# Session 1.1 – Pricing (draff design)

## • Requirement #1

In requirement #1, we see an object "coffee cup". So, we will create a class named CoffeeCup to present coffee cup. A coffee cup has some fields: drink type, size, topping or not, base price, so we design CoffeeCup class with those fields. A coffee cup can be changed drink type, size, topping or not, then we design some methods to change fields or coffee cup. Requirement #1 require calculate price of coffee cup, so we add method CalculatePrice1 to calculate price and Amount field to store the result. Class diagram present the design below:

□ CoffeeCup		
+ DrinkType: CoffeeDrinkType		
+ Size: CoffeeSize		
+ CreamTopping: bool		
- BasePrice: Double		
+ Amout: Double		
- ChangeDrinkType(type): void		
- ChangeSize(type): void		
- ChangeCreamTopping(bool): void		
+ CalculatePrice1(drinkType, size, topping): void		

### Requirement #2:

Requirement #2 add size XL and milk tea drink type, so we only add more type to CoffeeDrinkType and CoffeeSize enum. Milk tea drink type has base price is \$2.25, so we add \_milkTeaBasePrice field to CoffeeCup class. And finally, we add method AddMilkOption and MilkOption field

□ CoffeeCup			
+ DrinkType: CoffeeDrinkType			
+ Size: CoffeeSize			
+ CreamTopping: bool			
+ MilkOption: CoffeeMilkOption			
- BasePrice: Double			
- MilkTeaBasePrice: Double			
+ Amount: Double			
- ChangeDrinkType(type): void			
- ChangeSize(type): void			
- ChangeCreamTopping(bool): void			
- AddMilkOption(milkOption): void			
+ CalculatePrice1(drinkType, size, topping): void			
+ CalculatePrice2(drinkType, size, toping, milkOption): void			

### • Requirement #3

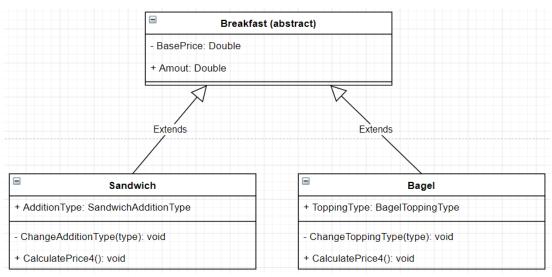
• Design same the requirement #2, we add *ChocolateSauce* field, *AddChocolateSauce* and *CalculatePrice3* method follow the requirement.

□ CoffeeCup		
+ DrinkType:	CoffeeDrinkType	
+ Size: Coffe	eSize	
+ CreamTopp	ing: bool	
+ MilkOption:	CoffeeMilkOption	
+ ChocolateS	sauce: int	
- BasePrice:	Double	
- MilkTeaBas	ePrice: Double	
+ Amount: Do	puble	
- ChangeDrin	kType(type): void	
- ChangeSize	e(type): void	
- ChangeCre	amTopping(bool): void	
- AddMilkOpt	ion(milkOption): void	
+ CalculatePi	rice1(drinkType, size, topping): void	
+ CalculatePi	rice2(drinkType, size, toping, milkOption): void	
+ CalculatePi	rice3(drinkType, size, toping, milkOption, chocolateSauce): void	

0

## Requirement #4

In requirement #4, we have two objects: sandwich and bagel. Both have base price is \$3 and have Amount after calculation. But, sandwich can add egg or turkey and bagel can add butter or cream cheese. So, we design a class named Breakfast as a parent class and Sandwich, Bagel class extend from Breakfast class



#### • Requirement #5

- In requirement #5, we have to calculate list of items, so we need a class to contains items named *Order*. Order class includes:
  - *coffeeCup* field present information about a coffee cup
  - breakfast field present information about a breakfast
  - *TotalAmount* field store total amount of all items
  - *TaxAmount* field store amount of tax
  - CalculatePrice5 method to calculate amount of all items
    Price break down foreach item can be access from coffeeCup.Amount and breakfast.Amount field

Order
+ coffeeCup: CoffeeCup
+ breakfast: Breakfast
+ TotalAmount: double
+ TaxAmount: double
+ CalculatePrice5(): void