Slips 1

Q. 1)

a) Write a C++ program to check maximum and minimum of two integer numbers. **(Use Inline Function and Conditional Operator)** **[Marks 15]**

```
#include <iostream>
using namespace std;

// Inline function to find the maximum of two numbers
inline int max(int a, int b) {
   return (a > b) ? a : b;
}

// Inline function to find the minimum of two numbers
inline int min(int a, int b) {
   return (a < b) ? a : b;
}

int main() {
   int num1, num2;

   cout << "Enter two integers: ";
   cin >> num1 >> num2;

   cout << "Maximum: " << max(num1, num2) << endl;
   cout << "Minimum: " << min(num1, num2) << endl;

   return 0;
}
```

b) Write a C++ Program to find volume of cylinder, cone and sphere. **(Use function overloading)**

```
#include <iostream>
using namespace std;
```

```
float volume(float r, float h) // Cylinder Volume
{
   return (3.14 * r * r * h);
}

float coneVol(float r, float h) // Cone Volume
{
   return (3.14 * r * r * h / 3);
}

float volume(float r) // Sphere Volume
{
   return (4.0 / 3.0 * 3.14 * r * r * r);
}

int main() {
   float cy_h, cy_r, co_h, co_r, sp_r;
   cout << "Enter dimensions" << endl;

   cout << "1. Cylinder" << endl;
   cout << "Height: ";
   cin >> cy_h;
   cout << "Radius: ";
   cin >> cy_r;
   cout << endl;

   cout << "2. Cone" << endl;
   cout << "Height: ";
   cin >> co_h;
   cout << "Radius: ";
```

```
    cin >> co_r;

    cout << endl;

    cout << "3. Sphere" << endl;
    cout << "Radius: ";
    cin >> sp_r;
    cout << endl;

    cout << "The volume of Cylinder is: " << volume(cy_r, cy_h) << endl;
    cout << "The volume of Cone is: " << coneVol(co_r, co_h) << endl;
    cout << "The volume of Sphere is: " << volume(sp_r) << endl;

    return 0;
}
```

Q. 2)

a) Write a PHP script to create a simple calculator that can accept two numbers and perform operations like add, sub, mum (Use the concept of Class)

```
<?php
class Calculator {
    function add($a, $b) { return $a + $b; }
    function subtract($a, $b) { return $a - $b; }
    function multiply($a, $b) { return $a * $b; }
}

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $num1 = $_POST['num1'];
    $num2 = $_POST['num2'];
    $calc = new Calculator();
```

```
    echo "<p>Addition: {$calc->add($num1, $num2)}</p>";
    echo "<p>Subtraction: {$calc->subtract($num1, $num2)}</p>";
    echo "<p>Multiplication: {$calc->multiply($num1, $num2)}</p>";
}
?>

<form method="post">
    <p>Number 1: <input type="number" name="num1" required></p>
    <p>Number 2: <input type="number" name="num2" required></p>
    <p><input type="submit" value="Calculate"></p>
</form>
```

b) Write a PHP Script to create student.xml file which contains student roll no, name, address, college course. Print students details of specific course in tabular format after accepting course as input.

```
<?php
$xmlFile = 'student.xml';

// Check if XML file exists, if not, create it
// if exists , then it will not create new one
if (!file_exists($xmlFile)) {
    $students = new SimpleXMLElement('<students></students>');

    $student1 = $students->addChild('student');
    $student1->addChild('roll_no', '101');
    $student1->addChild('name', 'Nikhil');
    $student1->addChild('address', 'Silvassa');
    $student1->addChild('college', 'SSR College');
    $student1->addChild('course', 'BBA (CA)');

    $student2 = $students->addChild('student');
    $student2->addChild('roll_no', '102');
```

```
    $student2->addChild('name', 'Aman');
    $student2->addChild('address', 'Lavachha');
    $student2->addChild('college', 'SSR College');
    $student2->addChild('course', 'Business Administration');

    $student3 = $students->addChild('student');
    $student3->addChild('roll_no', '103');
    $student3->addChild('name', 'Kunal');
    $student3->addChild('address', 'TokarKhada');
    $student3->addChild('college', 'SSR College');
    $student3->addChild('course', 'Business Administration');

    $students->asXML($xmlFile);
}

// Load XML data
$xml = simplexml_load_file($xmlFile);
$selectedCourse = $_POST['course'] ?? '';

echo "<form method='post'>
        <p>Enter Course: <input type='text' name='course' required></p>
        <p><input type='submit' value='Search'></p>
      </form>";

if (!empty($selectedCourse)) {
    echo "<h2>Students in '$selectedCourse' Course</h2>";
    echo "<table border='1'>";
    echo "<tr><th>Roll
No</th><th>Name</th><th>Address</th><th>College</th><th>Course</th></tr>";

    $found = false;
```

```
    foreach ($xml->student as $student) {
        if (strtolower(trim($student->course)) == strtolower(trim($selectedCourse))) {
            echo "<tr>";
            echo "<td>{$student->roll_no}</td>";
            echo "<td>{$student->name}</td>";
            echo "<td>{$student->address}</td>";
            echo "<td>{$student->college}</td>";
            echo "<td>{$student->course}</td>";
            echo "</tr>";
            $found = true;
        }
    }

    if (!$found) {
        echo "<tr><td colspan='5'>No students found for '$selectedCourse'.</td></tr>";
    }

    echo "</table>";
}
?>
```

Slips 2

Q. 1)

a) Write a C++ Program to interchange values of two integer numbers. **(Use call by reference)**

[Marks 15]

```
#include <iostream>
using namespace std;

// Function to swap two integers using call by reference
void swap(int &a, int &b) {
    int temp = a;
```

```
    a = b;
    b = temp;
}

int main() {
    int num1, num2;

    cout << "Enter two integers: ";
    cin >> num1 >> num2;

    cout << "\nBefore swapping:" << endl;
    cout << "num1 = " << num1 << ", num2 = " << num2 << endl;

    swap(num1, num2); // call by reference

    cout << "\nAfter swapping:" << endl;
    cout << "num1 = " << num1 << ", num2 = " << num2 << endl;

    return 0;
}
```

b) Write a C++ Program to create two classes square and rectangle. Compare area of both the shapes using **Friend Function**. Accept appropriate data members for both the classes.

[Marks 25]

```
#include <iostream>
using namespace std;

// Forward declaration
class Rectangle;

class Square {
```

```
    float side;

public:
    void input() {
        cout << "Enter side of square: ";
        cin >> side;
    }

    float getArea() const {
        return side * side;
    }

    // Declare friend function
    friend void compareArea(Square s, Rectangle r);
};

class Rectangle {
    float length, breadth;

public:
    void input() {
        cout << "Enter length and breadth of rectangle: ";
        cin >> length >> breadth;
    }

    float getArea() const {
        return length * breadth;
    }

    // Declare friend function
    friend void compareArea(Square s, Rectangle r);
```

```
};

// Friend function to compare areas
void compareArea(Square s, Rectangle r) {
    float areaSquare = s.getArea();
    float areaRectangle = r.getArea();

    cout << "\nArea of Square = " << areaSquare << endl;
    cout << "Area of Rectangle = " << areaRectangle << endl;

    if (areaSquare > areaRectangle)
        cout << "Square has a larger area." << endl;
    else if (areaRectangle > areaSquare)
        cout << "Rectangle has a larger area." << endl;
    else
        cout << "Both have equal area." << endl;
}

int main() {
    Square s;
    Rectangle r;

    s.input();
    r.input();

    compareArea(s, r);

    return 0;
}
```

Q. 2)

a) Write a PHP script to demonstrate the introspection for examining classes and objects. (use function get_declared_classes() ,get_class_inethods() and get_class_vars()). **[Marks 15]**

```
<?php
// Get all declared classes before defining a new one
$before = get_declared_classes();

// Define a simple class
class SampleClass {
    public $property1 = "Hello";
    public $property2 = "World";
    public $property3 = "Bankai";
    private $hiddenProperty = "Secret";

    public function method1() {
        return "This is method1";
    }

    public function method2() {
        return "This is method2";
    }

    public function method3() {
        return "This is method3";
    }
}

// Create an instance
$obj = new SampleClass();
```

```
// Get declared classes after defining SampleClass
$after = get_declared_classes();

// Find user-defined classes
$userClasses = array_diff($after, $before);
if (!in_array('SampleClass', $userClasses)) {
   $userClasses[] = 'SampleClass';
}

// Display results
echo "\nUser-Defined Classes:\n";
print_r($userClasses);
echo"<br>";

echo "\nMethods of SampleClass:\n";
print_r(get_class_methods('SampleClass'));
echo"<br>";

echo "\nPublic Properties of SampleClass:\n";
print_r(get_class_vars('SampleClass'));
echo"<br>";
?>
```

b) Write a script to solve follow ing questions (Use “Student.xml” file) **[Marks 25]**

i) Create a D OM Document object and load this XML file.

ii) Get the output of this Docurrlent to the browser.

iii) Save this [. XML] document in another format i.e. in [.doc]

Write a XML Script to print the names of the student present in “Student.xml” file.

```
<?php
// Load the XML file
$xml = new DOMDocument();
```

```
$xml->load("student.xml");

// Extract and print student names
$students = $xml->getElementsByTagName("student");

echo "Student Names:<br>";
foreach ($students as $student) {
    echo "- " . $student->getElementsByTagName("name")->item(0)->nodeValue . "<br>";
}

// Save the XML content to a .doc file
file_put_contents("student.doc", $xml->saveXML());


?>
```

Slips 3

Q. 1)

a) Write a C++ program to accept Worker information worker name, no of hours worked, pay rate and salary. Write necessary functions to calculate and display the salary of worker. **(Use default value for pay_rate)** **[Marks 15]**

```
#include <iostream>
using namespace std;

class Worker {
    string name;
    int hours;

public:
    void accept() {
        cout << "\nEnter Name: ";
        cin >> name;
```

```
        cout << "Enter Hours: ";
        cin >> hours;
    }

    void calculateSalary(int rate = 20) {  // Default parameter in declaration
        cout << "\nSalary of " << name << " is: " << (hours * 10) * rate << endl;
    }
};

int main() {
    Worker w1, w2;

    w1.accept();
    w1.calculateSalary(50);  // Custom rate

    w2.accept();
    w2.calculateSalary();  // Default rate

    return 0;
}
```

b) Write a C++ Program to create a base class Employee(EmpCode, Empname, EmpSalary). Derive two classes as fulltime(DailyWages, No_Of_Days) and Parttime (No_of_Hrs, Hr_Wages).

Write a menu driven program to perform following functions.

1. Accept the details of ‘n’ employees and calculate salary.
2. Display the details of ‘n’ employees.
3. Display the details of employee having maximum salary for both types of employees

[Marks 25]

```
#include <iostream>
using namespace std;

class Employee
```

```
{
public:
    int empCode;
    string empName;
    float empSalary;

    virtual void accept() = 0;
    virtual void calculateSalary() = 0;
    virtual void display() = 0;
    float getSalary() { return empSalary; }
};

class FullTime : public Employee
{
public:
    float dailyWages;
    int noOfDays;

    void accept()
    {
        cout << "Enter Full-Time Employee Code: ";
        cin >> empCode;
        cout << "Enter Name: ";
        cin >> empName;
        cout << "Enter Daily Wages: ";
        cin >> dailyWages;
        cout << "Enter No. of Days: ";
        cin >> noOfDays;
    }

    void calculateSalary()
```

```
    {
        empSalary = dailyWages * noOfDays;
    }

    void display()
    {
        cout << "Full-Time Employee: " << empName << ", Salary: " << empSalary << endl;
    }
};

class PartTime : public Employee
{
public:
    int hours;
    float hourlyWage;

    void accept()
    {
        cout << "Enter Part-Time Employee Code: ";
        cin >> empCode;
        cout << "Enter Name: ";
        cin >> empName;
        cout << "Enter Hourly Wage: ";
        cin >> hourlyWage;
        cout << "Enter Hours Worked: ";
        cin >> hours;
    }

    void calculateSalary()
    {
        empSalary = hours * hourlyWage;
```

```
    }

    void display()
    {
        cout << "Part-Time Employee: " << empName << ", Salary: " << empSalary << endl;
    }
};

int main()
{
    FullTime ft[10];
    PartTime pt[10];
    int ftCount = 0, ptCount = 0;
    int choice;

    while (true)
    {
        cout << "\n1. Add Employee\n2. Show All\n3. Show Max Salary\n0. Exit\nEnter choice: ";
        cin >> choice;

        switch (choice)
        {
        case 1:
        {
            int type;
            cout << "Enter 1 for Full-Time, 2 for Part-Time: ";
            cin >> type;
            if (type == 1)
            {
                ft[ftCount].accept();
                ft[ftCount].calculateSalary();
```

```
            ftCount++;
        }
        else
        {
            pt[ptCount].accept();
            pt[ptCount].calculateSalary();
            ptCount++;
        }
        break;
    }
    case 2:
    {
        cout << "\nFull-Time Employees:\n";
        for (int i = 0; i < ftCount; i++)
            ft[i].display();
        cout << "\nPart-Time Employees:\n";
        for (int i = 0; i < ptCount; i++)
            pt[i].display();
        break;
    }
    case 3:
    {
        float maxFT = 0, maxPT = 0;
        int m1 = -1, m2 = -1;

        for (int i = 0; i < ftCount; i++)
        {
            if (ft[i].getSalary() > maxFT)
            {
                maxFT = ft[i].getSalary();
                m1 = i;
```

```
            }
        }
        for (int i = 0; i < ptCount; i++)
        {
            if (pt[i].getSalary() > maxPT)
            {
                maxPT = pt[i].getSalary();
                m2 = i;
            }
        }

        if (m1 != -1)
        {
            cout << "Highest Paid Full-Time: ";
            ft[m1].display();
        }
        if (m2 != -1)
        {
            cout << "Highest Paid Part-Time: ";
            pt[m2].display();
        }
        break;
    }
    case 0:
        return 0;
    default:
        cout << "Invalid choice.\n";
    }
  }
}
```

Q. 2)

a) Write a Calculator class that can accept two values, then add, subtract, multiply them or divide them on request.

For example:

$calc = new C alculator(3, 4); echo $ calc- > add(); // Displays “7”

echo $ calc- > multiply(); // Displays “12” **[Marks l5]**

```
<?php
class Calculator {
   public $num1, $num2;

   // Set values manually
   public function setValues($a, $b) {
      $this->num1 = $a;
      $this->num2 = $b;
   }

   // Addition
   public function add() {
      return $this->num1 + $this->num2;
   }

   // Subtraction
   public function subtract() {
      return $this->num1 - $this->num2;
   }

   // Multiplication
   public function multiply() {
      return $this->num1 * $this->num2;
   }
```

```
    // Division (with zero check)
    public function divide() {
        if ($this->num2 == 0) {
            return "Error: Division by zero!";
        }
        return $this->num1 / $this->num2;
    }
}

// Example Usage
$calc = new Calculator();
$calc->setValues(5, 4);  // Setting values manually

echo "Addition: ";
echo $calc->add() . "<br>";
echo"Subtraction: "   ;
echo $calc->subtract() . "<br>";
echo"Multiplication: "  ;
echo $calc->multiply() . "<br>";
echo"Division: "  ;
echo $calc->divide() . "<br>";
?>
```

b) Write a script to create “cricket.xml” file with multiple elements as shown below:

```
< CricketTeam>
< Team country=”Australia”>
< players>    </players>
<runs>    </runs>
<wicket>   </wicket>
```

```
</Team>
</CricketTeam>
```

Write a script to add multiple elements in “cricket.xml” file of category, country=”India”.**[Marks 25]**

```
<?php
$root = new SimpleXMLElement("<root/>");

$e1 = $root->addChild("CricketTeam");

$e2 = $e1->addChild("team");
$e2->addAttribute("country", "India");
$e2->addChild("player", "Virat Kohli");
$e2->addChild("run", "10000");
$e2->addChild("wicket", "5");

$e2 = $e1->addChild("team");
$e2->addAttribute("country", "Pakistan");
$e2->addChild("player", "Babar Azam");
$e2->addChild("run", "1000");
$e2->addChild("wicket", "2");

$e2 = $e1->addChild("team");
$e2->addAttribute("country", "India");
$e2->addChild("player", "Rohit Sharma");
$e2->addChild("run", "5000");
$e2->addChild("wicket", "4");

$root->asXML("cricket.xml");

?>
```

Slips 4

a) Consider the following C++ Class

Class Person

{

Char Name[20];

Char Add[30];

Float Salary;

Float Tax_Amt;

Public :

};

Calculate tax amount by checking salary of a person

-> for salary <=20000 Tax Rate = 0

-> for salary >20000 || <=40000 Tax Rate = 5% of salary

-> for salary >40000 Tax Rate = 10% of salary **[Marks 15]**

```
#include <iostream>
using namespace std;

class Person {
   char name[20];
   char address[30];
   float salary;
   float tax;

public:
   void getData() {
      cout << "Enter name (no spaces): ";
      cin >> name;

      cout << "Enter address (no spaces): ";
      cin >> address;
```

```
        cout << "Enter salary: ";
        cin >> salary;
    }

    void calculateTax() {
        if (salary <= 20000)
            tax = 0;
        else if (salary <= 40000)
            tax = salary * 0.05;
        else
            tax = salary * 0.10;
    }

    void display() {
        cout << "\nName: " << name << endl;
        cout << "Address: " << address << endl;
        cout << "Salary: " << salary << endl;
        cout << "Tax Amount: " << tax << endl;
    }
};

int main() {
    Person p;
    p.getData();
    p.calculateTax();
    p.display();
    return 0;
}
```

b) Design two base classes student(sid,sname,class) and Competition(Cid, Cname). Derive a class Stud_Comp(Rank) from it. Write a menu driven program to perform following functions.

i. Accept Information

ii. Display Information

iii. Display student details in the ascending order of rank of a specified competition.

(Use array of objects) **[Marks 25]**

```
#include <iostream>
#include <cstring> // for strcmp
using namespace std;

class Student
{
public:
   int sid;
   char sname[20];
   char sclass[10];
};

class Competition
{
public:
   int cid;
   char cname[20];
};

class Stud_Comp : public Student, public Competition
{
public:
   int rank;

   void accept()
```

```
    {
        cout << "Enter Student ID: ";
        cin >> sid;
        cout << "Enter Name: ";
        cin >> sname;
        cout << "Enter Class: ";
        cin >> sclass;
        cout << "Enter Competition ID: ";
        cin >> cid;
        cout << "Enter Competition Name: ";
        cin >> cname;
        cout << "Enter Rank: ";
        cin >> rank;
    }

    void display()
    {
        cout << "ID: " << sid << ", Name: " << sname << ", Class: " << sclass;
        cout << ", CompID: " << cid << ", CompName: " << cname << ", Rank: " << rank << endl;
    }
};

int main()
{
    Stud_Comp s[20];
    int n, choice;
    char compSearch[20];
    bool run = true;

    cout << "Enter number of students: ";
    cin >> n;
```

```
while (run)
{
    cout << "\n--- MENU ---\n";
    cout << "1. Accept Info\n";
    cout << "2. Display All Info\n";
    cout << "3. Display Sorted by Rank (Competition)\n";
    cout << "4. Exit\n";
    cout << "Enter choice: ";
    cin >> choice;

    switch (choice)
    {
    case 1:
        for (int i = 0; i < n; i++)
        {
            cout << "\nEnter info for student " << i + 1 << ":\n";
            s[i].accept();
        }
        break;

    case 2:
        for (int i = 0; i < n; i++)
        {
            s[i].display();
        }
        break;

    case 3:
        cout << "Enter Competition Name to search: ";
        cin >> compSearch;
```

```
    // Sort by rank (only matching competition names)
    for (int i = 0; i < n - 1; i++)
    {
        for (int j = i + 1; j < n; j++)
        {
            if (strcmp(s[i].cname, compSearch) == 0 &&
                strcmp(s[j].cname, compSearch) == 0 &&
                s[i].rank > s[j].rank)
            {
                Stud_Comp temp = s[i];
                s[i] = s[j];
                s[j] = temp;
            }
        }
    }

    cout << "\nStudents in " << compSearch << " sorted by rank:\n";
    for (int i = 0; i < n; i++)
    {
        if (strcmp(s[i].cname, compSearch) == 0)
        {
            s[i].display();
        }
    }
    break;

case 4:
    run = false;
    break;
```

```
        default:
            cout << "Invalid choice!\n";
            break;
        }
    }

    return 0;
}
```

Q. 2)

a) Write a PHP program to accept two string from user and check whether entered strings are matching or not. **(Use sticky form concept)** **[Marks 15]**

```
<!DOCTYPE html>
<html>
<head>
    <title>String Match Checker</title>
</head>
<body>
    <h2>Check if Two Strings Match</h2>

    <form method="post">
        Enter First String: <input type="text" name="str1" value="<?php if(isset($_POST['str1'])) echo $_POST['str1']; ?>"><br><br>
        Enter Second String: <input type="text" name="str2" value="<?php if(isset($_POST['str2'])) echo $_POST['str2']; ?>"><br><br>
        <input type="submit" value="Check">
    </form>

    <?php
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $str1 = $_POST['str1'];
        $str2 = $_POST['str2'];
```

```
    if ($str1 == $str2) {
        echo "<p style='color:green;'>Strings match ✅</p>";
    } else {
        echo "<p style='color:red;'>Strings do not match ❌</p>";
    }
}
?>
</body>
</html>
```

b) Write an Ajax script to get player details from XML file when user select player name. Create XML file to store details of player (name, country, wickets and runs). **[Marks 25]**

```
<!DOCTYPE html>
<html>
<head>
    <title>Player Details using AJAX and XML</title>
    <script>
    function getPlayerDetails(name) {
        var xhttp = new XMLHttpRequest();
        xhttp.onreadystatechange = function() {
            if (this.readyState == 4 && this.status == 200) {
                var xmlDoc = this.responseXML;
                var players = xmlDoc.getElementsByTagName("player");
                var output = "Player not found.";

                for (var i = 0; i < players.length; i++) {
                    var playerName =
players[i].getElementsByTagName("name")[0].childNodes[0].nodeValue;
                    if (playerName === name) {
                        var country =
players[i].getElementsByTagName("country")[0].childNodes[0].nodeValue;
```

```
            var wickets = players[i].getElementsByTagName("wickets")[0].childNodes[0].nodeValue;
            var runs = players[i].getElementsByTagName("runs")[0].childNodes[0].nodeValue;

            output = "<b>Name:</b> " + playerName + "<br>" +
                "<b>Country:</b> " + country + "<br>" +
                "<b>Wickets:</b> " + wickets + "<br>" +
                "<b>Runs:</b> " + runs;
            break;
          }
        }

        document.getElementById("result").innerHTML = output;
      }
    };
    xhttp.open("GET", "players.xml", true);
    xhttp.send();
  }
  </script>
</head>
<body>
  <h2>Select a Player</h2>
  <select onchange="getPlayerDetails(this.value)">
    <option value="">--Select--</option>
    <option value="Virat Kohli">Virat Kohli</option>
    <option value="Steve Smith">Steve Smith</option>
    <option value="Joe Root">Joe Root</option>
  </select>

  <div id="result" style="margin-top:20px;"></div>
</body>
</html>
```

Slips 5

Q. 1)

a) Create a C++ Class Student with data members Rollno, Sname, Class, Percentage. Accept two students information and display information of students having maximum percentage. (Use this pointer. **[Marks 15]**

```
#include <iostream>

using namespace std;


class Student {

private:

   int rollno;

   char sname[20];

   char sclass[10];

   float percentage;


public:

   void accept() {

      cout << "Enter Roll Number: ";

      cin >> rollno;

      cout << "Enter Name: ";

      cin >> sname;

      cout << "Enter Class: ";

      cin >> sclass;

      cout << "Enter Percentage: ";

      cin >> percentage;

   }


   void display() {

      cout << "Roll No: " << rollno << endl;

      cout << "Name: " << sname << endl;

      cout << "Class: " << sclass << endl;
```

```
        cout << "Percentage: " << percentage << "%" << endl;
    }

    float getPercentage() {
        return this->percentage;
    }
};

int main() {
    Student s1, s2;

    cout << "Enter details of Student 1:\n";
    s1.accept();

    cout << "\nEnter details of Student 2:\n";
    s2.accept();

    cout << "\nStudent with Maximum Percentage:\n";

    if (s1.getPercentage() > s2.getPercentage()) {
        s1.display();
    } else {
        s2.display();
    }

    return 0;
}
```

b) Create a C++ Class FixDeposite with data members FD_No, Cust_Name, FD_Amt, Interest rate, Maturity_Amt, No_Of_Months. Create and initialize all values of FixDeposit object by using parameterized constructor default value for interest rate. Calculate maturity amt using interest rate

and display all the details.

[Marks 25]

```
#include <iostream>
using namespace std;

class FixDeposit
{
   int fd_no;
   string name;
   float amount;
   float interest;
   int months;
   float maturity;

public:
   // Constructor with default interest = 5%
   FixDeposit(int no, string n, float amt, int m, float rate = 5.0)
   {
      fd_no = no;
      name = n;
      amount = amt;
      months = m;
      interest = rate;
      maturity = amount + (amount * interest * months) / (100 * 12);
   }

   void display()
   {
      cout << "\nFD No: " << fd_no << endl;
      cout << "Name: " << name << endl;
      cout << "Amount: " << amount << endl;
```

```
        cout << "Interest Rate: " << interest << "%" << endl;
        cout << "Months: " << months << endl;
        cout << "Maturity Amount: " << maturity << endl;
    }
};

int main()
{
    int no, months,rate;
    char name[20];
    float amt;

    cout << "Enter FD No: ";
    cin >> no;
    cout << "Enter Name: ";
    cin >> name;
    cout << "Enter Amount: ";
    cin >> amt;
    cout << "Enter Months: ";
    cin >> months;
    cout << "Enter Interest Rate (default is 5%): ";
    cin >> rate;

    // Using default interest rate
    FixDeposit fd(no, name, amt, months);
    fd.display();

    cout << "\n\n";
    // Using user-defined interest rate
    FixDeposit fd2(no, name, amt, months, rate);
    fd2.display();
```

```
    return 0;
}
```

Q. 2)

a) Create an abstract class Shape with methods area( ) and volume( ). Derive three classes rectangle (length, breath), Circle(radius) and Cylinder(radius, height), Calculate area and volume of all. (Use Method overriding). **[Marks l5]**

```
<?php

abstract class Shape
{
    abstract public function area();
    abstract public function volume();
}

// Rectangle class
class Rectangle extends Shape
{
    private $length, $breadth;

    public function __construct($l, $b)
    {
        $this->length = $l;
        $this->breadth = $b;
    }

    public function area()
    {
        return $this->length * $this->breadth;
```

```
    }

    public function volume()
    {
        return 0; // Not applicable
    }
}

// Circle class
class Circle extends Shape
{
    private $radius;

    public function __construct($r)
    {
        $this->radius = $r;
    }

    public function area()
    {
        return 3.14 * $this->radius * $this->radius;
    }

    public function volume()
    {
        return 0; // Not applicable
    }
}

// Cylinder class
class Cylinder extends Shape
```

```
{
    private $radius, $height;

    public function __construct($r, $h)
    {
        $this->radius = $r;
        $this->height = $h;
    }

    public function area()
    {
        return 2 * 3.14 * $this->radius * ($this->radius + $this->height); // Surface Area
    }

    public function volume()
    {
        return 3.14 * $this->radius * $this->radius * $this->height;
    }
}

// Example usage:
$rectangle = new Rectangle(5, 3);
echo "Rectangle Area: " . $rectangle->area() . "<br>";
echo "Rectangle Volume: " . $rectangle->volume() . "<br><br>";

$circle = new Circle(4);
echo "Circle Area: " . $circle->area() . "<br>";
echo "Circle Volume: " . $circle->volume() . "<br><br>";

$cylinder = new Cylinder(3, 7);
echo "Cylinder Area: " . $cylinder->area() . "<br>";
```

```
echo "Cylinder Volume: " . $cylinder->volume() . "<br>";


// Output:
// Rectangle Area: 15
// Rectangle Volume: 0
// Circle Area: 50.24
// Circle Volume: 0
// Cylinder Area: 188.4
// Cylinder Volume: 63.72
// This code defines an abstract class Shape with two abstract methods: area() and volume().
// It then implements three concrete classes: Rectangle, Circle, and Cylinder, each providing their own implementations of the area() and volume() methods. The example usage demonstrates how to create instances of these classes and call their methods to calculate area and volume.
// The output shows the calculated area and volume for each shape.
// The Rectangle and Circle classes return 0 for volume since they are 2D shapes, while the Cylinder class calculates both area and volume based on its dimensions.
?>
```

b) Create student table as follows: Student(sno, sname, standard, Marks, per).

Write AJAX script to select the student name and print the student's details of particular standard. **[Marks 25]**

```
<?php


// 🛠 Step 1: Open PHPMyAdmin
// Open your browser and go to:
// http://localhost/phpmyadmin
// This will open the PHPMyAdmin interface where you can manage your MySQL databases.


// 🛠 Step 2: Create a New Database
// Click on "New" in the left sidebar.
// In the "Database name" field,
// type:
```

// student_db

// Click "Create".

// 🛠 Step 3: Create a Table student

// After creating the database, click on student_db from the sidebar.

// Under "Create table", type:

// Table name: student

// Number of columns: 5

// Click "Go".

// 🛠 Step 4: Define the Table Columns

// Column Name Type   Length Extra

// sno INT   PRIMARY KEY

// sname   VARCHAR 50

// standard VARCHAR 10

// marks   INT

// per FLOAT

// Make sure to set sno as PRIMARY KEY.

// Then click Save

// -- Insert sample data

// INSERT INTO student VALUES

// (1, 'Amit', '10', 450, 90),

// (2, 'Neha', '10', 400, 80),

// (3, 'Ravi', '9', 420, 84),

// (4, 'Sneha', '9', 460, 92);

// now create 2 files

// index.html

// 2.php

```
// Database connection
$conn = mysqli_connect("localhost", "root", "", "student_db");

if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}

if (isset($_GET['sname'])) {
   $sname = $_GET['sname'];

   $sql = "SELECT * FROM student WHERE sname='$sname'";
   $result = mysqli_query($conn, $sql);

   if (mysqli_num_rows($result) > 0) {
      echo "<h3>Student Details:</h3>";
      while ($row = mysqli_fetch_assoc($result)) {
         echo "S.No: " . $row['sno'] . "<br>";
         echo "Name: " . $row['sname'] . "<br>";
         echo "Standard: " . $row['standard'] . "<br>";
         echo "Marks: " . $row['marks'] . "<br>";
         echo "Percentage: " . $row['per'] . "%<br>";
      }
   } else {
      echo "No student found.";
   }
}
?>
```

Slips 6

Q. 1)

a) Write a C++ program to create a class date with data members day, month and year. Use default and parameterized constructor to initialize date and display date in dd-mon-yyyy format (Example: input: 04-01-2021 Output : 04-jan-2021) **[Marks 15]**

```
#include <iostream>
using namespace std;

class Date
{
private:
   int day, month, year;

public:
   // Default constructor
   Date()
   {
      day = 1;
      month = 1;
      year = 2000;
   }

   // Parameterized constructor
   Date(int d, int m, int y)
   {
      day = d;
      month = m;
      year = y;
   }

   // Function to display date in dd-mon-yyyy format
   void display()
   {
      string months[] = {"jan", "feb", "mar", "apr", "may", "jun",
                         "jul", "aug", "sep", "oct", "nov", "dec"};
      if (month >= 1 && month <= 12)
```

```
        {
            // Adding 0 if day < 10
            if (day < 10)
                cout << "0";
            cout << day << "-";
            cout << months[month - 1] << "-";
            cout << year << endl;
        }
        else
        {
            cout << "Invalid month!" << endl;
        }
    }
};

int main()
{
    // Using default constructor
    Date d1;
    cout << "Default date: ";
    d1.display();

    // Using parameterized constructor
    int d, m, y;
    cout << "\nEnter date (dd mm yyyy): ";
    cin >> d >> m >> y;

    Date d2(d, m, y);
    cout << "Formatted date: ";
    d2.display();
```

```
    return 0;
}

// Output:
// Default date: 01-jan-2000
// Enter date (dd mm yyyy): 15 8 2023
// Formatted date: 15-aug-2023
// Note: The program assumes valid input for the date. In a real-world scenario, you would want to add error checking for invalid dates.
```

b) Write a C++ Program to find volume of cylinder, cone and sphere. **(Use function overloading)**

[Marks 25]

```
#include <iostream>
#define PI 3.1416
using namespace std;

// Volume of Cylinder: π * r^2 * h
float volume(float radius, float height) {
    return PI * radius * radius * height;
}

// Volume of Cone: (1/3) * π * r^2 * h
float volume(float radius, float height, int type) {
    return (PI * radius * radius * height) / 3;
}

// Volume of Sphere: (4/3) * π * r^3
float volume(float radius) {
    return (4.0 / 3) * PI * radius * radius * radius;
}

int main() {
```

```
    float r, h;

    cout << "--- Volume Calculator ---" << endl;

    // Cylinder
    cout << "\nEnter radius and height of cylinder: ";
    cin >> r >> h;
    cout << "Volume of Cylinder = " << volume(r, h) << endl;

    // Cone
    cout << "\nEnter radius and height of cone: ";
    cin >> r >> h;
    cout << "Volume of Cone = " << volume(r, h, 1) << endl;

    // Sphere
    cout << "\nEnter radius of sphere: ";
    cin >> r;
    cout << "Volume of Sphere = " << volume(r) << endl;

    return 0;
}
```

Q. 2)

a) Write a PHP program to create class circle having radius data member and two member functions find circumference() and find area(). Display area and circumference depending on user's preference. **[Marks l5]**

```
<!DOCTYPE html>
<html>

<head>
    <title>Circle Area and Circumference</title>
```

```
</head>

<body>
  <h2>Circle Area and Circumference</h2>
  <form method="post">
    Enter Radius: <input type="text" name="radius" required><br><br>
    <input type="checkbox" name="calc[]" value="area"> Find Area<br>
    <input type="checkbox" name="calc[]" value="circ"> Find Circumference<br><br>
    <input type="submit" value="Calculate">
  </form>

  <?php
  class Circle
  {
    private $radius;

    function __construct($r)
    {
      $this->radius = $r;
    }

    function findArea()
    {
      return 3.1416 * $this->radius * $this->radius;
    }

    function findCircumference()
    {
      return 2 * 3.1416 * $this->radius;
    }
  }
```

```
    if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $radius = $_POST["radius"];
        $circle = new Circle($radius);

        if (!empty($_POST['calc'])) {
            echo "<h3>Results:</h3>";
            foreach ($_POST['calc'] as $value) {
                if ($value == "area") {
                    echo "Area: " . $circle->findArea() . "<br>";
                }
                if ($value == "circ") {
                    echo "Circumference: " . $circle->findCircumference() . "<br>";
                }
            }
        } else {
            echo "<p>Please select at least one option.</p>";
        }
    }
    ?>
</body>

</html>
```

b) Write a PHP program to convert temperature Fahrenheit to Celsius using sticky form. **[Marks 25]**

```
<!DOCTYPE html>
<html>

<head>
    <title>Fahrenheit to Celsius Converter</title>
</head>
```

```
<body>
   <h2>Fahrenheit to Celsius Converter</h2>

   <form method="post">
      Enter Temperature in Fahrenheit:
      <input type="text" name="fahrenheit" value="<?php if (isset($_POST['fahrenheit']))
         echo $_POST['fahrenheit']; ?>"
         required>
      <input type="submit" value="Convert">
   </form>

   <?php
   if ($_SERVER["REQUEST_METHOD"] == "POST") {
      $f = $_POST['fahrenheit'];
      if (is_numeric($f)) {
         $c = ($f - 32) * 5 / 9;
         echo "<p><strong>$f °F = " . round($c, 2) . " °C</strong></p>";
      } else {
         echo "<p>Please enter a valid number.</p>";
      }
   }
   ?>
</body>

</html>
```

Slips 7

Q. 1)

a) Write a c++ program to create a class product with data members product_id, product_name, qty, price. Write necessary function to accept and display product information also display number of objects created for product class. **[Marks 15]**

```
#include <iostream>
using namespace std;

class Product {
private:
    int product_id;
    char product_name[30];
    int qty;
    float price;
    static int count;  // static variable to count objects

public:
    // Constructor
    Product() {
        count++;
    }

    // Function to accept product details
    void accept() {
        cout << "Enter Product ID: ";
        cin >> product_id;
        cout << "Enter Product Name: ";
        cin >> product_name;
        cout << "Enter Quantity: ";
        cin >> qty;
        cout << "Enter Price: ";
        cin >> price;
    }

    // Function to display product details
    void display() {
```

```
        cout << "\nProduct ID: " << product_id;
        cout << "\nProduct Name: " << product_name;
        cout << "\nQuantity: " << qty;
        cout << "\nPrice: " << price << endl;
    }

    // Static function to display object count
    static void showCount() {
        cout << "\nTotal number of Product objects created: " << count << endl;
    }
};

// Definition of static member
int Product::count = 0;

int main() {
    int n;
    cout << "Enter number of products: ";
    cin >> n;

    Product p[n];  // array of objects

    for (int i = 0; i < n; i++) {
        cout << "\nEnter details for product " << i + 1 << ":\n";
        p[i].accept();
    }

    cout << "\n--- Product Details ---";
    for (int i = 0; i < n; i++) {
        p[i].display();
    }
```

```
    // Show number of objects created
    Product::showCount();

    return 0;
}
```

b) Create a C++ class person with data members person_name, mobile_number, age, city. Write necessary member functions for the following.

1. accept person details
2. display person details
3. search person detail with mobile number

[Marks 25]

```
#include <iostream>
#include <string>
using namespace std;

class Person {
private:
    string person_name;
    string mobile_number;
    int age;
    string city;

public:
    void accept() {
        cout << "Enter Name: ";
        cin >> person_name;
        cout << "Enter Mobile Number: ";
        cin >> mobile_number;
        cout << "Enter Age: ";
```

```
        cin >> age;
        cout << "Enter City: ";
        cin >> city;
    }

    void display() {
        cout << "\nName: " << person_name;
        cout << "\nMobile: " << mobile_number;
        cout << "\nAge: " << age;
        cout << "\nCity: " << city << endl;
    }

    bool searchByMobile(const string& number) {
        return mobile_number == number;
    }
};

int main() {
    int n;
    cout << "Enter number of persons: ";
    cin >> n;

    Person p[n];

    for (int i = 0; i < n; i++) {
        cout << "\nEnter details for person " << i + 1 << ":\n";
        p[i].accept();
    }

    cout << "\n--- All Person Details ---";
    for (int i = 0; i < n; i++) {
```

```
        p[i].display();
    }

    string search_mobile;
    cout << "\nEnter Mobile Number to search: ";
    cin >> search_mobile;

    bool found = false;
    for (int i = 0; i < n; i++) {
        if (p[i].searchByMobile(search_mobile)) {
            cout << "\n--- Person Found ---\n";
            p[i].display();
            found = true;
            break;
        }
    }

    if (!found) {
        cout << "\nPerson with mobile number " << search_mobile << " not found.\n";
    }

    return 0;
}

// Output:
// Enter number of persons: 2

// Enter details for person 1:
// Enter Name: Ravi
// Enter Mobile Number: 9876543210
// Enter Age: 25
```

// Enter City: Mumbai

// Enter details for person 2:

// Enter Name: Anu

// Enter Mobile Number: 9998887776

// Enter Age: 22

// Enter City: Pune

// --- All Person Details ---

// Name: Ravi

// Mobile: 9876543210

// Age: 25

// City: Mumbai

// Name: Anu

// Mobile: 9998887776

// Age: 22

// City: Pune

// Enter Mobile Number to search: 9998887776

// --- Person Found ---

// Name: Anu

// Mobile: 9998887776

// Age: 22

// City: Pune

Q. 2)

a) Write PHP program to select list of subjects front list box and displays the selected subject information on next page. (Use sticky Multivalued parameter). **[Marks l5]**

<!DOCTYPE html>

```
<html>

<head>
  <title>Select Subjects</title>
</head>

<body>
  <h2>Select Your Subjects</h2>

  <form method="POST" action="">
    <select name="subjects[]" multiple size="5">
      <option value="Math" <?php if (isset($_POST['subjects']) && in_array("Math", $_POST['subjects']))
        echo "selected"; ?>>Math</option>
      <option value="Science" <?php if (isset($_POST['subjects']) && in_array("Science", $_POST['subjects']))
        echo "selected"; ?>>Science</option>
      <option value="History" <?php if (isset($_POST['subjects']) && in_array("History", $_POST['subjects']))
        echo "selected"; ?>>History</option>
      <option value="English" <?php if (isset($_POST['subjects']) && in_array("English", $_POST['subjects']))
        echo "selected"; ?>>English</option>
      <option value="Computer" <?php if (isset($_POST['subjects']) && in_array("Computer", $_POST['subjects']))
        echo "selected"; ?>>Computer</option>
    </select>
    <br><br>
    <input type="submit" value="Show Subjects">
  </form>

  <?php
  if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_POST['subjects'])) {
```

```
        echo "<h3>You selected:</h3><ul>";
        foreach ($_POST['subjects'] as $subject) {
            echo "<li>" . ($subject) . "</li>";
        }
        echo "</ul>";
    }
    ?>
</body>

</html>
```

b) Write a PHP script using AJAX concept, to give hint to user when he/she type city name in the text field. **[Marks 25]**

```
<?php
// Sample city names
$cities = [ "Mumbai", "Delhi", "Pune", "Bengaluru", "Hyderabad", "Chennai", "Kolkata", "Surat", "Ahmedabad", "Jaipur" ];

$q = isset($_GET["q"]) ? strtolower($_GET["q"]) : "";

$hint = "";

if ($q !== "") {
    foreach ($cities as $city) {
        if (stristr($city, $q)) {
            $hint .= "<div class='hint-item'>" . htmlspecialchars($city) . "</div>";
        }
    }
}

echo $hint === "" ? "<div class='hint-item'>No match found</div>" : $hint;
?>
```

Slips 8

Slips 3 ka copy h cpp Q1 a and b

Slips 7 ka copy h php Q2 a and b

Slips 9

Q. 1)

a) Create a C++ class time with data members Hours, Minutes and Seconds. Write necessary member functions for the following (Use Object as arguments)

1. to accept time

2. to display time in hh:mm:ss

3. to find difference between two time and display difference in hh:mm:ss

```
#include <iostream>
#include <iomanip>
using namespace std;

class Time {
   int hours, minutes, seconds;

public:
   // Accept time
   void acceptTime() {
      cout << "Enter hours: ";
      cin >> hours;
      cout << "Enter minutes: ";
      cin >> minutes;
      cout << "Enter seconds: ";
      cin >> seconds;
   }

   // Display time
   void displayTime() {
      cout << setfill('0') << setw(2) << hours << ":"
```

```
            << setw(2) << minutes << ":"
            << setw(2) << seconds << endl;
    }

    // Calculate time difference
    void timeDifference(Time t1, Time t2) {
        int s1 = t1.hours * 3600 + t1.minutes * 60 + t1.seconds;
        int s2 = t2.hours * 3600 + t2.minutes * 60 + t2.seconds;
        int diff = abs(s1 - s2);

        hours = diff / 3600;
        diff %= 3600;
        minutes = diff / 60;
        seconds = diff % 60;
    }
};

int main() {
    Time t1, t2, diff;

    cout << "Enter first time:\n";
    t1.acceptTime();

    cout << "\nEnter second time:\n";
    t2.acceptTime();

    cout << "\nFirst Time: ";
    t1.displayTime();

    cout << "Second Time: ";
    t2.displayTime();
```

```
    diff.timeDifference(t1, t2);
    cout << "Time Difference: ";
    diff.displayTime();

    return 0;
}
```

Q B copy h slips 4 ka cpp

Q2 a copy h slips 4 ka php

b) Write a Ajax program to get book details from XML file when user select book name. Create XML file to store details of book(name, author, year and price)

```
<?php
if (isset($_GET['book'])) {
    $selectedBook = $_GET['book'];
    $xml = simplexml_load_file("books.xml");

    foreach ($xml->Book as $book) {
        if ((string) $book->Name == $selectedBook) {
            echo "<strong>Book Name:</strong> $book->Name <br>";
            echo "<strong>Author:</strong> $book->Author <br>";
            echo "<strong>Year:</strong> $book->Year <br>";
            echo "<strong>Price:</strong> ₹$book->Price";
            break;
        }
    }
}
?>
```

Book.xml

```
<Books>
<Book>
```

<Name>PHP Basics</Name>

<Author>John Doe</Author>

<Year>2021</Year>

<Price>299</Price>

</Book>

<Book>

<Name>Learn AJAX</Name>

<Author>Jane Smith</Author>

<Year>2020</Year>

<Price>349</Price>

</Book>

<Book>

<Name>HTML & CSS</Name>

<Author>Mark Brown</Author>

<Year>2019</Year>

<Price>199</Price>

</Book>

</Books>

Slips 10

Q1 a and b copy of slips 5 cpp

a) Create an application that reads “Sports.xml” file into sample XML object. Display attributes and elements. **(Hint : Use simple_xml_load_file() function)** **[Marks l5]**

```
<?php
// Load XML file
$xml = simplexml_load_file("Sports.xml") or die("Error: Cannot load file");

// Loop through each <Game> element
foreach ($xml->Game as $game) {
    echo "<strong>Game:</strong> " . $game['name'] . "<br>";
    echo "Players: " . $game->Players . "<br>";
```

```
    echo "Origin: " . $game->Origin . "<br><br>";
}
?>
```

Sports.xml

```
<Sports>
<Game name="Cricket">
<Players>11</Players>
<Origin>England</Origin>
</Game>
<Game name="Football">
<Players>11</Players>
<Origin>Europe</Origin>
</Game>
<Game name="Tennis">
<Players>2</Players>
<Origin>France</Origin>
</Game>
</Sports>
```

b) Write a simple PHP program which implements Ajax for addition of two numbers. **[Marks 25]**

```
<?php
// Get the numbers from query string
$num1 = $_GET['num1'];
$num2 = $_GET['num2'];

// Perform addition
$sum = $num1 + $num2;

// Return result
echo $sum;
?>
```

Slips 11

a) Write a C++ program to create a class E_Bill with data members Cust_name, Meter_ID, No_of_Units and Total_Charges. Write member function to accept and display customer information by calculating charges. (Rules to calculate electricity board charges)

```
#include <iostream>

using namespace std;


class E_Bill

{

private:

   string cust_name;

   int meter_id;

   int no_of_units;

   float total_charges;


public:

   void accept()

   {

      cout << "Enter Customer Name: ";

      cin >> cust_name;

      cout << "Enter Meter ID: ";

      cin >> meter_id;

      cout << "Enter Number of Units Consumed: ";

      cin >> no_of_units;


      // Charges:

      // 1-100 units: Rs. 1.5/unit

      // 101-200 units: Rs. 2.5/unit

      // 201-300 units: Rs. 3.5/unit

      // above 300: Rs. 5/unit
```

```
        if (no_of_units <= 100)
            total_charges = no_of_units * 1.5;
        else if (no_of_units <= 200)
            total_charges = 100 * 1.5 + (no_of_units - 100) * 2.5;
        else if (no_of_units <= 300)
            total_charges = 100 * 1.5 + 100 * 2.5 + (no_of_units - 200) * 3.5;
        else
            total_charges = 100 * 1.5 + 100 * 2.5 + 100 * 3.5 + (no_of_units - 300) * 5.0;
    }

    void display()
    {
        cout << "\n--- Electricity Bill ---" << endl;
        cout << "Customer Name: " << cust_name << endl;
        cout << "Meter ID: " << meter_id << endl;
        cout << "Units Consumed: " << no_of_units << endl;
        cout << "Total Charges: Rs. " << total_charges << endl;
    }
};

int main()
{
    E_Bill bill;
    bill.accept();
    bill.display();
    return 0;
}

// ✅ Rules to Calculate Charges (Example Logic):
// Units    Rate per Unit
// First 100 units  ₹1.5
```

```
// Next 100 units   ₹2.5
// Above 200 units  ₹3.5

// Example Input/Output:

// Enter Customer Name: John
// Enter Meter ID: 101
// Enter Number of Units Consumed: 250

// --- Electricity Bill ---
// Customer Name: John
// Meter ID: 101
// Units Consumed: 250
// Total Charges: ₹575.00
```

b) Create a C++ class visiting staff with data members name, no of subjects, name of subjects[], working hr, total salary. (number of subjects varies for a staff). Write a parameterized constructor to initialize the data members and create an array for name of subjects dynamically. Display visiting staff details by calculating salary. (Assume renumeration 300 per working hour)

```
#include <iostream>
using namespace std;

class VisitingStaff
{
private:
    string name;
    int numSubjects;
    string subjects[10]; // Fixed-size array
    int workingHours;
    float totalSalary;

public:
    // Parameterized Constructor
```

```
    VisitingStaff(string n, int ns, string sub[], int hours)
    {
        name = n;
        numSubjects = ns;
        workingHours = hours;

        for (int i = 0; i < numSubjects; i++)
        {
            subjects[i] = sub[i];
        }

        totalSalary = workingHours * 300; // ₹300/hour
    }

    void display()
    {
        cout << "\n--- Visiting Staff Details ---\n";
        cout << "Name: " << name << endl;
        cout << "Number of Subjects: " << numSubjects << endl;
        cout << "Subjects: ";
        for (int i = 0; i < numSubjects; i++)
        {
            cout << subjects[i];
            if (i < numSubjects - 1)
                cout << ", ";
        }
        cout << "\nWorking Hours: " << workingHours << endl;
        cout << "Total Salary: Rs." << totalSalary << endl;
    }
};
```

```
int main()
{
    string name;
    int numSubjects;
    string subjects[10];
    int hours;

    cout << "Enter Staff Name (one word): ";
    cin >> name;

    cout << "Enter Number of Subjects (max 10): ";
    cin >> numSubjects;

    for (int i = 0; i < numSubjects; i++)
    {
        cout << "Enter Subject " << (i + 1) << ": ";
        cin >> subjects[i];
    }

    cout << "Enter Working Hours: ";
    cin >> hours;

    VisitingStaff staff(name, numSubjects, subjects, hours);
    staff.display();

    return 0;
}
```

a) Write a PHP script to read book. XML and print book details in tabular format using simple XML(Content of book. XML are book_code, book_name, author, year, price)

```
<?php
```

```
$xml = simplexml_load_file("book.xml") or die("Unable to load XML file!");

echo "<h2>Book Details</h2>";
echo "<table border='1' cellpadding='10'>
<tr>
   <th>Book Code</th>
   <th>Book Name</th>
   <th>Author</th>
   <th>Year</th>
   <th>Price</th>
</tr>";

foreach ($xml->book as $book) {
   echo "<tr>";
   echo "<td>" . $book->book_code . "</td>";
   echo "<td>" . $book->book_name . "</td>";
   echo "<td>" . $book->author . "</td>";
   echo "<td>" . $book->year . "</td>";
   echo "<td>₹" . $book->price . "</td>";
   echo "</tr>";
}

echo "</table>";
?>
```

b) Derive a class Rectangle from class Square. Create one more class Triangle. Create an interface with only one method called cal_area(). Implement this interface in all the classes. Include appropriate data members and constructors in all classes. Write a program to accept details of Rectangle, Square and Triangle and display the area.

```
<?php

interface Shape
{
```

```
    public function cal_area();
}

// Base class: Square
class Square implements Shape
{
    protected $side;

    public function __construct($side)
    {
        $this->side = $side;
    }

    public function cal_area()
    {
        return $this->side * $this->side;
    }
}

// Derived class: Rectangle extends Square
class Rectangle extends Square
{
    private $height;

    public function __construct($width, $height)
    {
        parent::__construct($width);  // width is treated as side from Square
        $this->height = $height;
    }

    public function cal_area()
```

```
    {
        return $this->side * $this->height;
    }
}

// Independent class: Triangle
class Triangle implements Shape
{
    private $base;
    private $height;

    public function __construct($base, $height)
    {
        $this->base = $base;
        $this->height = $height;
    }

    public function cal_area()
    {
        return 0.5 * $this->base * $this->height;
    }
}

// ====== Accept and display area ======
echo "<h3>Area Calculation</h3>";

// Rectangle
$rectangle = new Rectangle(5, 8);
echo "Rectangle Area (5 x 8): " . $rectangle->cal_area() . "<br>";

// Square
```

```
$square = new Square(4);
echo "Square Area (4 x 4): " . $square->cal_area() . "<br>";

// Triangle
$triangle = new Triangle(6, 7);
echo "Triangle Area (0.5 x 6 x 7): " . $triangle->cal_area() . "<br>";

?>
```

Slips 12

a) Write a c++ program to create a class Account with data members AccNo, AccType and Balance.write member function to accept and display 'n' account details (Use dynamic memory allocation)

```
#include <iostream>
using namespace std;

class Account
{
   int accNo;
   string accType;
   float balance;

public:
   void accept()
   {
      cout << "Enter Account Number: ";
      cin >> accNo;
      cout << "Enter Account Type: ";
      cin >> accType;
      cout << "Enter Balance: ";
      cin >> balance;
   }
```

```
    void display()
    {
        cout << "Account No: " << accNo
             << ", Type: " << accType
             << ", Balance: " << balance << endl;
    }
};

int main()
{
    int n;
    cout << "Enter how many accounts (max 100): ";
    cin >> n;

    Account acc[100]; // Using fixed size array

    // Accept details
    for (int i = 0; i < n; i++)
    {
        cout << "\nEnter details for Account " << (i + 1) << ":\n";
        acc[i].accept();
    }

    // Display details
    cout << "\n--- Account Details ---\n";
    for (int i = 0; i < n; i++)
    {
        acc[i].display();
    }

    return 0;
```

```
}

// Sample Output:

// Enter how many accounts: 2

// Enter details for Account 1:
// Enter Account Number: 101
// Enter Account Type: Savings
// Enter Balance: 5000

// Enter details for Account 2:
// Enter Account Number: 102
// Enter Account Type: Current
// Enter Balance: 7500

// --- Account Details ---
// Account No: 101, Type: Savings, Balance: 5000
// Account No: 102, Type: Current, Balance: 7500
```

b) Create a C++ class city with data member City_code, City_name, Population. Write necessary member functions for following.

1. To accept details of n cities

2. Display Details of n cities (Use Array of Object and manipulators)

```
#include <iostream>
using namespace std;

class City {
public:
    int code;
    string name;
    int population;
```

```
    void accept() {
        cout << "Enter City Code: ";
        cin >> code;
        cout << "Enter City Name: ";
        cin >> name;
        cout << "Enter Population: ";
        cin >> population;
    }

    void display() {
        cout << "City Code: " << code << endl;
        cout << "City Name: " << name << endl;
        cout << "Population: " << population << endl;
    }
};

int main() {
    int n;
    cout << "Enter number of cities: ";
    cin >> n;

    City cities[100]; // You can store up to 100 cities

    // Accept details
    for (int i = 0; i < n; i++) {
        cout << "\nEnter details for City " << i + 1 << ":\n";
        cities[i].accept();
    }

    // Display details
```

```
    cout << "\n--- City Details ---\n";
    for (int i = 0; i < n; i++) {
        cout << "\nCity " << i + 1 << ":\n";
        cities[i].display();
    }

    return 0;
}
```

a) Write a PHP program to create a class Worker that has data members as Worker_Name, No_of_Days_worked, Pay_Rate. Create and initialize the object using default constructor, Parameterized constructor. Also write necessary member function to calculate and display the salary of worker.

```
<?php

class Worker
{
    public $worker_name;
    public $no_of_days_worked;
    public $pay_rate;

    // Default constructor
    function __construct($name = "Default Worker", $days = 0, $rate = 0)
    {
        $this->worker_name = $name;
        $this->no_of_days_worked = $days;
        $this->pay_rate = $rate;
    }

    // Function to calculate salary
    function calculateSalary()
    {
```

```
        return $this->no_of_days_worked * $this->pay_rate;
    }

    // Function to display worker details
    function displayDetails()
    {
        echo "Worker Name: " . $this->worker_name . "<br>";
        echo "Days Worked: " . $this->no_of_days_worked . "<br>";
        echo "Pay Rate: " . $this->pay_rate . "<br>";
        echo "Total Salary: " . $this->calculateSalary() . "<br><br>";
    }
}

// Using default constructor
$worker1 = new Worker();
$worker1->displayDetails();

// Using parameterized constructor
$worker2 = new Worker("John Doe", 25, 500);
$worker2->displayDetails();

// 📜 Output:

// Worker Name: Default Worker
// Days Worked: 0
// Pay Rate: 0
// Total Salary: 0

// Worker Name: John Doe
// Days Worked: 25
// Pay Rate: 500
```

```
// Total Salary: 12500

?>
```

b) Write a script to create “cricket.xml” file with multiple elements as shown below:

```
<CricketTeam>
 <Team country=”India”>
        <player>        </player>
         <runs>         </runs>
         <wicket>       </wicket>
  </Team>
  </CricketTeam>
```

Write a script to add multiple elements in “cricket.xml” file of category, country=”Australia”.

```
<?php
$xml = new SimpleXMLElement("<CricketTeam></CricketTeam>");

$team = $xml->addChild("Team");
$team->addAttribute("country", "India");

$team->addChild("player", "Virat Kohli");
$team->addChild("runs", "12345");
$team->addChild("wicket", "15");

// Save to file
$xml->asXML("cricket.xml");

echo "Initial cricket.xml created successfully!";
?>
```

Next fie

```
<?php
$xml = simplexml_load_file("cricket.xml");
```

```
// Add new team for Australia
$team = $xml->addChild("Team");
$team->addAttribute("country", "Australia");

$team->addChild("player", "Steve Smith");
$team->addChild("runs", "9500");
$team->addChild("wicket", "8");

// Add another player
$team2 = $xml->addChild("Team");
$team2->addAttribute("country", "Australia");

$team2->addChild("player", "David Warner");
$team2->addChild("runs", "8500");
$team2->addChild("wicket", "2");

// Save back to file
$xml->asXML("cricket.xml");

echo "Australian players added to cricket.xml!";
?>
```

Slips 13

a) Create a C++ class MyArray, which contains single dimensional integer array of a given size. Write a member function to display sum of given array elements. (Use Dynamic Constructor and Destructor)

```
#include <iostream>
using namespace std;

class MyArray
{
```

```
    int arr[100]; // Fixed-size array
    int size;

public:
    // Constructor to accept size and array values
    MyArray(int s)
    {
        size = s;
        cout << "Enter " << size << " elements:\n";
        for (int i = 0; i < size; i++)
        {
            cin >> arr[i];
        }
    }

    // Function to display the sum
    void displaySum()
    {
        int sum = 0;
        for (int i = 0; i < size; i++)
        {
            sum += arr[i];
        }
        cout << "Sum of array elements = " << sum << endl;
    }
};

int main()
{
    int n;
    cout << "Enter size of array: ";
```

```
    cin >> n;

    MyArray a(n);
    a.displaySum();

    return 0;
}
```

b) Create a base class Student with data members Roll No, Name. Derives two classes from it, class Theory with data members M 1, M2, M3, M4 and class Practical w ith data members P1, P2. Class Result (Total Marks, Percentage, Grade) inherits both Theory and Practical classes. (Use concept of Virtual Base Class and protected access specifiers) Write a C++ menu driven program to perform the following functions:

i. Accept Student Information

ii. Display Student Information

iii. Calculate Total marks, Percentage and Grade.

```
#include <iostream>
using namespace std;

// Base class
class Student {
protected:
    int rollNo;
    string name;

public:
    void acceptStudent() {
        cout << "Enter Roll No: ";
        cin >> rollNo;
        cout << "Enter Name: ";
        cin >> name;
    }
```

```
    void displayStudent() {
        cout << "Roll No: " << rollNo << endl;
        cout << "Name: " << name << endl;
    }
};

// Theory class (virtually inherits Student)
class Theory : virtual public Student {
protected:
    int m1, m2, m3, m4;

public:
    void acceptTheory() {
        cout << "Enter 4 theory marks: ";
        cin >> m1 >> m2 >> m3 >> m4;
    }

    void displayTheory() {
        cout << "Theory Marks: " << m1 << " " << m2 << " " << m3 << " " << m4 << endl;
    }

    int theoryTotal() {
        return m1 + m2 + m3 + m4;
    }
};

// Practical class (virtually inherits Student)
class Practical : virtual public Student {
protected:
    int p1, p2;
```

```
public:
    void acceptPractical() {
        cout << "Enter 2 practical marks: ";
        cin >> p1 >> p2;
    }

    void displayPractical() {
        cout << "Practical Marks: " << p1 << " " << p2 << endl;
    }

    int practicalTotal() {
        return p1 + p2;
    }
};

// Result class inherits both Theory and Practical
class Result : public Theory, public Practical {
private:
    int total;
    float percentage;
    char grade;

public:
    void acceptAll() {
        acceptStudent();
        acceptTheory();
        acceptPractical();
    }

    void displayAll() {
        displayStudent();
```

```
        displayTheory();
        displayPractical();
    }

    void calculateResult() {
        total = theoryTotal() + practicalTotal();
        percentage = total / 6.0;

        if (percentage >= 75)
            grade = 'A';
        else if (percentage >= 60)
            grade = 'B';
        else if (percentage >= 50)
            grade = 'C';
        else if (percentage >= 35)
            grade = 'D';
        else
            grade = 'F';

        cout << "Total Marks: " << total << endl;
        cout << "Percentage: " << percentage << "%" << endl;
        cout << "Grade: " << grade << endl;
    }
};

// Main program
int main() {
    Result r;
    int ch;

    do {
```

```
        cout << "\n--- Menu ---\n";
        cout << "1. Accept Student Info\n";
        cout << "2. Display Student Info\n";
        cout << "3. Calculate Result\n";
        cout << "4. Exit\n";
        cout << "Enter choice: ";
        cin >> ch;

        switch (ch) {
        case 1:
            r.acceptAll();
            break;
        case 2:
            r.displayAll();
            break;
        case 3:
            r.calculateResult();
            break;
        case 4:
            cout << "Exiting program.\n";
            break;
        default:
            cout << "Invalid choice!\n";
        }
    } while (ch != 4);

    return 0;
}

// 📝 Example Input:
```

// Enter Roll No: 101

// Enter Name: Raj

// Enter 4 theory marks: 70 75 80 85

// Enter 2 practical marks: 40 45

// 💡 Output:

// Total Marks: 395

// Percentage: 65.8333%

// Grade: B

a) Write a PHP script to read item.XML and print item details in tabular format using simple XML(Content of book. XML are item_code, item_name, quantity, price)

```
<?php

// Load XML file

$xml = simplexml_load_file("item.xml") or die("Error: Cannot load XML file");


echo "<h2>Item Details</h2>";

echo "<table border='1' cellpadding='5' cellspacing='0'>";

echo "<tr>

      <th>Item Code</th>

      <th>Item Name</th>

      <th>Quantity</th>

      <th>Price</th>

    </tr>";


// Loop through each item

foreach ($xml->item as $item) {

   echo "<tr>";

   echo "<td>" . $item->item_code . "</td>";

   echo "<td>" . $item->item_name . "</td>";

   echo "<td>" . $item->quantity . "</td>";
```

```
    echo "<td>" . $item->price . "</td>";
    echo "</tr>";
}

echo "</table>";
?>
```

b) Define an interface which has methods area( ), volume( ). Define constant PI. Create a class cylinder which implements this interface and calculate area and volume. [Marks 25]

```
<?php
// Define an interface
interface Shape
{
    public function area();
    public function volume();
}

// Define constant PI
define("PI", 3.14159);

// Cylinder class implements the interface
class Cylinder implements Shape
{
    private $radius;
    private $height;

    // Constructor
    public function __construct($r, $h)
    {
        $this->radius = $r;
        $this->height = $h;
```

```
    }

    // Calculate area (Surface area)
    public function area()
    {
        $area = 2 * PI * $this->radius * ($this->radius + $this->height);
        echo "Surface Area of Cylinder: " . round($area, 2) . "<br>";
    }

    // Calculate volume
    public function volume()
    {
        $volume = PI * $this->radius * $this->radius * $this->height;
        echo "Volume of Cylinder: " . round($volume, 2) . "<br>";
    }
}

// Example usage
$cyl = new Cylinder(5, 10);
$cyl->area();
$cyl->volume();
?>
```

Slips 14

a) Create a class Mymatrix. Write c C++ program to accept and display a Matrix overload binary '-' operator to calculate substraction of two matrix.

```
#include <iostream>
using namespace std;

class MyMatrix
{
```

```
    int matrix[10][10]; // Fixed size for simplicity
    int rows, cols;

public:
    // Accept matrix from user
    void accept()
    {
        cout << "Enter number of rows and columns: ";
        cin >> rows >> cols;
        cout << "Enter matrix elements:\n";
        for (int i = 0; i < rows; ++i)
            for (int j = 0; j < cols; ++j)
                cin >> matrix[i][j];
    }

    // Display matrix
    void display()
    {
        cout << "Matrix:\n";
        for (int i = 0; i < rows; ++i)
        {
            for (int j = 0; j < cols; ++j)
                cout << matrix[i][j] << "\t";
            cout << endl;
        }
    }

    // Overload '-' operator
    MyMatrix operator-(MyMatrix m)
    {
        MyMatrix result;
```

```
        result.rows = rows;
        result.cols = cols;
        for (int i = 0; i < rows; ++i)
            for (int j = 0; j < cols; ++j)
                result.matrix[i][j] = matrix[i][j] - m.matrix[i][j];
        return result;
    }
};

int main()
{
    MyMatrix m1, m2, result;

    cout << "Enter first matrix:\n";
    m1.accept();

    cout << "Enter second matrix:\n";
    m2.accept();

    result = m1 - m2;

    cout << "\nFirst Matrix:\n";
    m1.display();

    cout << "\nSecond Matrix:\n";
    m2.display();

    cout << "\nSubtraction (First - Second):\n";
    result.display();

    return 0;
```

}

b) Design two base classes Student (S id, Name, C lass) and Competition (C id, C Name). Derive a class Stud Comp(Rank) from it. Write a menu driven program to perform following functions:

i. Accept information.

ii. Display information.

iii. Display Student Details in the ascending order of Rank of a specified competition.

(Use array of objects)

[Marks 25]

```
#include <iostream>
#include <algorithm>
using namespace std;

class Student {
protected:
   int sid;
   string name;
   string cls;

public:
   void acceptStudent() {
      cout << "Enter Student ID: ";
      cin >> sid;
      cout << "Enter Name: ";
      cin >> name;
      cout << "Enter Class: ";
      cin >> cls;
   }

   void displayStudent() {
      cout << sid << "\t" << name << "\t" << cls << "\t";
   }
```

```
    int getSid() {
        return sid;
    }
};

class Competition {
protected:
    int cid;
    string cname;

public:
    void acceptCompetition() {
        cout << "Enter Competition ID: ";
        cin >> cid;
        cout << "Enter Competition Name: ";
        cin >> cname;
    }

    void displayCompetition() {
        cout << cid << "\t" << cname << "\t";
    }

    int getCid() {
        return cid;
    }
};

class StudComp : public Student, public Competition {
    int rank;
```

```
public:
    void acceptAll() {
        acceptStudent();
        acceptCompetition();
        cout << "Enter Rank: ";
        cin >> rank;
    }

    void displayAll() {
        displayStudent();
        displayCompetition();
        cout << rank << endl;
    }

    int getRank() {
        return rank;
    }
};

int main() {
    StudComp arr[50];
    int n = 0, ch, compId;

    while (1) {
        cout << "\n--- MENU ---\n";
        cout << "1. Accept Information\n";
        cout << "2. Display Information\n";
        cout << "3. Display Students in Ascending Order of Rank for a Competition\n";
        cout << "4. Exit\n";
        cout << "Enter your choice: ";
        cin >> ch;
```

```
switch (ch) {
case 1:
    cout << "Enter number of students: ";
    cin >> n;
    for (int i = 0; i < n; i++) {
        cout << "\nEnter details for student " << i + 1 << ":\n";
        arr[i].acceptAll();
    }
    break;

case 2:
    cout << "\nStudent ID\tName\tClass\tComp ID\tComp Name\tRank\n";
    for (int i = 0; i < n; i++) {
        arr[i].displayAll();
    }
    break;

case 3:
    cout << "Enter Competition ID to search: ";
    cin >> compId;

    // Create a temporary array for matching competition
    StudComp temp[50];
    int k = 0;

    for (int i = 0; i < n; i++) {
        if (arr[i].getCid() == compId) {
            temp[k++] = arr[i];
        }
    }
```

```
        if (k == 0) {
            cout << "No students found for this competition.\n";
            break;
        }

        // Sort based on rank
        for (int i = 0; i < k - 1; i++) {
            for (int j = i + 1; j < k; j++) {
                if (temp[i].getRank() > temp[j].getRank()) {
                    swap(temp[i], temp[j]);
                }
            }
        }

        cout << "\nSorted Students by Rank:\n";
        cout << "Student ID\tName\tClass\tComp ID\tComp Name\tRank\n";
        for (int i = 0; i < k; i++) {
            temp[i].displayAll();
        }

        break;

    case 4:
        return 0;

    default:
        cout << "Invalid choice!\n";
    }
}
```

```
    return 0;
}
```

a) Write PHP program accept name, select your cities you would like to visit and display selected information on page. (Use multi-valued parameter),.

```
<!DOCTYPE html>
<html>
<head>
    <title>Visit Cities</title>
</head>
<body>

<?php
$name = "";
$cities = [];

if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = $_POST['name'];
    if (isset($_POST['cities'])) {
        $cities = $_POST['cities'];
    }

    echo "<h2>Hello, $name!</h2>";
    echo "<h3>You selected the following cities to visit:</h3><ul>";
    foreach ($cities as $city) {
        echo "<li>$city</li>";
    }
    echo "</ul><hr>";
}
?>

<form method="post" action="">
    <label>Enter Your Name:</label>
    <input type="text" name="name" value="<?php echo htmlspecialchars($name); ?>"
required><br><br>

    <label>Select Cities You Would Like to Visit:</label><br>
    <input type="checkbox" name="cities[]" value="Mumbai"
        <?php if (in_array("Mumbai", $cities)) echo "checked"; ?>> Mumbai<br>
    <input type="checkbox" name="cities[]" value="Delhi"
        <?php if (in_array("Delhi", $cities)) echo "checked"; ?>> Delhi<br>
    <input type="checkbox" name="cities[]" value="Goa"
        <?php if (in_array("Goa", $cities)) echo "checked"; ?>> Goa<br>
    <input type="checkbox" name="cities[]" value="Jaipur"
        <?php if (in_array("Jaipur", $cities)) echo "checked"; ?>> Jaipur<br>
    <input type="checkbox" name="cities[]" value="Kolkata"
```

```
    <?php if (in_array("Kolkata", $cities)) echo "checked"; ?>> Kolkata<br><br>

  <input type="submit" value="Submit">
</form>

</body>
</html>
```

b) Write Ajax program to fetch suggestions when is user is typing in a textbox. (eg like google suggestions.Hint create array of suggestions and matching string will be displayed)

```
<?php
// Sample array of suggestions
$suggestions = [ "apple", "banana", "cherry", "date", "dragonfruit", "grape", "guava", "kiwi",
"lemon", "mango", "orange", "papaya", "peach", "pineapple", "strawberry", "watermelon" ];

$q = strtolower($_GET['q'] ?? "");
$result = "";

if ($q !== "") {
  foreach ($suggestions as $item) {
    if (str_starts_with($item, $q)) {
      $result .= $item . "<br>";
    }
  }
}

echo $result === "" ? "No suggestions found." : $result;
?>
```

Slips 15

a) Create a C++ Class employee with data members with data members Emp_id, Emp_name, Company_name and salary. Write member functions to accept and display employee information. Design user defined manipulators to print salary.

```
#include <iostream>

using namespace std;


class Employee {

  int Emp_id;

  string Emp_name;

  string Company_name;

  float salary;
```

```
public:
    void accept() {
        cout << "Enter Employee ID: ";
        cin >> Emp_id;
        cout << "Enter Employee Name: ";
        cin >> Emp_name;
        cout << "Enter Company Name: ";
        cin >> Company_name;
        cout << "Enter Salary: ";
        cin >> salary;
    }

    void display() {
        cout << "\n--- Employee Details ---\n";
        cout << "ID: " << Emp_id << endl;
        cout << "Name: " << Emp_name << endl;
        cout << "Company: " << Company_name << endl;
        cout << "Salary: Rs. " << salary << endl;
    }
};

int main() {
    Employee emp;
    emp.accept();
    emp.display();
    return 0;
}
```

b) Write a C++ program to accept radius of a circle. Calculate and display diameter, circumference as well as area of circle. (Use inline function)

```
#include <iostream>
using namespace std;

const float PI = 3.14159;
```

```
class Circle {
   float radius;

public:
   void accept() {
      cout << "Enter Radius of Circle: ";
      cin >> radius;
   }

   inline float diameter() {
      return 2 * radius;
   }

   inline float circumference() {
      return 2 * PI * radius;
   }

   inline float area() {
      return PI * radius * radius;
   }

   void display() {
      cout << "\n--- Circle Details ---\n";
      cout << "Diameter: " << diameter() << endl;
      cout << "Circumference: " << circumference() << endl;
      cout << "Area: " << area() << endl;
   }
};

int main() {
   Circle c;
   c.accept();
   c.display();
   return 0;
}
```

a) Write a PHP program to accept two strings from user and check whether entered strings are matching or not. (Use sticky form concept).

```
<!DOCTYPE html>
<html>
<head>
   <title>String Match Checker</title>
</head>
<body>

<h2>Check if Two Strings Match</h2>

<form method="post">
```

```
   Enter First String: <input type="text" name="str1" value="<?php
if(isset($_POST['str1'])) echo $_POST['str1']; ?>"><br><br>
   Enter Second String: <input type="text" name="str2" value="<?php
if(isset($_POST['str2'])) echo $_POST['str2']; ?>"><br><br>
   <input type="submit" name="check" value="Check">
</form>

<?php
if (isset($_POST['check'])) {
   $str1 = $_POST['str1'];
   $str2 = $_POST['str2'];

   if ($str1 === $str2) {
      echo "<h3>✅ Strings Match</h3>";
   } else {
      echo "<h3>❌ Strings Do Not Match</h3>";
   }
}
?>

</body>
</html>
```

b) Write Ajax program to get player details from XML file when user select a player name. Create XML file for storing details of player (Country, player name, wickets, runs). **[Marks 25]**

```
<?php
// Create the XML object from the XML file
$xml = simplexml_load_file('players.xml') or die("Error: Cannot create object");

// Get the 'name' parameter from the URL
$playerName = $_GET['name'];

// Initialize a flag to indicate if the player is found
$found = false;
$playerDetails = "";

// Loop through the players in the XML file
foreach ($xml->player as $player) {
   if ((string) $player->name == $playerName) {
      // Create the HTML table with player details
      $playerDetails = "
         <table border='1'>
            <tr><th>Country</th><td>" . (string) $player->country . "</td></tr>
            <tr><th>Name</th><td>" . (string) $player->name . "</td></tr>
            <tr><th>Wickets</th><td>" . (int) $player->wickets . "</td></tr>
            <tr><th>Runs</th><td>" . (int) $player->runs . "</td></tr>
         </table>
```

```
        ";
        $found = true;
        break;
    }
}

// If player not found, display a message
if (!$found) {
    $playerDetails = "Player not found!";
}

// Return the HTML content
echo $playerDetails;
?>
```

Slips 16

a) Create a C++ Class student with data members rollno, name ,class , percentage. Accept two students information and display information of student having maximum percentage. (Use this pointer)

```
#include <iostream>
using namespace std;

class Student {
private:
    int rollno;
    string name;
    string student_class;
    float percentage;

public:
    // Constructor to initialize data members
    Student(int r, string n, string c, float p) {
        rollno = r;
        name = n;
        student_class = c;
        percentage = p;
    }

    // Function to display student information
    void display() {
        cout << "Roll No: " << rollno << endl;
        cout << "Name: " << name << endl;
        cout << "Class: " << student_class << endl;
        cout << "Percentage: " << percentage << "%" << endl;
    }

    // Function to compare two students' percentage and return the one with the higher percentage
```

```
    Student* max_percentage(Student* otherStudent) {
        if (this->percentage > otherStudent->percentage)
            return this; // Return the current student (this pointer)
        else
            return otherStudent; // Return the other student
    }
};

int main() {
    // Accepting information for two students
    int rollno1, rollno2;
    string name1, name2, class1, class2;
    float percentage1, percentage2;

    // Input student 1 information
    cout << "Enter details for Student 1:" << endl;
    cout << "Roll No: ";
    cin >> rollno1;
    cout << "Name: ";
    cin >> name1;
    cout << "Class: ";
    cin >> class1;
    cout << "Percentage: ";
    cin >> percentage1;

    // Input student 2 information
    cout << "\nEnter details for Student 2:" << endl;
    cout << "Roll No: ";
    cin >> rollno2;
    cout << "Name: ";
    cin >> name2;
    cout << "Class: ";
    cin >> class2;
    cout << "Percentage: ";
    cin >> percentage2;

    // Creating two Student objects
    Student student1(rollno1, name1, class1, percentage1);
    Student student2(rollno2, name2, class2, percentage2);

    // Compare the two students and get the one with the highest percentage
    Student* topStudent = student1.max_percentage(&student2);

    // Display the student with the maximum percentage
    cout << "\nStudent with Maximum Percentage:" << endl;
    topStudent->display();

    return 0;
```

}

b) Write a C++ class number with integer data member. Write necessary member function to overload the operator unary pre and post increment '++'.

[Marks 25]

```
#include <iostream>
using namespace std;

class Number {
private:
   int value;  // Data member to store the integer value

public:
   // Constructor to initialize value
   Number(int v) {
      value = v;
   }

   // Function to display the value
   void display() {
      cout << "Value: " << value << endl;
   }

   // Pre-increment operator overload
   Number& operator++() {
      value++;  // Increment value
      return *this;  // Return the current object
   }

   // Post-increment operator overload
   Number operator++(int) {
      Number temp = *this;  // Make a copy of the current object
      value++;  // Increment the value
      return temp;  // Return the copy (before increment)
   }
};

int main() {
   Number num(5);  // Create a Number object with value 5

   cout << "Initial value: ";
   num.display();

   // Pre-increment
   ++num;
   cout << "After pre-increment: ";
   num.display();
```

```
    // Post-increment
    num++;
    cout << "After post-increment: ";
    num.display();

    return 0;
}
```

Slips 15 ka Q2 a&b both copy h slips 16 m