OOP Principles - Part 2

1 Shapes

- Define abstract class `Shape` with only one `abstract` method `CalculateSurface()` and fields `width` and `height`.
- Define two new classes `Triangle` and `Rectangle` that implement the `virtual` method and return the surface of the figure (height * width for rectangle and height * width/2 for triangle).
- Define class `Square` and suitable constructor so that at initialization `height` must be kept equal to `width` and implement the `CalculateSurface()` method.
- Write a program that tests the behaviour of the `CalculateSurface()` method for different shapes (`Square`, `Rectangle`, `Triangle`) stored in an array.

2 Bank accounts

- A `bank` holds different types of accounts for its customers: `deposit` accounts, `loan` accounts and `mortgage` accounts. Customers could be `individuals` or `companies`.
- All accounts have customer, balance and interest rate (monthly based).
  - Deposit accounts are allowed to deposit and with draw money.
  - Loan and mortgage accounts can only deposit money.
- All accounts can calculate their interest amount for a given period (in months). In the common case its is calculated as follows: `number_of_months` * `interest_rate`.
- Loan accounts have no interest for the first `3` months if are held by individuals and for the first `2` months if are held by a company.
- Deposit accounts have no interest if their balance is positive and less than `1000`.
- Mortgage accounts have ½ interest for the first `12` months for companies and no interest for the first `6` months for individuals.
- Your task is to write a program to model the bank system by classes and interfaces.
- You should identify the classes, interfaces, base classes and abstract actions and implement the calculation of the interest functionality through overridden methods.

3 Range Exceptions

- Define a class `InvalidRangeException<T>` that holds information about an error condition related to invalid range. It should hold error message and a range definition [`start … end`].
- Write a sample application that demonstrates the `InvalidRangeException<int>` and `InvalidRangeException<DateTime>` by entering numbers in the range [`1..100`] and dates in the range [`1.1.1980 … 31.12.2013`].