Simple Calculator: Create a program that prompts the user to enter two numbers and perform basic arithmetic operations such as addition, subtraction, multiplication, and division. Use variables to store the user inputs and the results of the calculations.

Temperature Conversion: Build a program that converts temperature values between Celsius and Fahrenheit. Prompt the user to enter a temperature in one unit and display the converted value in the other unit. Utilize variables to store the input and output values.

Circle Properties: Design a program that calculates and displays various properties of a circle, such as its radius, diameter, circumference, and area. Prompt the user to enter the radius, and use variables to store the calculations.

Grade Calculator: Develop a program that calculates the average grade of a student based on individual test scores. Prompt the user to enter the number of tests and the corresponding scores. Use variables to store the input values and calculate the average.

BMI Calculator: Create a program that calculates the Body Mass Index (BMI) based on user input for weight and height. Display the calculated BMI along with an appropriate message indicating the weight category. Utilize variables to store the weight, height, and BMI values.

Rectangle Area and Perimeter: Build a program that prompts the user to enter the length and width of a rectangle. Calculate and display the area and perimeter of the rectangle using variables to store the input and calculated values.

Currency Conversion: Design a program that converts an amount of money from one currency to another. Prompt the user to enter the amount and the exchange rate. Use variables to store the input values and the converted amount.

Time Conversion: Develop a program that converts time from seconds to hours, minutes, and seconds. Prompt the user to enter the time in seconds, and use variables to store the calculated values for hours, minutes, and remaining seconds.

Create a Class Hierarchy: Define a class hierarchy representing different types of vehicles. Start with a base class "Vehicle" and create derived classes such as "Car," "Motorcycle," and "Truck." Add relevant attributes and methods to each class that capture their unique characteristics.

Implement Polymorphism: Extend the vehicle hierarchy assignment by implementing polymorphic behavior. Create a method, such as "drive," in the base class, and override it in the derived classes to

provide specific implementations. Demonstrate polymorphic behavior by invoking the method on different objects of the derived classes.

Shape Inheritance: Create a class hierarchy representing different shapes, such as "Circle," "Rectangle," and "Triangle." Each shape should have common attributes like area and perimeter, as well as specific attributes and methods. Implement inheritance and polymorphism to calculate and display the area and perimeter of different shapes.

Employee Inheritance: Design an employee management system using inheritance. Create a base class "Employee" and derived classes such as "Manager," "Developer," and "Tester." Each derived class should have specific attributes and methods, along with common attributes like name and salary. Implement functionality to calculate the total salary of all employees.

Bank Account Inheritance: Create a class hierarchy representing different types of bank accounts, such as "SavingsAccount" and "CurrentAccount." Implement common functionalities like deposit and withdrawal in the base class and specific functionalities like interest calculation in the derived classes.

Animal Inheritance: Design an animal hierarchy with a base class "Animal" and derived classes such as "Cat," "Dog," and "Bird." Add appropriate methods and attributes to represent different behaviors and characteristics of each animal. Implement polymorphism by invoking common methods on objects of different animal types.

School System Inheritance: Model a school system using inheritance. Create a base class "Person" and derived classes such as "Teacher" and "Student." Implement functionalities like assigning subjects to teachers and registering students for courses. Demonstrate inheritance and polymorphism by accessing and displaying information about different types of people in the school system.