}

export default App

1.7/1.8

import { useState } from "react"

const Stat = ({ text, value }) => {

if (text === "positive") {

return (

<div>

{text} {value} %

</div>

)

}

return (

<div>

{text} {value}

</div>

)

}

const Statistics = ({ good, neutral, bad }) => {

const total = good + neutral + bad

const average = (good \* 1 + bad \* -1) / total

const positive = (good / total) \* 100

return (

<div>

<Stat text="good" value={good} />

<Stat text="neutral" value={neutral} />

<Stat text="bad" value={bad} />

<Stat text="total" value={total} />

<Stat text="average" value={average} />

<Stat text="positive" value={positive} />

</div>

)

}

const Button = ({ handleClick, text }) => (

<button onClick={handleClick}>{text}</button>

)

const Header = (props) => {

return (

<div>

<h1>{props.title}</h1>

</div>

)

}

const App = () => {

*// save clicks of each button to its own state*

const [good, setGood] = useState(0)

const [neutral, setNeutral] = useState(0)

const [bad, setBad] = useState(0)

const title = "give feedback"

const stats = "statistics"

const handleGoodClick = () => {

setGood(good + 1)

}

const handleNeutralClick = () => {

setNeutral(neutral + 1)

}

const handleBadClick = () => {

setBad(bad + 1)

}

return (

<div>

<Header title={title} />

<Button handleClick={handleGoodClick} text="good" />

<Button handleClick={handleNeutralClick} text="neutral" />

<Button handleClick={handleBadClick} text="bad" />

<Header title={stats} />

<Statistics good={good} neutral={neutral} bad={bad} />

</div>

)

}

export default App

1.9/1.10

import { useState } from "react"

const Stat = ({ text, value }) => {

if (text === "positive") {

return (

<div>

{text} {value} %

</div>

)

}

return (

<div>

{text} {value}

</div>

)

}

const Statistics = ({ good, neutral, bad }) => {

const total = good + neutral + bad

const average = (good \* 1 + bad \* -1) / total

const positive = (good / total) \* 100

if (total === 0) {

return <div>No feedback given</div>

}

return (

<div>

<Stat text="good" value={good} />

<Stat text="neutral" value={neutral} />

<Stat text="bad" value={bad} />

<Stat text="total" value={total} />

<Stat text="average" value={average} />

<Stat text="positive" value={positive} />

</div>

)

}

const Button = ({ handleClick, text }) => (

<button onClick={handleClick}>{text}</button>

)

const Header = (props) => {

return (

<div>

<h1>{props.title}</h1>

</div>

)

}

const App = () => {

*// save clicks of each button to its own state*

const [good, setGood] = useState(0)

const [neutral, setNeutral] = useState(0)

const [bad, setBad] = useState(0)

const title = "give feedback"

const stats = "statistics"

const handleGoodClick = () => {

setGood(good + 1)

}

const handleNeutralClick = () => {

setNeutral(neutral + 1)

}

const handleBadClick = () => {

setBad(bad + 1)

}

return (

<div>

<Header title={title} />

<Button handleClick={handleGoodClick} text="good" />

<Button handleClick={handleNeutralClick} text="neutral" />

<Button handleClick={handleBadClick} text="bad" />

<Header title={stats} />

<Statistics good={good} neutral={neutral} bad={bad} />

</div>

)

}

export default App

1.11

import { useState } from "react"

const Stat = ({ text, value }) => {

if (text === "positive") {

return (

<tr>

<td>{text}</td><td>{value}%</td>

</tr>

)

}

return (

<tr>

<td>{text}</td><td>{value}</td>

</tr>

)

}

const Statistics = ({ good, neutral, bad }) => {

const total = good + neutral + bad

const average = (good \* 1 + bad \* -1) / total

const positive = (good / total) \* 100

if (total === 0) {

return <div>No feedback given</div>

}

return (

<table>

<tbody>

<Stat text="good" value={good} />

<Stat text="neutral" value={neutral} />

<Stat text="bad" value={bad} />

<Stat text="total" value={total} />

<Stat text="average" value={average} />

<Stat text="positive" value={positive} />

</tbody>

</table>

)

}

const Button = ({ handleClick, text }) => (

<button onClick={handleClick}>{text}</button>

)

const Header = (props) => {

return (

<div>

<h1>{props.title}</h1>

</div>

)

}

const App = () => {

*// save clicks of each button to its own state*

const [good, setGood] = useState(0)

const [neutral, setNeutral] = useState(0)

const [bad, setBad] = useState(0)

const title = "give feedback"

const stats = "statistics"

const handleGoodClick = () => {

setGood(good + 1)

}

const handleNeutralClick = () => {

setNeutral(neutral + 1)

}

const handleBadClick = () => {

setBad(bad + 1)

}

return (

<div>

<Header title={title} />

<Button handleClick={handleGoodClick} text="good" />

<Button handleClick={handleNeutralClick} text="neutral" />

<Button handleClick={handleBadClick} text="bad" />

<Header title={stats} />

<Statistics good={good} neutral={neutral} bad={bad} />

</div>

)

}

export default App

1.12

import { useState } from "react"

const Button = ({ handleClick, text }) => {

return (

<div>

<button onClick={handleClick}>{text}</button>

</div>

)

}

const App = () => {

const anecdotes = [

"If it hurts, do it more often.",

"Adding manpower to a late software project makes it later!",

"The first 90 percent of the code accounts for the first 10 percent of the development time...The remaining 10 percent of the code accounts for the other 90 percent of the development time.",

"Any fool can write code that a computer can understand. Good programmers write code that humans can understand.",

"Premature optimization is the root of all evil.",

"Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.",

"Programming without an extremely heavy use of console.log is same as if a doctor would refuse to use x-rays or blood tests when diagnosing patients.",

"The only way to go fast, is to go well."

]

const [selected, setSelected] = useState(0)

const handleSelectedClick = () => {

setSelected(Math.floor(Math.random() \* anecdotes.length))

}

return (

<div>

{anecdotes[selected]}

<Button handleClick={handleSelectedClick} text="next anecdote" />

</div>

)

}

export default App

1.13

import { useState } from "react"

const Anecdote = ({ text, count }) => {

return (

<div>

<div>

{text}

</div>

<div>

has {count} votes

</div>

</div>

)

}

const Header = (props) => {

return (

<div>

<h1>{props.title}</h1>

</div>

)

}

const Button = ({ handleClick, text }) => {

return (

<div>

<button onClick={handleClick}>{text}</button>

</div>

)

}

const App = () => {

const anecdotes = [

"If it hurts, do it more often.",

"Adding manpower to a late software project makes it later!",

"The first 90 percent of the code accounts for the first 10 percent of the development time...The remaining 10 percent of the code accounts for the other 90 percent of the development time.",

"Any fool can write code that a computer can understand. Good programmers write code that humans can understand.",

"Premature optimization is the root of all evil.",

"Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.",

"Programming without an extremely heavy use of console.log is same as if a doctor would refuse to use x-rays or blood tests when diagnosing patients.",

"The only way to go fast, is to go well."

]

const [selected, setSelected] = useState(0)

const [vote, setVote] = useState(Array(anecdotes.length).fill(0))

const handleSelectedClick = () => {

const arrayIndex = Math.floor(Math.random() \* anecdotes.length)

setSelected(arrayIndex)

}

const handleVoteClick = () => {

const points = [...vote]

points[selected] += 1

*//console.log(points[selected])*

setVote(points)

}

return (

<div>

<Anecdote text={anecdotes[selected]} count={vote[selected]}/>

<Button handleClick={handleSelectedClick} text="next anecdote" />

<Button handleClick={handleVoteClick} text="vote" />

</div>

)

}

export default App

1.14

import { useState } from "react"

const Winner = ({ text, vote }) => {

const index = vote.indexOf(Math.max(...vote))

return (

<div>

<Header title="Anecdote with most votes" />

<Anecdote text={text[index]} count={vote[index]} />

</div>

)

}

const Anecdote = ({ text, count }) => {

return (

<div>

<div>{text}</div>

<div>has {count} votes</div>

</div>

)

}

const Header = (props) => {

return (

<div>

<h1>{props.title}</h1>

</div>

)

}

const Button = ({ handleClick, text }) => (

<button onClick={handleClick}>{text}</button>

)

const App = () => {

const anecdotes = [

"If it hurts, do it more often.",

"Adding manpower to a late software project makes it later!",

"The first 90 percent of the code accounts for the first 10 percent of the development time...The remaining 10 percent of the code accounts for the other 90 percent of the development time.",

"Any fool can write code that a computer can understand. Good programmers write code that humans can understand.",

"Premature optimization is the root of all evil.",

"Debugging is twice as hard as writing the code in the first place. Therefore, if you write the code as cleverly as possible, you are, by definition, not smart enough to debug it.",

"Programming without an extremely heavy use of console.log is same as if a doctor would refuse to use x-rays or blood tests when diagnosing patients.",

"The only way to go fast, is to go well."

]

const [selected, setSelected] = useState(0)

const [vote, setVote] = useState(Array(anecdotes.length).fill(0))

const handleSelectedClick = () => {

const arrayIndex = Math.floor(Math.random() \* anecdotes.length)

setSelected(arrayIndex)

}

const handleVoteClick = () => {

const points = [...vote]

points[selected] += 1

*//console.log(points[selected])*

setVote(points)

}

return (

<div>

<Header title="Anecdote of the day" />

<Anecdote text={anecdotes[selected]} count={vote[selected]} />

<Button handleClick={handleVoteClick} text="vote" />

<Button handleClick={handleSelectedClick} text="next anecdote" />

<Winner text={anecdotes} vote={vote} />

</div>

)

}

export default App

2.1

const Header = (props) => {

return (

<div>

<h1>{props.course}</h1>

</div>

)

}

const Content = ({parts}) => {

return (

<div>

{parts.map((part, i) =>

<Part key={i} part={part.name} exercises={part.exercises} />

)}

</div>

)

}

const Part = ({part, exercises}) => {

return (

<p>

{part} {exercises}

</p>

)

}

const Course = ({course}) =>{

return(

<div>

<div key = {course.id}>

<Header course={course.name} />

<Content parts={course.parts} />

</div>

</div>

)

}

const App = () => {

const course = {

id: 1,

name: 'Half Stack application development',

parts: [

{

name: 'Fundamentals of React',

exercises: 10,

id: 1

},

{

name: 'Using props to pass data',

exercises: 7,

id: 2

},

{

name: 'State of a component',

exercises: 14,

id: 3

}

]

}

return (

<div>

<Course course={course} />

</div>

)

}

export default App

2.2/2.3

const Header = (props) => {

return (

<div>

<h1>{props.course}</h1>

</div>

)

}

const Total = ({parts}) =>{

const total = parts.reduce((s,p) => s + p.exercises,0)

*//s = current value, p = current index, 0 = initial value*

return(

<div>

<b>total of {total} exercises</b>

</div>

)

}

const Part = ({ part, exercises }) => {

return (

<p>

{part} {exercises}

</p>

)

}

const Content = ({ parts }) => {

return (

<div>

{parts.map((part, i) => (

<Part key={i} part={part.name} exercises={part.exercises} />

))}

</div>

)

}

const Course = ({ course }) => {

return (

<div>

<div key={course.id}>

<Header course={course.name} />

<Content parts={course.parts} />

<Total parts = {course.parts}/>

</div>

</div>

)

}

const App = () => {

const course = {

id: 1,

name: "Half Stack application development",

parts: [

{

name: "Fundamentals of React",

exercises: 10,

id: 1

},

{

name: "Using props to pass data",

exercises: 7,

id: 2

},

{

name: "State of a component",

exercises: 14,

id: 3

},

{

name: 'Redux',

exercises: 11,

id: 4

}

]

}

return (

<div>

<Course course={course} />

</div>

)

}

export default App

2.4

const Title = ({title}) =>{

return(

<h2>{title}</h2>

)

}

const Header = (props) => {

return (

<div>

<h3>{props.course}</h3>

</div>

)

}

const Total = ({parts}) =>{

const total = parts.reduce((s,p) => s + p.exercises,0)

*//s = current value, p = current index, 0 = initial value*

return(

<div>

<b>total of {total} exercises</b>

</div>

)

}

const Part = ({ part, exercises }) => {

return (

<p>

{part} {exercises}

</p>

)

}

const Content = ({ parts }) => {

return (

<div>

{parts.map((part, i) => (

<Part key={i} part={part.name} exercises={part.exercises} />

))}

</div>

)

}

const Course = ({ course }) => {

return (

<div>

{course.map(courses =>

<div key={courses.id}>

<Header course={courses.name} />

<Content parts={courses.parts} />

<Total parts = {courses.parts}/>

</div>

)}

</div>

)

}

const App = () => {

const courses = [

{

name: 'Half Stack application development',

id: 1,

parts: [

{

name: 'Fundamentals of React',

exercises: 10,

id: 1

},

{

name: 'Using props to pass data',

exercises: 7,

id: 2

},

{

name: 'State of a component',

exercises: 14,

id: 3

},

{

name: 'Redux',

exercises: 11,

id: 4

}

]

},

{

name: 'Node.js',

id: 2,

parts: [

{

name: 'Routing',

exercises: 3,

id: 1

},

{

name: 'Middlewares',

exercises: 7,

id: 2

}

]

}

]

return (

<div>

<Title title = "Web development curriculum"/>

<Course course={courses} />

</div>

)

}

export default App

2.5

Course.js

const Header = (props) => {

return (

<div>

<h3>{props.course}</h3>

</div>

)

}

const Total = ({ parts }) => {

const total = parts.reduce((s, p) => s + p.exercises, 0)

*//s = current value, p = current index, 0 = initial value*

return (

<div>

<b>total of {total} exercises</b>

</div>

)

}

const Part = ({ part, exercises }) => {

return (

<p>

{part} {exercises}

</p>

)

}

const Content = ({ parts }) => {

return (

<div>

{parts.map((part, i) => (

<Part key={i} part={part.name} exercises={part.exercises} />

))}

</div>

)

}

const Course = ({ course }) => {

return (

<div>

{course.map((courses) => (

<div key={courses.id}>

<Header course={courses.name} />

<Content parts={courses.parts} />

<Total parts={courses.parts} />

</div>

))}

</div>

)

}

export default Course

App.js

import Course from "./Course"

const Title = ({ title }) => {

return <h2>{title}</h2>

}

const App = () => {

const courses = [

{

name: "Half Stack application development",

id: 1,

parts: [

{

name: "Fundamentals of React",

exercises: 10,

id: 1

},

{

name: "Using props to pass data",

exercises: 7,

id: 2

},

{

name: "State of a component",

exercises: 14,

id: 3

},

{

name: "Redux",

exercises: 11,

id: 4

}

]

},

{

name: "Node.js",

id: 2,

parts: [

{

name: "Routing",

exercises: 3,

id: 1

},

{

name: "Middlewares",

exercises: 7,

id: 2

}

]

}

]

return (

<div>

<Title title="Web development curriculum" />

<Course course={courses} />

</div>

)

}

export default App

2.6

import { useState } from "react"

const Header = ({ title }) => {

return <h2>{title}</h2>

}

const Names = ({ persons }) => {

return (

<div>

{persons.map((person, i) => (

<div key={i}>{persons[i].name}</div>

))}

</div>

)

}

const App = () => {

const [persons, setPersons] = useState([{ name: "Arto Hellas" }])

const [newName, setNewName] = useState("")

const addPerson = (event) => {

event.preventDefault() *//prevents submitting form*

const personObject = {

name: newName

}

setPersons(persons.concat(personObject))

setNewName("")

}

const handlePersonChange = (event) => {

setNewName(event.target.value)

}

return (

<div>

<Header title="Phonebook" />

<form onSubmit={addPerson}>

<div>

name: <input value={newName} onChange={handlePersonChange} />

</div>

<div>

<button type="submit">add</button>

</div>

</form>

<Header title="Numbers" />

<Names persons={persons} />

</div>

)

}

export default App

2.7

import { useState } from "react"

const Header = ({ title }) => {

return <h2>{title}</h2>

}

const Names = ({ persons }) => {

return (

<div>

{persons.map((person, i) => (

<div key={i}>{persons[i].name}</div>

))}

</div>

)

}

const App = () => {

const [persons, setPersons] = useState([{ name: "Arto Hellas" }])

const [newName, setNewName] = useState("")

const addPerson = (event) => {

event.preventDefault() *//prevents submitting form*

const personObject = {

name: newName

}

setPersons(persons.concat(personObject))

setNewName("")

}

const handlePersonChange = (event) => {

const inputValue = event.target.value

if (persons.some((person) => person.name === inputValue)) {

alert(`${inputValue} is already added to the phonebook`)

} else {

setNewName(inputValue)

}

}

return (

<div>

<Header title="Phonebook" />

<form onSubmit={addPerson}>

<div>

name: <input value={newName} onChange={handlePersonChange} />

</div>

<div>

<button type="submit">add</button>

</div>

</form>

<Header title="Numbers" />

<Names persons={persons} />

</div>

)

}

export default App

2.8

import { useState } from "react"

const Header = ({ title }) => {

return <h2>{title}</h2>

}

const Numbers = ({ persons }) => {

return (

<div>

{persons.map((person, i) => (

<div key={i}>

{persons[i].name} {persons[i].number}

</div>

))}

</div>

)

}

const PersonForm = ({

onSubmit,

newName,

handlePersonChange,

newNum,

handleNumberChange

}) => {

return (

<form onSubmit={onSubmit}>

<div>

name: <input value={newName} onChange={handlePersonChange} />

</div>

<div>

number: <input value={newNum} onChange={handleNumberChange} />

</div>

<div>

<button type="submit">add</button>

</div>

</form>

)

}

const App = () => {

const [persons, setPersons] = useState([])

const [newName, setNewName] = useState("")

const [newNum, setNewNum] = useState("")

const addPerson = (event) => {

event.preventDefault() *//prevents submitting form*

if (newName.length === 0 || newNum.length === 0) {

alert("Must enter both name and number")

} else {

const personObject = {

name: newName,

number: newNum

}

setPersons(persons.concat(personObject))

setNewName("")

setNewNum("")

}

}

const handlePersonChange = (event) => {

const inputValue = event.target.value

if (persons.some((person) => person.name === inputValue)) {

alert(`${inputValue} is already added to the phonebook`)

} else {

setNewName(inputValue)

}

}

const handleNumberChange = (event) => {

setNewNum(event.target.value)

}

return (

<div>

<Header title="Phonebook" />

<PersonForm

onSubmit={addPerson}

newName={newName}

handlePersonChange={handlePersonChange}

newNum={newNum}

handleNumberChange={handleNumberChange}

/>

<Header title="Numbers" />

<Numbers persons={persons} />

</div>

)

}

export default App

2.9

import { useState } from "react"

const Header = ({ title }) => {

return <h2>{title}</h2>

}

const People = ({ persons }) => {

return (

<div>

{persons.map((person, i) => (

<div key={i}>

{persons[i].name} {persons[i].number}

</div>

))}

</div>

)

}

const PersonForm = ({

onSubmit,

newName,

handlePersonChange,

newNum,

handleNumberChange

}) => {

return (

<form onSubmit={onSubmit}>

<div>

name: <input value={newName} onChange={handlePersonChange} />

</div>

<div>

number: <input value={newNum} onChange={handleNumberChange} />

</div>

<div>

<button type="submit">add</button>

</div>

</form>

)

}

const SetFilter = ({ newFilter, handleFilterChange, showAll, setShowAll }) => {

return (

<div>

<div>

filter shown with:

<input value={newFilter} onChange={handleFilterChange} />

</div>

</div>

)

}

const App = () => {

const [persons, setPersons] = useState([

{ name: "Arto Hellas", number: "040-123456", id: 1 },

{ name: "Ada Lovelace", number: "39-44-5323523", id: 2 },

{ name: "Dan Abramov", number: "12-43-234345", id: 3 },

{ name: "Mary Poppendieck", number: "39-23-6423122", id: 4 }

])

const [newName, setNewName] = useState("")

const [newNum, setNewNum] = useState("")

const [showAll, setShowAll] = useState("")

const [newFilter, setNewFilter] = useState("")

const nameToShow = showAll

? persons

: persons.filter((persons) =>

persons.name.toLowerCase().match(newFilter.toLowerCase())

)

const addPerson = (event) => {

event.preventDefault() *//prevents submitting form*

if (newName.length === 0 || newNum.length === 0) {

alert("Must enter both name and number")

} else {

const personObject = {

name: newName,

number: newNum

}

setPersons(persons.concat(personObject))

setNewName("")

setNewNum("")

}

}

const handlePersonChange = (event) => {

const inputValue = event.target.value

if (persons.some((person) => person.name === inputValue)) {

alert(`${inputValue} is already added to the phonebook`)

} else {

setNewName(inputValue)

}

}

const handleNumberChange = (event) => {

setNewNum(event.target.value)

}

const handleFilterChange = (event) => {

const newFilter = event.target.value

setNewFilter(newFilter)

setShowAll(showAll)

}

return (

<div>

<Header title="Phonebook" />

<SetFilter

newFilter={newFilter}

handleFilterChange={handleFilterChange}

showAll={showAll}

setShowAll={setShowAll}

/>

<Header title="add a new" />

<PersonForm

onSubmit={addPerson}

newName={newName}

handlePersonChange={handlePersonChange}

newNum={newNum}

handleNumberChange={handleNumberChange}

/>

<Header title="Numbers" />

<People persons={nameToShow} />

</div>

)

}

export default App