Assignment 05

5.1 An educational institution wishes to maintain a database of its employees. The database is divided into a number of classes whose hierarchical relationships are shown in Fig. 2.

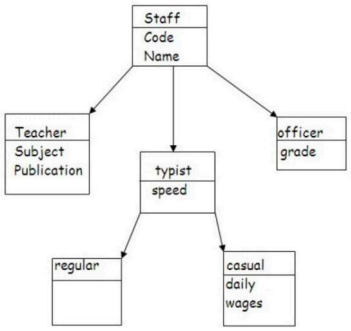

Figure 2: Class Relationships

The figure also shows the minimum information required for each class. Specify all the classes and define functions to create the database and retrieve individual information as and when required.

✅ 5.1: Educational Institution Employee Database (Inheritance)

🔹 **Class Hierarchy:**

Assuming the hierarchy based on the description:

- Staff (Base)
  - Teacher
  - Typist
    - Regular
    - Casual
  - Officer

🔸 **Code:**

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

class Staff {
protected:
    int code;
    string name;
```

```
public:
    virtual void getData() {
        cout << "Enter code and name: ";
        cin >> code;
        cin.ignore();
        getline(cin, name);
    }
    virtual void display() const {
        cout << "Code: " << code << ", Name: " << name << endl;
    }
};

class Teacher : public Staff {
    string subject, publication;
public:
    void getData() override {
        Staff::getData();
        cout << "Enter subject and publication: ";
        getline(cin, subject);
        getline(cin, publication);
    }
    void display() const override {
        Staff::display();
        cout << "Subject: " << subject << ", Publication: " <<
publication << endl;
    }
};

class Officer : public Staff {
    string grade;
public:
    void getData() override {
        Staff::getData();
        cout << "Enter grade: ";
        getline(cin, grade);
    }
    void display() const override {
```

```
        Staff::display();
        cout << "Grade: " << grade << endl;
    }
};

class Typist : public Staff {
protected:
    int speed;
public:
    void getData() override {
        Staff::getData();
        cout << "Enter typing speed: ";
        cin >> speed;
        cin.ignore();
    }
    void display() const override {
        Staff::display();
        cout << "Typing Speed: " << speed << " wpm" << endl;
    }
};

class Regular : public Typist {
public:
    void display() const override {
        Typist::display();
        cout << "Type: Regular\n";
    }
};

class Casual : public Typist {
    float wage;
public:
    void getData() override {
        Typist::getData();
        cout << "Enter daily wage: ";
        cin >> wage;
        cin.ignore();
```

```
    }
    void display() const override {
        Typist::display();
        cout << "Type: Casual, Wage: Rs. " << wage << endl;
    }
};

int main() {
    Teacher t;
    Officer o;
    Regular r;
    Casual c;

    cout << "\nEnter Teacher Details:\n"; t.getData();
    cout << "\nEnter Officer Details:\n"; o.getData();
    cout << "\nEnter Regular Typist Details:\n"; r.getData();
    cout << "\nEnter Casual Typist Details:\n"; c.getData();

    cout << "\n--- Employee Records ---\n";
    t.display();
    o.display();
    r.display();
    c.display();

    return 0;
}
```

- **Sample Output:**

```
Enter Teacher Details:
Enter code and name: 101
Alice Smith
Enter subject and publication: Mathematics
Oxford Press

Enter Officer Details:
Enter code and name: 102
John Doe
```

```
Enter grade: A

Enter Regular Typist Details:
Enter code and name: 103
Priya Mehta
Enter typing speed: 50

Enter Casual Typist Details:
Enter code and name: 104
Arun Kumar
Enter typing speed: 40
Enter daily wage: 500

--- Employee Records ---
Code: 101, Name: Alice Smith
Subject: Mathematics, Publication: Oxford Press
Code: 102, Name: John Doe
Grade: A
Code: 103, Name: Priya Mehta
Typing Speed: 50 wpm
Type: Regular
Code: 104, Name: Arun Kumar
Typing Speed: 40 wpm
Type: Casual, Wage: Rs. 500
```

5.2 Create a base class called shape. Use this class to store two double type values that could be used to compute the area of figures. Derive two specific classes called triangle and rectangle from the base shape. Add to the base class, a member function get data() to initialize base class data members and another member function display area() to compute and display the area of figures. Make display area() as a virtual function and redefine this function in the derived classes to suit their requirements.

Using these three classes, design a program that will accept dimensions of a triangle or a rectangle interactively, and display the area.

Remember the two values given as input will be treated as lengths of two sides in the case of rectangles and as base and height in the case of triangles, and used as follows:

Area of rectangle = x * y

Area of triangle = ½ * x * y

✅ 5.2: Shape, Triangle, and Rectangle using Virtual Functions

🔸 Code:

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

class Shape {
protected:
    double x, y;
public:
    void getData() {
        cout << "Enter two dimensions: ";
        cin >> x >> y;
    }
    virtual void displayArea() const = 0; // pure virtual
};

class Triangle : public Shape {
public:
    void displayArea() const override {
        cout << "Triangle Area: " << 0.5 * x * y << endl;
    }
};

class Rectangle : public Shape {
public:
    void displayArea() const override {
        cout << "Rectangle Area: " << x * y << endl;
    }
};
```

```
int main() {
    Shape *shape;
    int choice;
    cout << "1. Triangle\n2. Rectangle\nEnter choice: ";
    cin >> choice;

    if (choice == 1) shape = new Triangle();
    else shape = new Rectangle();

    shape->getData();
    shape->displayArea();

    delete shape;
    return 0;
}
```

🔸 **Sample Output:**

```
1. Triangle
2. Rectangle
Enter choice: 1
Enter two dimensions: 10 5
Triangle Area: 25
```