# BSC Computing and Information Systems

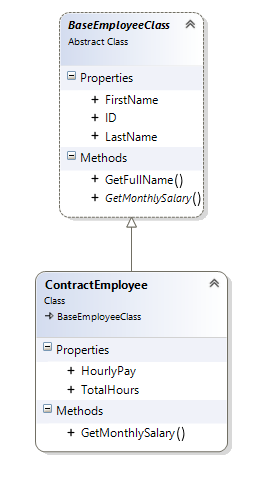
MOD005437 Object Oriented Programming Development

Week 6 – Abstract classes and Interfaces

**What are we doing today?**

* **Abstract**
  + **Practical**
    - **Employee salary program**
    - **Cinema Booking**

**Question:** Why and when should we use an abstract class? Provide an example of where we could use an abstract class.  
 **Answer**  
  
Let's understand with an example. In an organisation there are two types of employees FullTimeEmployee and contractEmployee. Full time employees have a monthly salary where as contract employees work a variable number of hours each month and complete a timesheet to state how many hours they have worked.  
  
For the first we will create a FullTimeEmployee class in a console application and provide the implementation of the class with the following procedure.  
  
**The base employee class defines shared attributes for all employees, but it will never be instantiated on its own. There is no such thing as a BaseEmployee.**

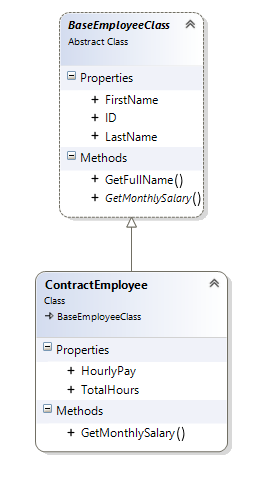
****

This class is comprised of public properties for the basic employee details, a method that returns the full name of the employee and an *abstract* method to return the monthly salary.

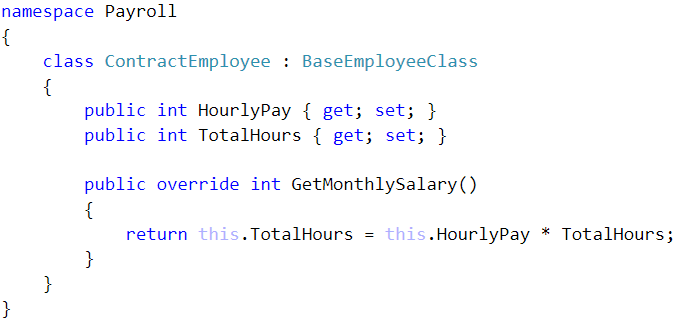
**A screenshot of a cell phone

Description generated with very high confidence**

As the GetMonthlySalary method is abstract it does not provide any implementation. Instead the monthly salary will be calculated differently depending on the type of employee.

****

The implementation for a Contract employee is then defined via overriding the abstract method:

****

|  |  |
| --- | --- |
|  | Implement the class diagram in C#  Extension:   * Instantiate examples of each employee type and output their Monthly Salaries to the screen * How would you handle overtime? * Add this to your implementation |

Activity 2

Working in pairs: Design a class diagram for an object oriented program that uses abstract classes to provide a solution to the following scenario

* Produce a class diagram which represents the following scenario
* Design a simple booking system for a cinema
* There should be at least two types of screens
  + Standard screens
    - Always publically available
    - Seats assigned automatically (sequentially)
  + Deluxe screens
    - Can be used by public
    - Can be hired for private screenings
    - Seats allocated manually
  + Screens contain seats which have a seat number, can be occupied and might have a cup holder
  + Filmgoers book seats
  + Parties book multiple seats for private bookings
* You should include at least 1 example of an abstract class

Extension: Implement the design in C#