R18

Code No: 155DM

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY HYDERABAD
B. Tech III Year I Semester Examinations, January/February - 2023
WEB PROGRAMMING
(Common to CESE, CSE(AIML), CSE(DS))

Time: 3 Hours **Max. Marks: 75**

Note: i) Question paper consists of Part A, Part B.
ii) Part A is compulsory, which carries 25 marks. In Part A, Answer all questions.
iii) In Part B, Answer any one question from each unit. Each question carries 10 marks and may have a, b as sub questions.

PART – A

(25 Marks)

1.a) Define Heading Tags in HTML with an example. [2]

b) What is the scope of the variables in JavaScript? [3]

c) List string manipulation functions of Java String class. [2]

d) What is a package? How to define it and access it? [3]

e) Define TCP sockets. [2]

f) Discuss about Inetaddress class. [3]

g) Define event handling. [2]

h) Write a short note on cookies. [3]

i) Define UDDI. [2]

j) Write down the applications of XML. [3]

PART – B

(50 Marks)

2.a) Explain the structure of the HTML webpage with an example.
b) Explain various operators and data types available in java script with examples.[5+5]

OR

3.a) 'Javascript is referred to as an Object based programming language'. Justify with an example.
b) Explain how basic tables and nested tables are created using HTML. [5+5]

4.a) Does Java support multi way selection statements? Justify your answer.
b) Can inheritance be applied between interfaces? Justify your answer. [5+5]

OR

5. What is an exception? How are exceptions handled in Java programming? Explain with suitable programs. [10]

6.a) Write a JDBC program to update the amount balance in an account after every withdrawal. Assume the necessary database table.

b) Discuss about parameter passing in RMI, with examples. [5+5]

OR

7. Explain the following methods bind(), rebind(), createSubcontext(), getAttributes(), modifyAttributes() methods with syntax. [10]

8.a) What is the role of servlet in web application? What are the differences between HttpServlet and GenericServlet?

b) Present and explain the life cycle of a servlet. [5+5]

OR

9. What is a Layout manager? Discuss different types of layout managers used in JAVA in brief. [10]

10.a) How is XML defined? Write down the XML syntax and structure rules. What is DTD? Discuss its applications.

b) Discuss the role of WSDL web service in AJAX. [5+5]

OR

11. Design an XML schema for hospital information management. [10]

---ooOoo—

Answer Key

PART - A

1.a) Define Heading Tags in HTML with an example.
Heading tags in HTML (H1 to H6) define the headings on a webpage, with H1 being the highest level.
Example: `<h1>This is a heading</h1>`

1.b) What is the scope of the variables in JavaScript?
In JavaScript, variables can have global scope (accessible everywhere) or local scope (accessible within a function).
Variables declared with `let` and `const` have block scope.

1.c) List string manipulation functions of Java String class.
Common string manipulation functions in Java's String class include `length()`, `substring()`, `indexOf()`, `replace()`, `toLowerCase()`, and `toUpperCase()`.

1.d) What is a package? How to define it and access it?
A package in Java is a namespace for organizing classes and interfaces.
It is defined with the `package` keyword at the top of the Java file.
Classes in a package are accessed using the `import` statement.

1.e) Define TCP sockets.
TCP sockets are endpoints for communication between two machines over a network.
They use the Transmission Control Protocol (TCP) to ensure reliable data transfer.

1.f) Discuss about Inetaddress class.
The `InetAddress` class in Java provides methods to get IP addresses of hosts.
It can handle both IPv4 and IPv6 addresses.
Methods include `getByName()`, `getLocalHost()`, and `getHostAddress()`.

1.g) Define event handling.
Event handling is the mechanism in programming that captures and responds to user actions or other events.
It typically involves event listeners and event handlers.

1.h) Write a short note on cookies.
Cookies are small pieces of data stored on a user's browser.
They are used to remember information about the user, such as login status or preferences.
Cookies are sent with HTTP requests to the server.

1.i) Define UDDI.
UDDI (Universal Description, Discovery, and Integration) is a platform-independent framework for listing web services.
It allows businesses to publish and discover services over the web.

1.j) Write down the applications of XML.
XML (eXtensible Markup Language) is used for data representation and storage.
Applications include web services (SOAP), configuration files, data interchange (between systems), and document formatting.

PART - B

2.a) Explain the structure of the HTML webpage with an example.
1. HTML Document Structure: An HTML document consists of various tags such as `<html>`, `<head>`, and `<body>`.
2. <!DOCTYPE html>: This declaration defines the document type and version of HTML.
3. <html>: This tag encloses the entire HTML document and includes attributes like `lang`.
4. <head>: This section contains metadata, title, and links to scripts and stylesheets.
5. <title>: This tag specifies the title of the webpage, displayed in the browser tab.
6. <body>: This section contains the content of the webpage such as text, images, links, tables, etc.
7. Example:
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>My Webpage</title>
```

```
</head>
<body>
    <h1>Welcome to My Webpage</h1>
    <p>This is a paragraph.</p>
</body>
</html>
```

2.b) Explain various operators and data types available in JavaScript with examples.

1. Arithmetic Operators: Perform mathematical calculations like addition (+), subtraction (-), multiplication (*), and division (/).
Example: `let sum = 5 + 3;`
2. Comparison Operators: Compare values and return a boolean result, such as equal (==), strict equal (===), greater than (>), and less than (<).
Example: `let isEqual = (5 == '5');`
3. Logical Operators: Perform logical operations and return boolean values, including AND (&&), OR (||), and NOT (!).
Example: `let result = (5 > 3 && 2 < 4);`
4. Assignment Operators: Assign values to variables, such as `=`, `+=`, `-=`.
Example: `let x = 10; x += 5;`
5. Data Types:
Number: Represents numeric values.
Example: `let age = 25;`
String: Represents sequences of characters.
Example: `let name = "John";`
Boolean: Represents true or false values.
Example: `let isActive = true;`
Object: Represents complex data structures.
Example: `let person = { name: "Alice", age: 30 };`
Undefined: Indicates a variable has not been assigned a value.
Example: `let x;`
Null: Represents an intentional absence of any object value.
Example: `let emptyValue = null;`

3.a) 'JavaScript is referred to as an Object-based programming language'. Justify with an example.

1. Objects: JavaScript allows creating and manipulating objects.

Example: `let car = { brand: "Toyota", model: "Camry", year: 2020 };`
2. Methods: Functions can be stored as properties within objects.
Example: `car.start = function() { console.log("Car started"); };`
3. Prototypes: JavaScript objects can inherit properties and methods from prototypes.
Example:

```javascript
function Person(name, age) {
    this.name = name;
    this.age = age;
}
Person.prototype.greet = function() {
    console.log("Hello, " + this.name);
};
let person1 = new Person("John", 25);
person1.greet(); // "Hello, John"
```

3.b) Explain how basic tables and nested tables are created using HTML.
1. Basic Table:
<table>: Defines the table structure.
<tr>: Defines a table row.
<td>: Defines a table cell.
Example:

```html
<table border="1">
    <tr>
        <td>Row 1, Cell 1</td>
        <td>Row 1, Cell 2</td>
    </tr>
    <tr>
        <td>Row 2, Cell 1</td>
        <td>Row 2, Cell 2</td>
    </tr>
</table>
```

2. Nested Table:
<table> inside <td>: Defines a table within a table cell.

Example:

```html
<table border="1">
  <tr>
    <td>
      <table border="1">
        <tr>
          <td>Nested Row 1, Cell 1</td>
        </tr>
        <tr>
          <td>Nested Row 2, Cell 1</td>
        </tr>
      </table>
    </td>
    <td>Main Table Row 1, Cell 2</td>
  </tr>
</table>
```

4.a) Does Java support multi-way selection statements? Justify your answer.

1. Yes, Java supports multi-way selection statements.
2. Switch Statement: Allows multi-way branching based on the value of an expression.

Example:

```java
int day = 3;
switch(day) {
  case 1:
    System.out.println("Monday");
    break;
  case 2:
    System.out.println("Tuesday");
    break;
  case 3:
    System.out.println("Wednesday");
    break;
  default:
    System.out.println("Other day");
```

```
        break;
    }
```

4.b) Can inheritance be applied between interfaces? Justify your answer.

1. Yes, inheritance can be applied between interfaces in Java.
2. Extends Keyword: An interface can extend another interface, inheriting its methods.

Example:

```java
interface Animal {
    void eat();
}
interface Dog extends Animal {
    void bark();
}
class Bulldog implements Dog {
    public void eat() {
        System.out.println("Bulldog eats");
    }
    public void bark() {
        System.out.println("Bulldog barks");
    }
}
```

5. What is an exception? How are exceptions handled in Java programming? Explain with suitable programs.

1. Exception Definition: An exception is an event that disrupts the normal flow of a program's execution.
2. Try-Catch Block: Exceptions are handled using try-catch blocks.
3. Throw Keyword: Used to throw an exception explicitly.
4. Finally Block: Contains code that executes regardless of whether an exception occurred.
5. Example Program:

```java
public class ExceptionExample {
    public static void main(String[] args) {
```

```
        try {
            int result = 10 / 0; // This will cause an ArithmeticException
        } catch (ArithmeticException e) {
            System.out.println("Exception caught: " + e.getMessage());
        } finally {
            System.out.println("This block executes no matter what.");
        }
    }
}
```

6.a) Write a JDBC program to update the amount balance in an account after every withdrawal. Assume the necessary database table.

1. Import JDBC Packages:
```java
import java.sql.*;
```
2. Database Connection Setup:
```java
String url = "jdbc:mysql://localhost:3306/bank";
String username = "root";
String password = "password";
```
3. SQL Update Statement:
```java
String sql = "UPDATE accounts SET balance = balance - ? WHERE account_id = ?";
```
4. Establish Connection and Execute Update:
```java
try (Connection conn = DriverManager.getConnection(url, username, password);
     PreparedStatement pstmt = conn.prepareStatement(sql)) {
    pstmt.setDouble(1, 100.00); // Amount to withdraw
    pstmt.setInt(2, 1); // Account ID
    int rowsAffected = pstmt.executeUpdate();
    System.out.println(rowsAffected + " rows updated.");
} catch (SQLException e) {
```

```
        e.printStackTrace();
    }
```

6.b) Discuss parameter passing in RMI, with examples.

1. Remote Method Invocation (RMI): Allows objects to invoke methods on remote objects.
2. Pass-by-Value: Primitive data types and Serializable objects are passed by value.
Example: `public void updateBalance(double amount);`
3. Pass-by-Reference: Remote objects (subclasses of `UnicastRemoteObject`) are passed by reference.
Example: `public void updateAccount(RemoteAccount account);`
4. Example Code:

```java
// Remote Interface
public interface AccountService extends Remote {
    void withdraw(double amount) throws RemoteException;
}

// Remote Object Implementation
public class AccountServiceImpl extends UnicastRemoteObject implements AccountService {
    protected AccountServiceImpl() throws RemoteException {
        super();
    }
    public void withdraw(double amount) throws RemoteException {
        System.out.println("Withdrawal of " + amount);
    }
}
```

7. Explain the following methods bind(), rebind(), createSubcontext(), getAttributes(), modifyAttributes() methods with syntax.

1. bind(): Binds a name to an object in a directory.
Syntax: `context.bind("name", object);`
2. rebind(): Rebinds a name to a new object, replacing any existing binding.
Syntax: `context.rebind("name", object);`

3. createSubcontext(): Creates a subcontext (subdirectory) within the current context.
Syntax: `context.createSubcontext("subcontextName");`
4. getAttributes(): Retrieves the attributes associated with a named object.
Syntax: `Attributes attrs = dirContext.getAttributes("name");`
5. modifyAttributes(): Modifies the attributes of a named object.
Syntax: `dirContext.modifyAttributes("name", DirContext.REPLACE_ATTRIBUTE, attrs);`

8.a) What is the role of servlet in web application? What are the differences between HttpServlet and GenericServlet?
1. Role of Servlet:
Request Handling: Processes client requests and generates dynamic responses.
Lifecycle Management: Managed by the web container (init, service, destroy).
Session Management: Tracks user sessions.
2. Differences:
HttpServlet: Specialized for HTTP protocol, handles HTTP-specific methods (GET, POST).
Example: `protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException;`
GenericServlet: Protocol-independent, must override `service()` method.
Example: `public void service(ServletRequest req, ServletResponse res) throws ServletException, IOException;`

8.b) Present and explain the life cycle of a servlet.
1. Initialization (`init` method): Called once when the servlet is first loaded, used to initialize resources.
Example: `public void init() throws ServletException { /* Initialization code */ }`
2. Request Handling (`service` method): Called for each request, processes the request and generates a response.
Example: `public void service(ServletRequest req, ServletResponse res) throws ServletException, IOException { /* Request processing code */ }`
3. Destruction (`destroy` method): Called once when the servlet is being unloaded, used to release resources.
Example: `public void destroy() { /* Cleanup code */ }`

4. Example Lifecycle Code:

```java
public class MyServlet extends HttpServlet {
    public void init() throws ServletException {
        // Initialization code
    }
    protected void doGet(HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException {
        // Handle GET request
    }
    public void destroy() {
        // Cleanup code
    }
}
```

9. What is a Layout manager? Discuss different types of layout managers used in JAVA in brief.

1. Definition: A Layout Manager in Java is used to arrange components within a container.
2. FlowLayout: Arranges components in a left-to-right flow, wrapping to a new line as needed.
Example: `container.setLayout(new FlowLayout());`
3. BorderLayout: Divides the container into five regions (North, South, East, West, Center).
Example: `container.setLayout(new BorderLayout());`
4. GridLayout: Arranges components in a grid of cells with equal size.
Example: `container.setLayout(new GridLayout(2, 2));`
5. CardLayout: Allows switching between different components like cards.
Example: `container.setLayout(new CardLayout());`
6. GridBagLayout: Provides a flexible way to arrange components with varying sizes and positions.
Example: `container.setLayout(new GridBagLayout());`

10.a) How is XML defined? Write down the XML syntax and structure rules. What is DTD? Discuss its applications.

1. Definition: XML (eXtensible Markup Language) is used to define and transport data.

2. Syntax and Structure Rules:
Prolog: `<?xml version="1.0" encoding="UTF-8"?>`
Root Element: Every XML document must have one root element.
Tags: Must be properly nested and case-sensitive.
Attributes: Enclosed in quotes.
3. Example:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<book>
   <title>XML Guide</title>
   <author>John Doe</author>
</book>
```

4. DTD (Document Type Definition): Defines the structure and rules for an XML document.
Example: `<!DOCTYPE book [ <!ELEMENT book (title, author)> <!ELEMENT title (#PCDATA)> <!ELEMENT author (#PCDATA)> ]>`
5. Applications:
Data Validation: Ensures XML document structure adheres to defined rules.
Interoperability: Standardizes data exchange between systems.
Configuration Files: Used in various applications to store configuration settings.

10.b) Discuss the role of WSDL web service in AJAX.
1. Definition of WSDL: WSDL (Web Services Description Language) is an XML-based language for describing web services.
2. Service Description: Specifies the location, methods, and data types of web services.
3. Role in AJAX:
Service Discovery: AJAX can use WSDL to locate and interact with web services.
Dynamic Calls: Enables AJAX applications to make dynamic calls to web services without hardcoding service details.
Data Interchange: Facilitates data exchange between client-side AJAX and server-side web services.
4. Example Usage:
AJAX Request:

```javascript
var xhr = new XMLHttpRequest();
```

```
xhr.open("POST", "http://example.com/service?wsdl", true);
xhr.setRequestHeader("Content-Type", "text/xml");
xhr.send(soapMessage);
```

11. Design an XML schema for hospital information management.

1. Schema Definition: Defines the structure and constraints of XML documents.
2. Root Element: `<hospital>`
3. Child Elements: `<patient>`, `<doctor>`, `<department>`, `<appointment>`
4. Example XML Schema:

```xml
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="hospital">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="patient" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="id" type="xs:int"/>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="age" type="xs:int"/>
              <xs:element name="gender" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="doctor" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="id" type="xs:int"/>
              <xs:element name="name" type="xs:string"/>
              <xs:element name="specialization" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="department" maxOccurs="unbounded">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="id" type="xs:int"/>
```

```
            <xs:element name="name" type="xs:string"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="appointment" maxOccurs="unbounded">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="patient_id" type="xs:int"/>
            <xs:element name="doctor_id" type="xs:int"/>
            <xs:element name="date" type="xs:date"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
 </xs:element>
</xs:schema>
```