CSE 230 Problem Set 13

# Problem 29.1: Currency

Consider the following problem definition:

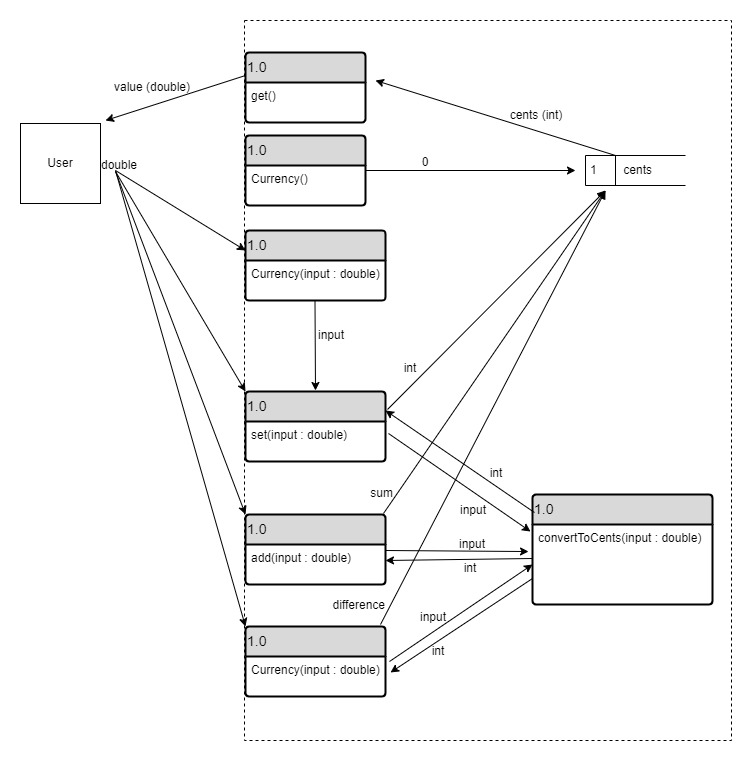
A class is designed to represent currency (money). In this case, the currency is always positive and the smallest denominations are cents. This class is used in a financial application that has direct user textual input.

Please do the following

1. Create a class diagram to describe the class:

|  |
| --- |
| Currency |
| -int cents |
| +Currency()  +Currency(input : double)  +set(input : double)  +add(input : double)  +subtract(input : double)  +get() : double  -convertToCents(input : double) : int |

1. Create a DFD to illustrate how data moves into and out of the member variables. Hint: you might want to draw this out and insert a picture.



1. Provide pseudocode for the methods responsible for keeping the member variables in a valid state.

Currency()

sents 🡨 0

Currency(input : double)

set(double)

set(input : double)

IF double < 0

Cents 🡨 0

ELSE

Cents 🡨 convertToCents(input)

convertToCents(input : double)

RETURN (int)(input \* 100)

add(input : double)

input 🡨 convertToCents(input)

cents 🡨 cents + input

subtract(input : double)

input 🡨 convertToCents(input)

IF cents > input

cents 🡨 cents – input

ELSE

cents 🡨 0

get() const

returnValue 🡨 (double)(cents) / 100

RETURN returnValue

# Problem 29.2: Spaceship Fuel

Consider the following problem definition:

A class is designed to represent the fuel amount in a spaceship. A variety of interfaces can adjust the fuel level, including refueling stations, bonus fuel loads, and the engines that consume fuel the more they are used.

Please do the following

1. Create a class diagram to describe the class:

|  |
| --- |
| Spaceship\_Fuel |
| -double fuel\_level  -double max\_fuel\_level |
| +Spaceship\_Fuel()  +Spaceship\_Fuel(double amount,double max\_amount)  +add\_fuel(double amount)  +spend\_fuel(double amount)  +get\_fuel\_level() : double  +get\_max\_fuel\_level() : double  -set\_fuel\_level(double amount)  -set\_max\_fuel\_level(double amount) |

1. Create a DFD to illustrate how data moves into and out of the member variables. Hint: you might want to draw this out and insert a picture.

A diagram of a data flow

Description automatically generated

1. Provide pseudocode for the methods responsible for keeping the member variables in a valid state.

Spaceship\_Fuel()

set\_fuel\_level(0.0)

set\_max\_fuel\_level(10.0)

Spaceship\_Fuel(amount: double, max\_amount: double)

IF max\_amount <= 0

RETURN Spacehip\_Fuel()

ELSE

set\_max\_fuel\_level(max\_amount)

set\_fuel\_level(amount)

get\_fuel\_level() const

RETURN fuel\_level

get\_max\_fuel\_level() const

RETURN max\_fuel\_level

set\_max\_fuel\_level(amount: double)

IF max\_amount <= 0

RETURN error

ELSE

max\_fuel\_level = amount

set\_fuel\_level(amount: double)

IF amount < 0

RETURN error

IF amount > get\_max\_fuel\_level()

fuel\_level = get\_max\_fuel\_level()

ELSE

fuel\_level = amount

add\_fuel(amount: double)

IF amount <= 0

RETURN error

IF amount + get\_fuel\_level() >= get\_max\_fuel\_level()

set\_fuel\_level(get\_max\_fuel\_level())

ELSE

set\_fuel\_level(get\_fuel\_level() + amount)

spend\_fuel(amount: double)

IF amount <= 0

RETURN error

IF get\_fuel\_level() - amount < 0

RETURN error

ELSE

set\_fuel\_level(get\_fuel\_level() - amount)