Задачи про формы

1. Сделать компонент, отображающий селектбокс с городами. Если выбран не Рио, писать снизу: "Нет, это не Рио-де-Жанейро!"

```
import React, { useState } from 'react';

const CitySelector = () => {
  const [selectedCity, setSelectedCity] = useState('');

  const handleCityChange = (event) => {
    setSelectedCity(event.target.value);
  };

  return (
    <div>
      <h2>Выберите город:</h2>
      <select value={selectedCity} onChange={handleCityChange}>
        <option value="">Выберите город</option>
        <option value="rio">Рио-де-Жанейро</option>
        <option value="paris">Париж</option>
        <option value="london">Лондон</option>
        <option value="tokyo">Токио</option>
      </select>
      {selectedCity !== 'rio' && (
        <p>Нет, это не Рио-де-Жанейро!</p>
      )}
    </div>
  );
};

export default CitySelector;
```

2. Сделать калькулятор с двумя полями ввода чисел, полем выбора действия и выводом результата текстом рядом: 2 + 2 = 4

```
import React, { useState } from 'react';

const Calculator = () => {
  const [num1, setNum1] = useState('');
  const [num2, setNum2] = useState('');
  const [operator, setOperator] = useState('+');
  const [result, setResult] = useState('');

  const handleNum1Change = (event) => {
    setNum1(event.target.value);
  };

  const handleNum2Change = (event) => {
    setNum2(event.target.value);
  };

  const handleOperatorChange = (event) => {
    setOperator(event.target.value);
  };

  const calculateResult = () => {
    const number1 = parseFloat(num1);
    const number2 = parseFloat(num2);

    switch (operator) {
      case '+':
        setResult(number1 + number2);
        break;
      case '-':
        setResult(number1 - number2);
        break;
      case '*':
        setResult(number1 * number2);
        break;
      case '/':
        setResult(number1 / number2);
        break;
      default:
        setResult('');
    }
  };

  return (
    <div>
      <h2>Калькулятор</h2>
      <input type="number" value={num1} onChange={handleNum1Change} />
      <select value={operator} onChange={handleOperatorChange}>
        <option value="+">+</option>
        <option value="-">-</option>
        <option value="*">*</option>
        <option value="/">/</option>
      </select>
      <input type="number" value={num2} onChange={handleNum2Change} />
      <button onClick={calculateResult}>=</button>
      <p>Результат: {result}</p>
    </div>
  );
};

export default Calculator;
```

3. Калькулятор систем счисления:

15 в 10-чной системе. 1111 в 2 системе.

Исходное число вводится в поле ввода, система счисления выбирается из предложенных, результат отображается во втором поле

```
import React, { useState } from 'react';

const baseOptions = [
  { label: 'Десятичная (10)', value: 10 },
  { label: 'Двоичная (2)', value: 2 },
  { label: 'Шестнадцатеричная (16)', value: 16 },
];

const Converter = () => {
  const [inputNumber, setInputNumber] = useState('');
  const [inputBase, setInputBase] = useState(10);
  const [outputBase, setOutputBase] = useState(2);
  const [outputNumber, setOutputNumber] = useState('');

  const handleChangeInputNumber = (event) => {
    setInputNumber(event.target.value);
  };

  const handleChangeInputBase = (event) => {
    setInputBase(Number(event.target.value));
  };

  const handleChangeOutputBase = (event) => {
    setOutputBase(Number(event.target.value));
  };

  const convertNumber = () => {
    if (!inputNumber) {
      setOutputNumber('');
      return;
    }

    const parsedInputNumber = parseInt(inputNumber, inputBase);
    const convertedOutputNumber = parsedInputNumber.toString(outputBase);
    setOutputNumber(convertedOutputNumber.toUpperCase());
  };

  return (
    <div>
      <h2>Конвертер систем счисления</h2>
      <input
        type="text"
        value={inputNumber}
        onChange={handleChangeInputNumber}
        placeholder="Введите число"
      />
      <select value={inputBase} onChange={handleChangeInputBase}>
        {baseOptions.map(option => (
          <option key={option.value} value={option.value}>{option.label}</option>
        ))}
      </select>
      <span>в</span>
      <select value={outputBase} onChange={handleChangeOutputBase}>
        {baseOptions.map(option => (
          <option key={option.value} value={option.value}>{option.label}</option>
        ))}
      </select>
      <button onClick={convertNumber}>Конвертировать</button>
      <input type="text" value={outputNumber} readOnly />
    </div>
  );
};

export default Converter;
```

4. В поле ввода выбираем дату рождения: дд.мм.гггг

Ниже выводится текстом: "Вы прожили: 1232352345234 секунд."

Количество секунд увеличивается каждую секунду.

```
import React, { useState, useEffect } from 'react';

const AgeCalculator = () => {
  const [birthdate, setBirthdate] = useState('');
  const [livedSeconds, setLivedSeconds] = useState(0);

  useEffect(() => {
    const interval = setInterval(() => {
      if (birthdate) {
        const birthdateTimestamp = new Date(birthdate).getTime();
        const currentTimestamp = Date.now();
        const secondsDifference = Math.floor((currentTimestamp - birthdateTimestamp) / 1000);
        setLivedSeconds(secondsDifference);
      }
    }, 1000);

    return () => clearInterval(interval);
  }, [birthdate]);

  const handleInputChange = (event) => {
    setBirthdate(event.target.value);
  };

  return (
    <div>
      <h2>Калькулятор возраста</h2>
      <input
        type="text"
        value={birthdate}
        onChange={handleInputChange}
        placeholder="Введите дату рождения: дд.мм.гггг"
      />
      <p>Вы прожили: {livedSeconds} секунд.</p>
    </div>
  );
};

export default AgeCalculator;
```

5. Список чисел с фильтрацией: [поле ввода числа] [+] Все Четные Нечетные

1, 13, 6, 52, 4, 14

При нажатии на [+] в список добавляется очередное число из поля ввода. При изменении состояния

фильтра список обновляется.

```
import React, { useState } from 'react';

const NumberList = () => {
  const [inputNumber, setInputNumber] = useState('');
  const [filter, setFilter] = useState('All');
  const [numbers, setNumbers] = useState([1, 13, 6, 52, 4, 14]);

  const handleInputChange = (event) => {
    setInputNumber(event.target.value);
  };

  const handleAddNumber = () => {
    if (inputNumber !== '') {
      setNumbers(prevNumbers => [...prevNumbers, parseInt(inputNumber)]);
      setInputNumber('');
    }
  };

  const handleFilterChange = (event) => {
    setFilter(event.target.value);
  };

  const filteredNumbers = numbers.filter(number => {
    if (filter === 'All') return true;
    if (filter === 'Even') return number % 2 === 0;
    if (filter === 'Odd') return number % 2 !== 0;
    return true;
  });

  return (
    <div>
      <h2>Список чисел с фильтрацией</h2>
      <div>
        <input
          type="text"
          value={inputNumber}
          onChange={handleInputChange}
          placeholder="Введите число"
        />
        <button onClick={handleAddNumber}>+</button>
      </div>
      <div>
        Фильтр:
        <select value={filter} onChange={handleFilterChange}>
          <option value="All">Все</option>
          <option value="Even">Четные</option>
          <option value="Odd">Нечетные</option>
        </select>
      </div>
      <ul>
        {filteredNumbers.map((number, index) => (
          <li key={index}>{number}</li>
        ))}
      </ul>
    </div>
  );
};

export default NumberList;
```

Задачи про валидацию

1. Написать валидацию формы регистрации:

- Поле "Логин":
  - Обязательное.
  - Содержит от 6 до 20 символов.
  - Туда можно вводить буквы латинского алфавита и цифры.
- Поле "Пароль":
  - Обязательное.
- Поле "Повтор пароля":
  - Обязательное.
  - Должно совпадать с паролем.

```
import React, { useState } from 'react';

const RegistrationForm = () => {
  const [formData, setFormData] = useState({
    username: '',
    password: '',
    confirmPassword: ''
  });

  const [errors, setErrors] = useState({});

  const handleChange = (event) => {
    const { name, value } = event.target;
    setFormData({
      ...formData,
      [name]: value
    });
  };

  const handleSubmit = (event) => {
    event.preventDefault();
    const validationErrors = {};

    // Валидация логина
    if (!formData.username.trim()) {
      validationErrors.username = 'Логин обязателен';
    } else if (formData.username.length < 6 || formData.username.length > 20) {
      validationErrors.username = 'Логин должен содержать от 6 до 20 символов';
    } else if (!/^[a-zA-Z0-9]+$/.test(formData.username)) {
      validationErrors.username = 'Логин может содержать только латинские буквы и цифры';
    }

    // Валидация пароля
    if (!formData.password.trim()) {
      validationErrors.password = 'Пароль обязателен';
    }

    // Валидация повтора пароля
    if (!formData.confirmPassword.trim()) {
      validationErrors.confirmPassword = 'Подтверждение пароля обязательно';
    } else if (formData.confirmPassword !== formData.password) {
      validationErrors.confirmPassword = 'Пароли не совпадают';
    }

    setErrors(validationErrors);
```

```
    if (Object.keys(validationErrors).length === 0) {
      // Здесь можно отправить данные на сервер или выполнить другие действия
      console.log('Данные формы валидны:', formData);
    }
  };

  return (
    <div>
      <h2>Форма регистрации</h2>
      <form onSubmit={handleSubmit}>
        <div>
          <label>Логин:</label>
          <input
            type="text"
            name="username"
            value={formData.username}
            onChange={handleChange}
          />
          {errors.username && <span style={{ color: 'red' }}>{errors.username}</span>}
        </div>
        <div>
          <label>Пароль:</label>
          <input
            type="password"
            name="password"
            value={formData.password}
            onChange={handleChange}
          />
          {errors.password && <span style={{ color: 'red' }}>{errors.password}</span>}
        </div>
        <div>
          <label>Повтор пароля:</label>
          <input
            type="password"
            name="confirmPassword"
            value={formData.confirmPassword}
            onChange={handleChange}
          />
          {errors.confirmPassword && <span style={{ color: 'red' }}>{errors.confirmPassword}</span>}
        </div>
        <button type="submit">Зарегистрироваться</button>
      </form>
    </div>
  );
};

export default RegistrationForm;
```

2. Написать валидацию формы редактирования профиля:

- Имя (обязательно).
- Отчество (обязательно).
- Фамилия (обязательно).
- Дата рождения (ДД.ММ.ГГГГ, опционально).
- Адрес (опционально).

```jsx
import React, { useState } from 'react';

const ProfileEditForm = () => {
  const [formData, setFormData] = useState({
    firstName: '',
    middleName: '',
    lastName: ''
  });

  const [errors, setErrors] = useState({});

  const handleChange = (event) => {
    const { name, value } = event.target;
    setFormData({
      ...formData,
      [name]: value
    });
  };

  const handleSubmit = (event) => {
    event.preventDefault();
    const validationErrors = {};

    // Валидация имени
    if (!formData.firstName.trim()) {
      validationErrors.firstName = 'Имя обязательно';
    }

    // Валидация отчества
    if (!formData.middleName.trim()) {
      validationErrors.middleName = 'Отчество обязательно';
    }

    // Валидация фамилии
    if (!formData.lastName.trim()) {
      validationErrors.lastName = 'Фамилия обязательна';
    }

    setErrors(validationErrors);

    if (Object.keys(validationErrors).length === 0) {
      // Здесь можно отправить данные на сервер или выполнить другие действия
      console.log('Данные формы валидны:', formData);
    }
  };
```

```
  return (
    <div>
      <h2>Форма редактирования профиля</h2>
      <form onSubmit={handleSubmit}>
        <div>
          <label>Имя:</label>
          <input
            type="text"
            name="firstName"
            value={formData.firstName}
            onChange={handleChange}
          />
          {errors.firstName && <span style={{ color: 'red' }}>{errors.firstName}</span>}
        </div>
        <div>
          <label>Отчество:</label>
          <input
            type="text"
            name="middleName"
            value={formData.middleName}
            onChange={handleChange}
          />
          {errors.middleName && <span style={{ color: 'red' }}>{errors.middleName}</span>}
        </div>
        <div>
          <label>Фамилия:</label>
          <input
            type="text"
            name="lastName"
            value={formData.lastName}
            onChange={handleChange}
          />
          {errors.lastName && <span style={{ color: 'red' }}>{errors.lastName}</span>}
        </div>
        <button type="submit">Сохранить изменения</button>
      </form>
    </div>
  );
};

export default ProfileEditForm;
```