# Object

* Định nghĩa object: trong một object có các **đặc tính và hành vi**
* Vd: object là Human, trong Human ta nhìn thấy các đặc điểm của object này như màu tóc, tuổi, chiều cao. Ngoài ra Human có thể thực hiện các hành động như đi, nói, ăn…
  + Human:
    - Attribute: hairColor, age, height
    - Behavior: talking, walking, eating
* Nếu lập trình OOP giúp gói gọn nhiều cấu trúc trong một cấu trúc dữ liệu object, thì các cấu trúc trong lập trình hàm lại độc lập nhau (đôi khi có biến global không giới hạn truy cập)
* Ngoài sử dụng các cấu trúc OOP, ta còn có thể điều khiển truy cập (**access control**) tới member trong object (attribute+behavior)
* Trong thuật ngữ OOP:
  + Data – Attribute
  + Behavior – method
  + Giới hạn truy cập tới attribute/method được gọi là **Data Hiding**

# Encapsulation

* Gộp lại attribute và method trong một entity, OOP gọi nó là **encapsulation** (bao đóng)
  + Vd: object Math, trong Math có 2 số nguyên là myInt1, myInt2
    - Giới hạn truy cập sử dụng trên 2 số nguyên này với chỉ trong phạm vi object, các method của các entity ngoài Math không thể tùy ý thay đổi 2 số nguyên này
    - Math cung cấp một số method định nghĩa cho các thao nhất định mà các object khác có thể sử dụng: get, set
    - Math có method Sum() sẽ trả về tổng, các object khác có thể truy cập
    - -> Encapsulate lại myInt1 myInt2 trong Math, encapsulate các method get/set, Sum() trong phạm vi rộng hơn
* Method Getter, Setter trợ giúp cho khái niệm data hiding, các object không nên được thao tác trực tiếp trên data của object khác, có tên gọi khác là accessor method
  + Vd, Payroll object có một method CalculatePay() chức năng là tính lương cho một nhân viên.
    - Payroll sẽ phải lấy được mã số an sinh xã hội của nhân viên
    - Employee:
      * Attr: socialSecurityNumber, gender, dob
      * Method: getters
    - Payroll:
      * Attr: pay
      * Method: calculatePay:double
* Encapsulation và Data hiding:
  + Một object không cần phải chia sẻ mọi attribute, behavior ra bên ngoài
  + Object nên chỉ thể hiện những interface mà các object khác muốn thực hiện, chứ không cần biết tới cài đặt chi tiết của interface là gì

# Demo Code: Xây dựng form tính tiền lương cho nhân viên

Requirement:

* User lấy thông tin nhân viên (Id, Name, Age, PayRate) theo Id
* User lấy tiền lương (basePay\*empPayRate) cho một nhân viên nhất định

Class design