1. What is the primary difference between `@Controller` **and** `@RestController` **in Spring MVC?**
A) `@Controller` returns view names, while `@RestController` returns data directly in response body
B) `@Controller` only supports GET requests, while `@RestController` supports all HTTP methods
C) `@RestController` cannot return HTML views
D) Both A and C

Answer: D

2. Which annotation combination makes a `@Controller` **class behave like a** `@RestController`**?**
A) `@Controller` + `@ResponseBody` on each method
B) `@Controller` + `@RequestBody`
C) `@Controller` + `@Component`
D) `@Controller` + `@Service`

Answer: A

3. What is the main purpose of `JdbcTemplate` **in Spring Boot?**
A) To automatically generate SQL queries
B) To simplify JDBC operations and reduce boilerplate code
C) To replace Hibernate and JPA
D) To manage database connections only

Answer: B

4. Which HTTP method is mapped by `@GetMapping` **annotation?**
A) POST
B) GET
C) PUT
D) All HTTP methods

Answer: B

5. In Spring MVC, what does the "Model" component typically contain?
A) Database connection information
B) Business logic
C) Data to be displayed in the view
D) HTTP request parameters

Answer: C

6. How would you inject a `JdbcTemplate` **into a Spring Boot repository class?**
A) Using `@Inject` annotation
B) Using `@Autowired` on constructor or field

C) Using `@Resource` annotation
D) All of the above

Answer: D (All are valid, though `@Autowired` is most common)

7. Which of the following correctly shows a `@RestController` method using `JdbcTemplate`?
A)

```
@GetMapping("/users")
public List<User> getUsers() {
   return jdbcTemplate.execute("SELECT * FROM users");
}
```

B)

```
@GetMapping("/users")
public List<User> getUsers() {
   return jdbcTemplate.query("SELECT * FROM users", new BeanPropertyRowMapper<>(U
ser.class));
}
```

C)

```
@PostMapping("/users")
public void getUsers() {
   jdbcTemplate.update("SELECT * FROM users");
}
```

D)

```
@GetMapping("/users")
public String getUsers() {
   return jdbcTemplate.queryForString("SELECT * FROM users");
}
```

Answer: B

8. What is the role of the "DispatcherServlet" in Spring MVC?
A) Renders HTML views
B) Acts as a front controller, routing requests to appropriate handlers
C) Manages database transactions
D) Configures application properties

Answer: B

9. In a `@Controller` class, how do you add data to the model for a view?

```
@GetMapping("/products")
public String getProducts(__________ model) {
   model.addAttribute("products", productService.findAll());
   return "products";
}
```

A) `HttpServletRequest`
B) `Model`
C) `ModelMap`
D) Both B and C

Answer: D

10. Which of these correctly demonstrates a complete Spring MVC flow using `@Controller` and `JdbcTemplate`?
A) Browser → `@RestController` → `JdbcTemplate` → View (HTML)
B) Browser → `@Controller` → Service → `JdbcTemplate` → Database → Model → View (HTML)
C) Browser → `@Controller` → `JdbcTemplate` → Database → JSON Response
D) Browser → `@GetMapping` → `JdbcTemplate` → View (HTML)

Answer: B

11. What is the difference between `query()` and `queryForObject()` methods in `JdbcTemplate`?
A) `query()` returns multiple rows, `queryForObject()` returns a single row
B) `query()` returns a List, `queryForObject()` returns a single object
C) `query()` is for SELECT statements, `queryForObject()` is for INSERT statements
D) Both A and B