

Aim: Run time polymorphism and the use of abstract base class.

Suppose that you are designing a payroll system for a company. You need to arrange payments for the employees according to their working type. There are 2 different types of employees; `PieceWorker` and `HourlyWorker`. A `PieceWorker` represents an employee whose salary is based on a wage per product and the number of total products produced. An `HourlyWorker` represents an employee whose salary is based on an hourly wage and the number of total hours worked.

1. Define an **abstract base class** `Employee` with a **pure virtual method** `calculateEarnings()`. This method will be defined later in concrete derived classes to calculate earnings of the employees.
2. Derive a **concrete class** `PieceWorker` from `Employee` class. `PieceWorker` class should contain an extra private data member: **pieces** to store the number of products produced. (You are free in your design that the wage per a product of a `PieceWorker` may either be an extra data member or a constant value. For example, the wage per a product may be constant as 10 TL.)
3. Derive a concrete class `HourlyWorker` from `Employee` class. `HourlyWorker` class should contain an extra private data member: **hours** to store the total hours worked. (You are free in your design that the wage per an hour of a `PieceWorker` may either be an extra data member or a constant value. For example, the hourly wage may be constant as 12 TL.)
4. In class `PieceWorker`, provide a **concrete implementation** of method `calculateEarnings()` that calculates the employee's earnings by multiplying the number of pieces produced by the wage per piece. In class `HourlyWorker`, provide a concrete implementation of method `calculateEarnings()` that calculates the employee's earnings by multiplying the number of hours worked by the wage per hour.
5. Finally, test your class hierarchy using polymorphism mechanism. (In `main()` function, define base class pointers that will point derived class objects and call virtual method `calculateEarnings()` through those base class pointers.)