Day.6

Abstract methods and Abstract classes
Abstract classes and runtime polymorphism
Interfaces
Interfaces and runtime polymorphism
Interfaces to share constants, marker interfaces

Q1) A scientific application needs to use several constants through out the application. It was decided to create an interface called PhysicalConstants. The constants that needs to be set are

– Speed of light in vacuum (C):299 792 458 m/s

– Gravitational constant (G): 6.674 28×10−11 m3 kg−1 s−2

– Standard Gravitational Acceleration(g) : 9.806 65 m/s2

• Use these constants for a class that has following functions.

– E= MC2

– F=G (m1 m2)/r2

– d=0.5 gt

– Make sure that constants names are used without repeating interface names with constants(Hint : use static imports).

Q2) Shape abstract class has 2 abstract methods

– area ()

– volume ()

a) Classes Cube, Rectangle, Triangle and Sphere are created . For Rectangle, Triangle volume returns -1.

b) Shapes that implement volume must be of type Spatial which is a marker interface.

c) The user enters 5 shape objects which is stored in an array.

d) Finally, all the Shape objects are printed. Only for Shape object of Spatial type, volume is printed.

Hint:

Triangle Area = 1/2 of the base X the height,

Rectangle Area: l X w

Sphere Area= 4pr2 , Volume = 4/3 pr3

Cube Area = 2lw + 2lh + 2wh Volume = l X w X h

Where l is length, w is width and h is height

Q3) Create an abstract class called BalanceComputer that has one implemented static method called getBalanceComputer (char type) and one abstract method getBalance(). The getBalanceComputer() method returns BalanceComputer object with one of the formula (listed below) implemented for getBalance() based on the type of account: current (C)or savings(S).

For current acct formula is : amt (end of the year)= principal amount+ (principal amount * rate * time)/100

For Savings formula is : amt (calculated quarterly)= principal amount (1+ rate/4)4t