JavaScript Assignments — Solutions and Outputs

Each question is numbered. Below each question: (a) HTML/CSS/JS code (ready to paste into an .html file) and (b) Expected output / how it behaves.

1. Registration Form Validation

Form with username, email, password, confirm password and validation using JavaScript.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1">
<title>1. Registration Form Validation</title>
<style>
  body{font-family:Arial;padding:16px} form{max-width:420px} label{display:block;margin-top:8px} input{width:100%;pa
  .success{color:green}
</style>
</head>
<body>
<h2>Registration</h2>
<form id="regForm" novalidate>
  <label>Username<input type="text" id="username"></label>
  <label>Email<input type="email" id="email"></label>
  <label>Password<input type="password" id="password"></label>
  <label>Confirm Password<input type="password" id="confirm"></label>
  <button type="submit">Register</button>
  <div id="message" aria-live="polite"></div>
</form>
<script>
document.getElementById('regForm').addEventListener('submit', function(e){
  e.preventDefault();
  const u = document.getElementById('username').value.trim();
  const em = document.getElementById('email').value.trim();
  const pw = document.getElementById('password').value;
  const cpw = document.getElementById('confirm').value;
  const msg = document.getElementById('message');
  msg.innerHTML = ''; msg.className = '';

  if(!u){ msg.textContent='Username should not be empty.'; msg.className='error'; return; }
  // basic email check: contains @ and domain after it
  if(!/^[^@\s]+@[^@\s]+\.[^@\s]+$/.test(em)){ msg.textContent='Please enter a valid email.'; msg.className='error';
  if(pw.length < 8){ msg.textContent='Password must be at least 8 characters.'; msg.className='error'; return; }
  if(pw !== cpw){ msg.textContent='Confirm password must match.'; msg.className='error'; return; }

  msg.textContent='Registration successful!'; msg.className='success';
});
</script>
</body>
</html>
```

Output / Behavior:

Expected behavior: On submit, shows inline error message if validations fail. Shows 'Registration successful!' when all checks pass.

2. Simple To■Do List (Add / Complete / Remove)

A minimal to-do list using DOM manipulation.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1">
<title>2. To-Do List</title>
<style>body{font-family:Arial;padding:12px} #task{width:70%} ul{margin-top:12px} li.completed{text-decoration:line-
</head>
<body>
<h2>To-Do</h2>
<input id="task" placeholder="New task"><button id="add">Add</button>
<ul id="list" aria-live="polite"></ul>
<script>
const addBtn=document.getElementById('add');
const list=document.getElementById('list');
addBtn.addEventListener('click', addTask);
document.getElementById('task').addEventListener('keydown', (e)=>{ if(e.key==='Enter') addTask(); });

function addTask(){
  const t = document.getElementById('task').value.trim();
  if(!t) return;
  const li = document.createElement('li');
  li.textContent = t;
  // complete toggle
  li.addEventListener('click', ()=> li.classList.toggle('completed'));
  // remove button
  const rem = document.createElement('button');
  rem.textContent='Remove';
  rem.style.marginLeft='8px';
  rem.addEventListener('click',(e)=>{ e.stopPropagation(); li.remove(); });
  li.appendChild(rem);
  list.appendChild(li);
  document.getElementById('task').value='';
}
</script>
</body>
</html>
```

Output / Behavior:

Expected behavior: Add tasks with button or Enter. Click a task to mark complete (toggle). Click Remove to delete the single task.

3. Countdown Timer

Timer that counts down every second from user-input seconds.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1">
<title>3. Countdown Timer</title>
<style>body{font-family:Arial;padding:12px} #display{font-size:2rem;margin-top:12px}</style>
</head>
<body>
<h2>Countdown Timer</h2>
<input type="number" id="seconds" placeholder="Seconds"><button id="start">Start</button>
<div id="display" aria-live="polite"></div>
<script>
let timer=null, remaining=0;
document.getElementById('start').addEventListener('click', ()=>{
  clearInterval(timer);
  remaining = Math.max(0, parseInt(document.getElementById('seconds').value,10) || 0);
  const disp = document.getElementById('display');
  if(remaining<=0){ disp.textContent='Enter a positive number'; return; }
  disp.textContent = remaining + 's';
  timer = setInterval(()=>{
    remaining--;
    if(remaining>0) disp.textContent = remaining + 's';
    else { clearInterval(timer); disp.textContent = "Time’s up!"; }
  },1000);
});
</script>
</body>
</html>
```

Output / Behavior:

Expected behavior: When started, shows remaining seconds every second. Displays "Time’s up!" when reaches zero.

4. Palindrome Checker Function

Function that returns true/false for palindrome strings (case & non-alphanumeric ignored).

Code:

```
// Palindrome function (can be run in browser console)
function isPalindrome(s){
  const cleaned = s.toString().toLowerCase().replace(/[^a-z0-9]/g,'');
  return cleaned === cleaned.split('').reverse().join('');
}

// Examples:
console.log(isPalindrome('madam')); // true
console.log(isPalindrome('A man, a plan, a canal: Panama')); // true
console.log(isPalindrome('hello')); // false
```

Output / Behavior:

Expected behavior: Returns true for palindromes, false otherwise. Ignores case and non-alphanumeric characters.

5. Sort Array without using sort()

Sort numbers ascending and descending using simple algorithms (bubble sort shown).

Code:

```
// Ascending and descending using bubble sort
function bubbleSortAsc(arr){
  const a = arr.slice();
  for(let i=0;i<a.length-1;i++){
    for(let j=0;j<a.length-1-i;j++){
      if(a[j]>a[j+1]){ const t=a[j]; a[j]=a[j+1]; a[j+1]=t; }
    }
  }
  return a;
}
function bubbleSortDesc(arr){ return bubbleSortAsc(arr).reverse(); }

// Example:
const nums = [5,2,9,1,5,6];
console.log('Asc:', bubbleSortAsc(nums)); // [1,2,5,5,6,9]
console.log('Desc:', bubbleSortDesc(nums)); // [9,6,5,5,2,1]
```

Output / Behavior:

Expected behavior: Shows ascending and descending sorted arrays without using built-in sort().

6. Quiz App (5 questions)

Single-page quiz with 5 MCQs, shows final score.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1">
<title>6. Quiz App</title>
<style>body{font-family:Arial;padding:12px} .q{margin-bottom:12px}</style>
</head>
<body>
<h2>Quiz</h2>
<form id="quizForm">
  <div class="q"><p>1. 2+2 = ?</p>
    <label><input type="radio" name="q1" value="3">3</label>
    <label><input type="radio" name="q1" value="4">4</label>
  </div>
  <div class="q"><p>2. Capital of France?</p>
    <label><input type="radio" name="q2" value="Paris">Paris</label>
    <label><input type="radio" name="q2" value="London">London</label>
  </div>
  <div class="q"><p>3. JS stands for?</p>
    <label><input type="radio" name="q3" value="JavaScript">JavaScript</label>
    <label><input type="radio" name="q3" value="Java">Java</label>
  </div>
  <div class="q"><p>4. 5*6 = ?</p>
    <label><input type="radio" name="q4" value="30">30</label>
    <label><input type="radio" name="q4" value="20">20</label>
  </div>
  <div class="q"><p>5. HTML is for?</p>
    <label><input type="radio" name="q5" value="structure">Structure</label>
    <label><input type="radio" name="q5" value="style">Style</label>
  </div>
  <button type="submit">Submit</button>
</form>
<div id="result" aria-live="polite"></div>
<script>
const answers = {q1:'4', q2:'Paris', q3:'JavaScript', q4:'30', q5:'structure'};
document.getElementById('quizForm').addEventListener('submit', function(e){
  e.preventDefault();
  let score=0;
  for(const k in answers){
    const val = (new FormData(this)).get(k);
    if(val === answers[k]) score++;
  }
  document.getElementById('result').textContent = 'Your score: ' + score + ' / 5';
});
</script>
</body>
</html>
```

Output / Behavior:

Expected behavior: Select answers, submit, and see score out of 5.

7. Stopwatch (Start / Stop / Reset)

Basic stopwatch using setInterval.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1">
<title>7. Stopwatch</title>
<style>body{font-family:Arial;padding:12px} #time{font-size:2rem;margin:8px 0}</style>
</head>
<body>
<h2>Stopwatch</h2>
<div id="time">00:00:00.00</div>
<button id="start">Start</button><button id="stop">Stop</button><button id="reset">Reset</button>
<script>
let startTime=0, elapsed=0, timer=null;
const fmt = ms => {
  const cent = Math.floor((ms%1000)/10);
  const s = Math.floor(ms/1000)%60;
  const m = Math.floor(ms/60000)%60;
  return [m,s,cent].map(n=>String(n).padStart(2,'0')).join(':').replace(/:(?=[0-9]{2}$)/,':');
};
function render(){ document.getElementById('time').textContent = fmt(elapsed + (Date.now()-startTime || 0)); }
document.getElementById('start').addEventListener('click', ()=>{
  if(timer) return;
  startTime = Date.now();
  timer = setInterval(render, 50);
});
document.getElementById('stop').addEventListener('click', ()=>{
  if(!timer) return;
  clearInterval(timer); timer=null; elapsed += Date.now() - startTime;
  render();
});
document.getElementById('reset').addEventListener('click', ()=>{
  clearInterval(timer); timer=null; startTime=0; elapsed=0; document.getElementById('time').textContent='00:00:00.00
});
</script>
</body>
</html>
```

Output / Behavior:

Expected behavior: Start begins counting, Stop pauses, Reset clears to 00:00:00.00.

8. Random Quote from Array

Displays a random quote from an array when a button is clicked.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1">
<title>8. Random Quotes</title>
<style>body{font-family:Arial;padding:12px} #quote{font-style:italic;margin-top:12px}</style>
</head>
<body>
<h2>Random Quote</h2>
<button id="qbtn">Show Quote</button>
<div id="quote" aria-live="polite"></div>
<script>
const quotes = [
  "The only limit is your mind.",
  "Dream big and dare to fail.",
  "Action is the foundational key to success.",
  "Stay hungry, stay foolish.",
  "Simplicity is the ultimate sophistication.",
  "Believe you can and you're halfway there.",
  "Quality is not an act, it's a habit.",
  "Small steps every day.",
  "Learn as if you will live forever.",
  "The future depends on what you do today."
];
document.getElementById('qbtn').addEventListener('click', ()=>{
  const r = quotes[Math.floor(Math.random()*quotes.length)];
  document.getElementById('quote').textContent = r;
});
</script>
</body>
</html>
```

Output / Behavior:

Expected behavior: Clicking the button displays a random quote from the 10 stored quotes.

9. Password Strength Meter

Shows Weak/Medium/Strong based on length, number and special chars.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1">
<title>9. Password Strength</title>
<style>body{font-family:Arial;padding:12px} #strength{font-weight:bold;margin-top:8px}</style>
</head>
<body>
<h2>Password Strength</h2>
<input type="password" id="pwd" placeholder="Enter password">
<div id="strength" aria-live="polite"></div>
<script>
document.getElementById('pwd').addEventListener('input', function(){
  const v = this.value;
  let score = 0;
  if(v.length >= 8) score++;
  if(/[0-9]/.test(v)) score++;
  if(/[^A-Za-z0-9]/.test(v)) score++;
  const out = document.getElementById('strength');
  if(score <= 1) { out.textContent='Weak'; out.style.color='red'; }
  else if(score === 2) { out.textContent='Medium'; out.style.color='orange'; }
  else { out.textContent='Strong'; out.style.color='green'; }
});
</script>
</body>
</html>
```

Output / Behavior:

Expected behavior: As user types, shows Weak/Medium/Strong based on simple scoring.

10. Shopping Cart with Dynamic Pricing

Product list with Add to Cart, update quantity, remove, and total calculation.

Code:

```
<!DOCTYPE html>
<html lang="en">
<head><meta charset="utf-8"><meta name="viewport" content="width=device-width,initial-scale=1">
<title>10. Shopping Cart</title>
<style>body{font-family:Arial;padding:12px} .product{margin-bottom:8px} table{width:100%;border-collapse:collapse} 
</head>
<body>
<h2>Products</h2>
<div id="products"></div>
<h3>Cart</h3>
<table id="cart"><thead><tr><th>Item</th><th>Qty</th><th>Price</th><th>Total</th><th>Action</th></tr></thead><tbody>
<div id="bill">Total: $0.00</div>
<script>
const products = [
  {id:1,name:'T-shirt',price:15.99},
  {id:2,name:'Mug',price:7.5},
  {id:3,name:'Cap',price:9.99}
];
const productsDiv = document.getElementById('products');
const cartTbody = document.querySelector('#cart tbody');
const bill = document.getElementById('bill');
let cart = {};

function renderProducts(){
  products.forEach(p=>{
    const d = document.createElement('div'); d.className='product';
    d.innerHTML = `<strong>${p.name}</strong> - $${p.price.toFixed(2)} <button data-id="${p.id}">Add to Cart</butto
    productsDiv.appendChild(d);
  });
}
function updateCartUI(){
  cartTbody.innerHTML='';
  let total=0;
  Object.values(cart).forEach(item=>{
    const tr = document.createElement('tr');
    const lineTotal = item.price * item.qty;
    total += lineTotal;
    tr.innerHTML = `<td>${item.name}</td>
      <td><input type="number" min="1" value="${item.qty}" data-id="${item.id}" class="qty"></td>
      <td>$${item.price.toFixed(2)}</td>
      <td>$${lineTotal.toFixed(2)}</td>
      <td><button class="remove" data-id="${item.id}">Remove</button></td>`;
    cartTbody.appendChild(tr);
  });
  bill.textContent = 'Total: $' + total.toFixed(2);
  // attach listeners for qty and remove
  document.querySelectorAll('.qty').forEach(input=> input.addEventListener('change', function(){
    const id = this.dataset.id; const q = Math.max(1,parseInt(this.value,10)||1);
    cart[id].qty = q; updateCartUI();
  }));
  document.querySelectorAll('.remove').forEach(b=> b.addEventListener('click', function(){
    delete cart[this.dataset.id]; updateCartUI();
  }));
}
productsDiv.addEventListener('click', e=>{
  if(e.target.tagName==='BUTTON'){
    const id = e.target.dataset.id;
    const p = products.find(x=> x.id==id);
    if(cart[id]) cart[id].qty++;
    else cart[id] = { ...p, qty: 1 };
    updateCartUI();
  }
});
renderProducts();
</script>
</body>
</html>
```

Output / Behavior:

Expected behavior: Click Add to Cart to add items. Change quantity to update totals. Remove removes item. Total updates dynamically.