Q1 (a) Explain why create a website. (5)

Definition (1 line):
A website is created to publish information or services online so users anywhere can access content, interact, transact or communicate.

Why create a website (3–4 short points):

- **Share information & presence:** Publish company/portfolio/product details 24/7 to a global audience.
- **Reach & marketing:** Promote brand, attract customers, perform SEO and lead generation.
- **Sales & transactions:** Enable e-commerce, bookings, online payments and business operations.
- **Communication & support:** Provide contact forms, chat, FAQs and customer support.
- **Credibility & trust:** Professional site builds trust; central hub for social links & contact info.

Short example sentence:
A small business builds a website to show services, accept orders and collect customer enquiries.

Q1 (b) Describe 5 golden rules of web designing. (5)

Golden rules (each short):

1. **Keep it simple & clear:** Minimal clutter — one main idea per page; reduce choices that distract users.
2. **Consistent layout:** Consistent fonts, colors, button styles and spacing across pages.
3. **Mobile-first / responsive:** Design for smallest screens first; ensure content adapts to all devices.
4. **Visible navigation:** Make menus intuitive and reachable in 1–2 clicks — use clear labels and hierarchy.
5. **Readable content & good contrast:** Use legible fonts, proper line-height, and sufficient color contrast for accessibility.

Tip line: Test with real users and measure load speed — fast and usable wins.

Q2 (a) Explain Page design techniques. (5)

Definition (1 line):
Page design techniques are practices used to arrange content and visuals to make web pages usable, attractive and effective.

Techniques (3–4 concise bullets):

- **Visual hierarchy:** Use size, weight and color to prioritize headings, CTAs and content.
- **Grid systems & alignment:** Use a grid (columns, gutters) for consistent spacing and layout balance.
- **Whitespace & breathing room:** Provide margins/padding around elements to improve readability.
- **Typography & contrast:** Choose readable fonts, proper line-length and contrast between text/background.
- **Consistent navigation & CTAs:** Keep menus and call-to-action placement consistent for scanning users.
- **Responsive layout:** Use flexible units, media queries and fluid images so layout adapts.

Small note: Sketch wireframes first (low-fi) before implementing.

Q2 (b) Design a student registration form by using form tag. (5)

HTML (exam-ready, minimal & accessible):

```
<form id="regForm" action="/submit" method="post">
 <label for="name">Name:</label>
 <input type="text" id="name" name="name" required><br>

 <label for="email">Email:</label>
 <input type="email" id="email" name="email" required><br>

 <label>Gender:</label>
```

```
 <input type="radio" id="m" name="gender" value="M"><label for="m">Male</label>
 <input type="radio" id="f" name="gender" value="F"><label
for="f">Female</label><br>

 <label for="course">Course:</label>
 <select id="course" name="course">
  <option value="MCA">MCA</option>
  <option value="BCA">BCA</option>
 </select><br>

 <label for="bio">About:</label><br>
 <textarea id="bio" name="bio" rows="4"></textarea><br>

 <button type="submit">Register</button>
</form>
```

Notes for full marks: Use required, type="email", labels with for, and server-side validation in addition to HTML validation.

Q3 (a) Explain CSS selectors and their types with an example. (5)

Definition (1 line):
CSS selectors specify the HTML elements to which style rules apply.

Common selector types (short + example):

- **Type (element) selector:** targets element names.
  p { color: #333; }
- **Class selector:** targets elements with class.
  .card { padding: 10px; }
- **ID selector:** unique element.
  #header { height: 60px; }
- **Attribute selector:** target by attribute.
  input[type="text"] { width: 200px; }

- **Descendant selector:** space — any descendant.
  nav a { color: blue; }
- **Child selector:** > immediate child.
  ul > li { display: inline-block; }
- **Adjacent sibling (+) / General sibling (~)** selectors.
- **Pseudo-classes:** a:hover, :first-child, :nth-child(2).
- **Pseudo-elements:** ::before, ::after, ::first-line.
- **Grouping selector (comma):** h1,h2,h3 { margin:0; } — applies same rule.

Example:

/* select all paragraphs inside .content and make first line bold */

.content p::first-line { font-weight: bold; }

Mark tip: mention specificity order (inline > id > class > element) briefly.

Q3 (b) Design a navigation bar with at least 5 links on it. (Use Internal CSS to style it). (5)

HTML + Internal CSS (exam-ready):

```
<head>
<style>
nav { background: #333; }
nav ul { list-style: none; margin: 0; padding: 0; display: flex; }
nav li { margin: 0; }
nav a {
  display: block; padding: 12px 16px; color: #fff; text-decoration: none;
}
nav a:hover { background: #444; }
@media (max-width: 600px) {
  nav ul { flex-direction: column; }
  nav a { padding: 10px; }
}
```

```
</style>
</head>
<body>
<nav>
 <ul>
  <li><a href="#">Home</a></li>
  <li><a href="#">About</a></li>
  <li><a href="#">Services</a></li>
  <li><a href="#">Portfolio</a></li>
  <li><a href="#">Contact</a></li>
 </ul>
</nav>
</body>
```

Notes: display:flex for horizontal layout; media query stacks links on small screens.

Q4 (a) List the Basics of Page Layout Design techniques. (5)

Concise list (each 1 line):

1. **Grid system:** Use column grids (e.g., 12-column) for consistency and alignment.
2. **Flexible (fluid) vs fixed layouts:** Decide percent-based (fluid) or pixel-based (fixed) based on need.
3. **Flexbox/Grid:** Use CSS Flexbox for 1D layouts and CSS Grid for complex 2D layouts.
4. **Whitespace & margins:** Use spacing to separate sections and improve readability.
5. **Header / main / sidebar / footer arrangement:** Logical flow places navigation/header above, content center, sidebar optional.
6. **Responsive breakpoints:** define breakpoints (mobile / tablet / desktop) and adapt layout.
7. **Content-first ordering:** place important content where users expect (top-left primary).

Tip: Sketch wireframe → build with CSS Grid/Flexbox → test responsively.

Q4 (b) Explain the different ways of designing navigation bar. (5)

Different ways (short descriptions):

1. **Horizontal top nav:** Common across the top — typical for desktop. (Use <nav> with inline/flex items.)
2. **Vertical sidebar nav:** Left or right fixed sidebar for apps or dashboards.
3. **Dropdown menus:** Hover/click opens nested links — good for many pages.
4. **Hamburger / off-canvas menu:** Compact toggle for mobile; slides in from side.
5. **Mega menu:** Large panel with grouped links for complex sites (e-commerce).
6. **Sticky/fixed nav:** Fixed at top while scrolling — good for quick access.
7. **Footer nav / breadcrumb:** Supplementary navigation for context and location.

Design notes: For accessibility use keyboard-focusable links, ARIA attributes for dropdowns, and descriptive link text.

Q5 (a) Explain how to create responsive website. (5)

Definition (1 line):
Responsive design makes web pages render well on a variety of devices and screen sizes.

Steps/techniques (3–4 concise points):

- **Viewport meta:** Include <meta name="viewport" content="width=device-width,initial-scale=1">.
- **Fluid grids & relative units:** Use %, em, rem for widths and typography instead of fixed px.
- **Responsive images:** img { max-width:100%; height:auto; } and use srcset for multiple resolutions.
- **CSS media queries:** change layout/styles for breakpoints.
- @media (max-width: 768px) { /* mobile rules */ }
- **Modern layout systems:** use Flexbox/Grid to rearrange elements easily.

- **Mobile-first approach:** write base styles for small screens, add enhancements for larger screens.

Short example (media query):

```
.container { display: grid; grid-template-columns: 1fr 300px; }
@media (max-width: 700px) {
  .container { grid-template-columns: 1fr; } /* stack columns */
}
```

Q5 (b) Design a webpage with different block with different background images (use different CSS color properties/values to design it). (5)

HTML + CSS (exam-ready example):

```
<style>
.section { height: 220px; color: white; display:flex; align-items:center; justify-content:center; font-size:20px; }
.block1 {
  background-image: url('bg1.jpg');
  background-size: cover; background-position: center;
  background-repeat: no-repeat;
  background-color: rgba(0,0,0,0.4); /* fallback/color overlay */
  background-blend-mode: multiply;
}
.block2 {
  background: linear-gradient(135deg, rgba(58,123,213,0.8), rgba(0,210,255,0.7));
  background-image: url('bg2.jpg');
  background-size: cover; background-position: center;
  color: #fff;
}
.block3 {
  background: radial-gradient(circle at top left, #ff9a9e, #fad0c4);
```

```
  color: #222;
}
.block4 {
  background-color: #222;
  background-image: url('bg3.jpg');
  background-size: contain;
  background-repeat: no-repeat;
  background-position: right center;
}
.container { display: grid; grid-template-columns: 1fr; gap: 10px; padding: 10px; }
@media(min-width:700px) { .container { grid-template-columns: repeat(2, 1fr); } }
</style>

<div class="container">
  <div class="section block1">Block 1 — Hero image</div>
  <div class="section block2">Block 2 — Gradient + image</div>
  <div class="section block3">Block 3 — Radial color</div>
  <div class="section block4">Block 4 — Image right</div>
</div>
```