1) **Dynamic Registration Form with Validation Enable/Disable Create a registration form with fields: Name, Email, Password.**
   • Use JavaScript to enable the Submit button only if all fields are non-empty and valid.
   • Show error messages if fields are invalid when the user moves away from an input field.
   • Use regular expressions to validate the email format and password strength (minimum 8 characters, one uppercase, one number).

   Source code:

```
<html>
<head>
 <title>question 1</title>
 <style>
  body { font-family: Arial; margin: 30px; }
  input { display:block; margin:8px 0; padding:5px; }
  small { color:red; }
  button { padding:5px 10px; }
  button:disabled { background:#ccc; }
 </style>
</head>
<body>
 <h2>Registration</h2>
 <form id="regForm">
  <input type="text" id="name" placeholder="Name">
  <small id="nameErr"></small>

  <input type="text" id="email" placeholder="Email">
  <small id="emailErr"></small>

  <input type="password" id="pass" placeholder="Password">
  <small id="passErr"></small>

  <button type="submit" id="submitBtn" disabled>Submit</button>
 </form>

<script>
let name=document.getElementById("name");
let email=document.getElementById("email");
```

```
let pass=document.getElementById("pass");
let btn=document.getElementById("submitBtn");

function validate(){
  let nameOk=name.value.trim()!=="";
  let emailOk=/^[^@\s]+@[^@\s]+\.[^@\s]+$/.test(email.value);
  let passOk=/^(?=.*[A-Z])(?=.*\d).{8,}$/.test(pass.value);
  btn.disabled=!(nameOk&&emailOk&&passOk);
}

name.onblur=()=> {

document.getElementById("nameErr").textContent=name.value?"":"Na
me required";
  validate();
}
email.onblur=()=> {

document.getElementById("emailErr").textContent=/^[^@\s]+@[^@\s]+
\.[^@\s]+$/.test(email.value)?"":"Invalid email";
  validate();
}
pass.onblur=()=> {
 document.getElementById("passErr").textContent=/^(?=.*[A-
Z])(?=.*\d).{8,}$/.test(pass.value)?"":"Weak password";
  validate();
}
</script>
</body>
</html>
```

2) **Email Field Regex Validation on Blur Event Design a form with an email field that performs real-time validation with JavaScript regex: /^[^@\\s]+@[^@\\s]+\\.[^@\\s]+$/.**
   • Trigger validation on the onblur event.
   • Display a success or error message dynamically below the input field.

Souce code:

```
<html>
<head>
 <title>question 2</title>
 <style>
  body { font-family: Arial; margin: 30px; }
  input { padding: 5px; margin-bottom: 5px; }
  small { display:block; margin-top:3px; }
  .ok { color:green; }
  .err { color:red; }
 </style>
</head>
<body>
 <h3>Email Check</h3>
 <input type="text" id="email" placeholder="Enter Email">
 <small id="msg"></small>
<script>
let email=document.getElementById("email");
let msg=document.getElementById("msg");
let regex=/^[^@\s]+@[^@\s]+\.[^@\s]+$/;

email.onblur=()=>{
 if(regex.test(email.value)){
  msg.textContent="Valid Email ✓";
  msg.className="ok";
 }else{
  msg.textContent="Invalid Email ✗";
  msg.className="err";
 }
}
</script>
</body>
</html>
```

3) **Dynamic Phone Number Validation Using Regular Expressions**
Create a phone number input field and validate user input as they type using regex to allow only valid phone numbers like:
- **US format: (123) 456-7890 or 123-456-7890.**
- **Use JavaScript input event to update validation status live.**

Source Code:

```
<html>
<head>
 <title>question 3</title>
 <style>
  body { font-family: Arial; margin:30px; }
  input { padding:5px; }
  small { display:block; margin-top:3px; }
  .ok { color:green; }
  .err { color:red; }
 </style>
</head>
<body>
 <h3>Phone Number (123)</h3>
 <input type="text" id="phone" placeholder="Enter phone">
 <small id="msg"></small>

<script>
let phone=document.getElementById("phone");
let msg=document.getElementById("msg");
let regex=/^(\(\d{3}\)\s?\d{3}-\d{4}|\d{3}-\d{3}-\d{4})$/;
phone.oninput=()=>{
 if(regex.test(phone.value)){
  msg.textContent="Valid Phone ✓";
  msg.className="ok";
 }else{
  msg.textContent="Invalid Format ✗";
  msg.className="err";
 }
}
</script>
</body>
</html>
```

4) **Dynamic Form Input Fields Addition Build a "hobbies" form that allows users to add multiple hobbies dynamically using a button.**
   • When the “Add Hobby” button is clicked, append a new input field below the existing ones.
   • Validate each hobby input to contain only alphabets using regex /^[A-Za-z\\s]+$/.

Source Code:

```
<html>
<head>
 <title>question 4</title>
 <style>
  body { font-family: Arial; margin:30px; }
  input { display:block; margin:5px 0; padding:5px; }
  small { color:red; display:block; margin-bottom:5px; }
  button { margin-top:5px; padding:5px 10px; }
 </style>
</head>
<body>
 <h3>Hobby Append</h3>
 <form id="hobbyForm">
  <div id="hobbyList">
   <input type="text" placeholder="Enter hobby">
   <small></small>
  </div>
  <button type="button" id="addBtn">Add Hobby</button>
 </form>

<script>
let addBtn=document.getElementById("addBtn");
let list=document.getElementById("hobbyList");
let regex=/^[A-Za-z\s]+$/;

function validate(input, msg){
 msg.textContent=regex.test(input.value)?"":"Only letters allowed";
}

addBtn.onclick=()=>{
 let div=document.createElement("div");
```

```
  let inp=document.createElement("input");
  let small=document.createElement("small");
  inp.placeholder="Enter hobby";
  inp.onblur=()=>validate(inp,small);
  div.appendChild(inp);
  div.appendChild(small);
  list.appendChild(div);
};

let firstInput=list.querySelector("input");
let firstMsg=list.querySelector("small");
firstInput.onblur=()=>validate(firstInput,firstMsg);
</script>
</body>
</html>
```

5) **Password and Confirm Password Real-time Matching**
   Create a form that takes Password and Confirm Password inputs.
   - **As user types in either field, dynamically check if they match.**
   - **Show messages: “Passwords match” (green) or “Passwords do not match” (red).**

Source Code:

```
<html>
<head>
  <title>question 5</title>
  <style>
    body{font-family:Arial;margin:30px;}
    input{display:block;margin:5px 0;padding:5px;}
    small{display:block;margin-top:3px;}
    .ok{color:green;}
    .err{color:red;}
  </style>
</head>
<body>
  <h3>Password Matching</h3>
  <input type="password" id="pass" placeholder="Password">
  <input type="password" id="cpass" placeholder="Confirm Password">
  <small id="msg"></small>
```

```
<script>
let pass=document.getElementById("pass");
let cpass=document.getElementById("cpass");
let msg=document.getElementById("msg");

function check(){
 if(cpass.value==="") msg.textContent="";
 else if(pass.value===cpass.value){
  msg.textContent="Passwords match ✓"; msg.className="ok";
 }else{
  msg.textContent="Passwords do not match ✗"; msg.className="err";
 }
}

pass.oninput=check;
cpass.oninput=check;
</script>
</body>
</html>
```

6) **Username Validation Using Regex to Restrict Special Characters**
   Develop a form where the username input must only allow letters, numbers, underscores without special characters.
   • Use regex /^[a-zA-Z0-9_]{3,16}$/ to validate username length between 3 and 16 characters.
   • Prevent form submission if invalid and display an error message.

Source Code:

```
<html>
<head>
 <title>question 6</title>
 <style>
  body{font-family:Arial;margin:30px;}
  input{padding:5px;margin-bottom:5px;}
  small{display:block;margin-top:3px;}
  .ok{color:green;}
  .err{color:red;}
 </style>
```

```
</head>
<body>
 <h3>Username</h3>
 <form id="uForm">
  <input type="text" id="user" placeholder="Enter username">
  <small id="msg"></small>
  <button type="submit">Submit</button>
 </form>

<script>
let form=document.getElementById("uForm");
let user=document.getElementById("user");
let msg=document.getElementById("msg");
let regex=/^[a-zA-Z0-9_]{3,16}$/;

form.onsubmit=(e)=>{
 if(!regex.test(user.value)){
  e.preventDefault();
  msg.textContent="Invalid username (3-16 letters, numbers, underscores
only)";
  msg.className="err";
 }else{
  msg.textContent="Valid username ✓";
  msg.className="ok";
 }
}
</script>
</body>
</html>
```

7) **Date Format Validation 'DD/MM/YYYY' with Regex**
   Design a form with a date input field that requires the date format DD/MM/YYYY.
   • Use the regex /^\\d{2}\\/\\d{2}\\/\\d{4}$/ to confirm the format.
   • Alert user with an error message if format is incorrect upon form submission.

Source Code:

```
<html>
<head>
 <title>question 7</title>
 <style>
  body{font-family:Arial;margin:30px;}
  input{padding:5px;margin-bottom:5px;}
  small{color:red;display:block;}
 </style>
</head>
<body>
 <h3>Date Format</h3>
 <form id="dForm">
  <input type="text" id="date" placeholder="DD/MM/YYYY">
  <small id="msg"></small>
  <button type="submit">Submit</button>
 </form>
<script>
let form=document.getElementById("dForm");
let date=document.getElementById("date");
let msg=document.getElementById("msg");
let regex=/^\d{2}\/\d{2}\/\d{4}$/;

form.onsubmit=(e)=>{
 if(!regex.test(date.value)){
  e.preventDefault();
  msg.textContent="Invalid date format! Use DD/MM/YYYY";
 }else{
  msg.textContent="";
  alert("Date submitted: "+date.value);
 }
}
</script>
</body>
</html>
```

8) Hexadecimal Color Code Validation with Live Preview
Create an input field for color codes (e.g., #FFF or #FFFFFF).
• Validate the input with regex /^#([A-Fa-f0-9]{6}|[A-Fa-f0-9]{3})$/.
• Provide a live color preview box beside the input that changes as the value updates.

Source Code:

```
<html>
<head>
 <title>Hex Color Validation</title>
 <style>
  #box {
   width: 50px;
   height: 50px;
   display: inline-block;
   border: 1px solid #000;
   vertical-align: middle;
   margin-left: 10px;
  }
  #msg {
   display: block;
   margin-top: 6px;
   font-family: Arial, sans-serif;
  }
 </style>
</head>
<body>
 <h2>Color Code</h2>
 <input id="color" placeholder="#FFF or #FFFFFF">
 <div id="box"></div>
 <small id="msg"></small>

 <script>
  let color = document.getElementById("color");
  let box = document.getElementById("box");
  let msg = document.getElementById("msg");
  let regex = /^#([A-Fa-f0-9]{6}|[A-Fa-f0-9]{3})$/;

  color.oninput = () => {
```

```
    if (regex.test(color.value)) {
      msg.textContent = "Valid ✓";
      msg.style.color = "green";
      box.style.background = color.value;
    } else {
      msg.textContent = "Invalid ✗";
      msg.style.color = "red";
      box.style.background = "white";
    }
  };
 </script>
</body>
</html>
```

9) Dynamic Country Dropdown Based on Selected Continent
Implement two cascading dropdowns: one for continents and another for countries.
• When the user selects a continent, JavaScript updates the country list dynamically.
• Use objects or maps to manage continent-country data.

Source Code:

```
<html>
<head>
 <title>question 9</title>
 <style>
  body{font-family:Arial;margin:30px;}
  select{padding:5px;margin:5px;}
 </style>
</head>
<body>
 <h2>Select Continent and Country</h2>
 <select id="continent">
  <option value="">--Choose Continent--</option>
  <option>Africa</option>
  <option>Asia</option>
  <option>Europe</option>
 </select>
```

```
 <select id="country">
  <option value="">--Choose Country--</option>
 </select>

<script>
let data={
 "Africa":["Nigeria","Kenya","Egypt","Argentina"],
 "Asia":["India","China","Japan","Korea"],
 "Europe":["Germany","France","Italy","Paris"]
};

let cont=document.getElementById("continent");
let coun=document.getElementById("country");

cont.onchange=()=>{
 coun.innerHTML="<option>--Choose Country--</option>";
 if(data[cont.value]){
  data[cont.value].forEach(c=>{
   let opt=document.createElement("option");
   opt.textContent=c;
   coun.appendChild(opt);
  });
 }
}
</script>
</body>
</html>
```

10) Dynamic Form Builder UI

Create a simple UI where users can add different types of form inputs dynamically (e.g., text, email, password).

- **Users select input type from a dropdown and click "Add".**
- **JavaScript generates the form inputs dynamically and appends them to the form.**
- **Validate all inputs with appropriate regex depending on type.**

Source Code:

```
<html>
<head>
 <title>question 10</title>
 <style>
  body{font-family:Arial;margin:30px;}
  select,button,input{margin:5px;padding:5px;}
  small{display:block;margin-bottom:5px;}
  .ok{color:green;} .err{color:red;}
 </style>
</head>
<body>
 <h2>Dynamic Form Builder</h2>

 <select id="type">
  <option value="text">Text</option>
  <option value="email">Email</option>
  <option value="password">Password</option>
  <option value="username">Username</option>
  <option value="date">Date (DD/MM/YYYY)</option>
 </select>
 <button type="button" id="addBtn">Add</button>

 <form id="dynForm"></form>

<script>
let addBtn=document.getElementById("addBtn");
let form=document.getElementById("dynForm");

let regex={
 text:/^[A-Za-z\s]+$/,
 email:/^[^@\s]+@[^@\s]+\.[^@\s]+$/,
 password:/^(?=.*[A-Z])(?=.*\d).{8,}$/,
 username:/^[a-zA-Z0-9_]{3,16}$/,
 date:/^\d{2}\/\d{2}\/\d{4}$/
};

addBtn.onclick=()=>{
 let type=document.getElementById("type").value;
```

```
  let div=document.createElement("div");
  let inp=document.createElement("input");
  let msg=document.createElement("small");

  inp.placeholder=type;
  inp.type=(type==="password")?"password":(type==="email"?"email":"text"
);

  inp.oninput=()=>{
    if(regex[type].test(inp.value)){
      msg.textContent="Valid ✓"; msg.className="ok";
    }else{
      msg.textContent="Invalid ✗"; msg.className="err";
    }
  };

  div.appendChild(inp);
  div.appendChild(msg);
  form.appendChild(div);
};
</script>
</body>
</html>
```