Lab Task # 7

You are tasked with developing a system to manage different types of employees in a company. Each employee has a specific role and receives a different type of salary. You need to implement a Python program that models this system using object-oriented programming concepts, with a focus on polymorphism.

1. **Define the Employee Base Class:**
    - Start by defining a base class called `Employee`. This class will serve as the blueprint for all types of employees in the company.
    - The `Employee` class should have a method called `calculate_salary()` that returns the employee's salary. Since the base class doesn't have specific salary information, this method can return a generic value (e.g., 0).
2. **Create Subclasses for Different Roles:**
    - Next, create subclasses for each specific role in the company. For this example, let's create subclasses for `Manager`, `Developer`, and `Designer`.
    - Each subclass should inherit from the `Employee` base class and override the `calculate_salary()` method to provide a specific implementation for calculating the salary based on the role.
3. **Implement Salary Calculation Logic in Subclasses:**
    - In the `Manager` subclass, override the `calculate_salary()` method to calculate the manager's salary. Managers typically receive a fixed base salary plus a bonus.
    - In the `Developer` subclass, override the `calculate_salary()` method to calculate the developer's salary. Developers often receive an hourly wage multiplied by the number of hours worked.
    - In the `Designer` subclass, override the `calculate_salary()` method to calculate the designer's salary. Designers may receive a fixed monthly salary.
4. **Instantiate Objects and Demonstrate Polymorphism:**
    - Instantiate objects of each subclass (e.g., a `Manager` object, a `Developer` object, and a `Designer` object).
    - Call the `calculate_salary()` method on each object to demonstrate polymorphism. Despite being of different types, all objects should respond uniformly to the method call, providing specific salary calculations based on their roles.
5. **Output Results:**
    - Print out the calculated salaries for each type of employee to demonstrate that the polymorphic behavior is working as expected.