CSE 230 Problem Set 13

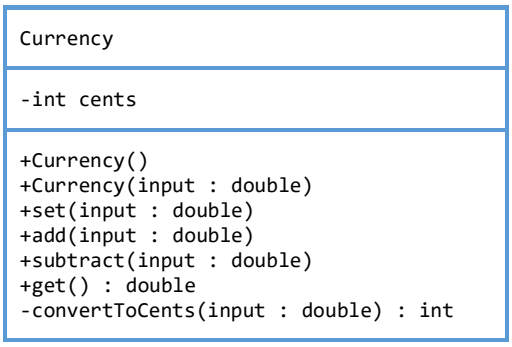
Problem 29.1: Currency

Consider the following problem definition:

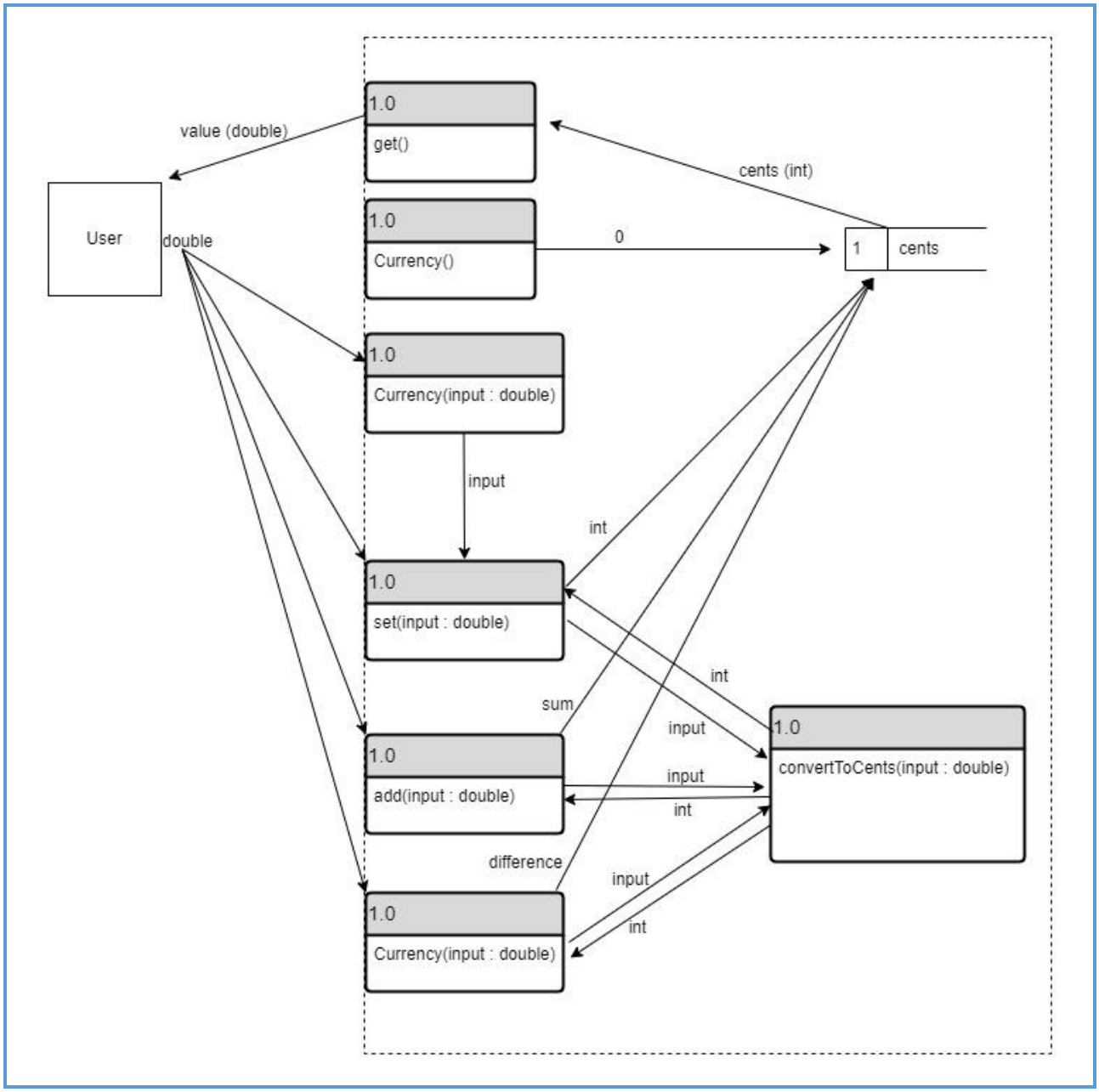
> A class is designed to represent currency (money). In this case, the currency is always positive and the smallest denominations are cents. This class is used in a financial application that has direct user textual input.

Please do the following

1. Create a class diagram to describe the class:



2. 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.



3. 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:

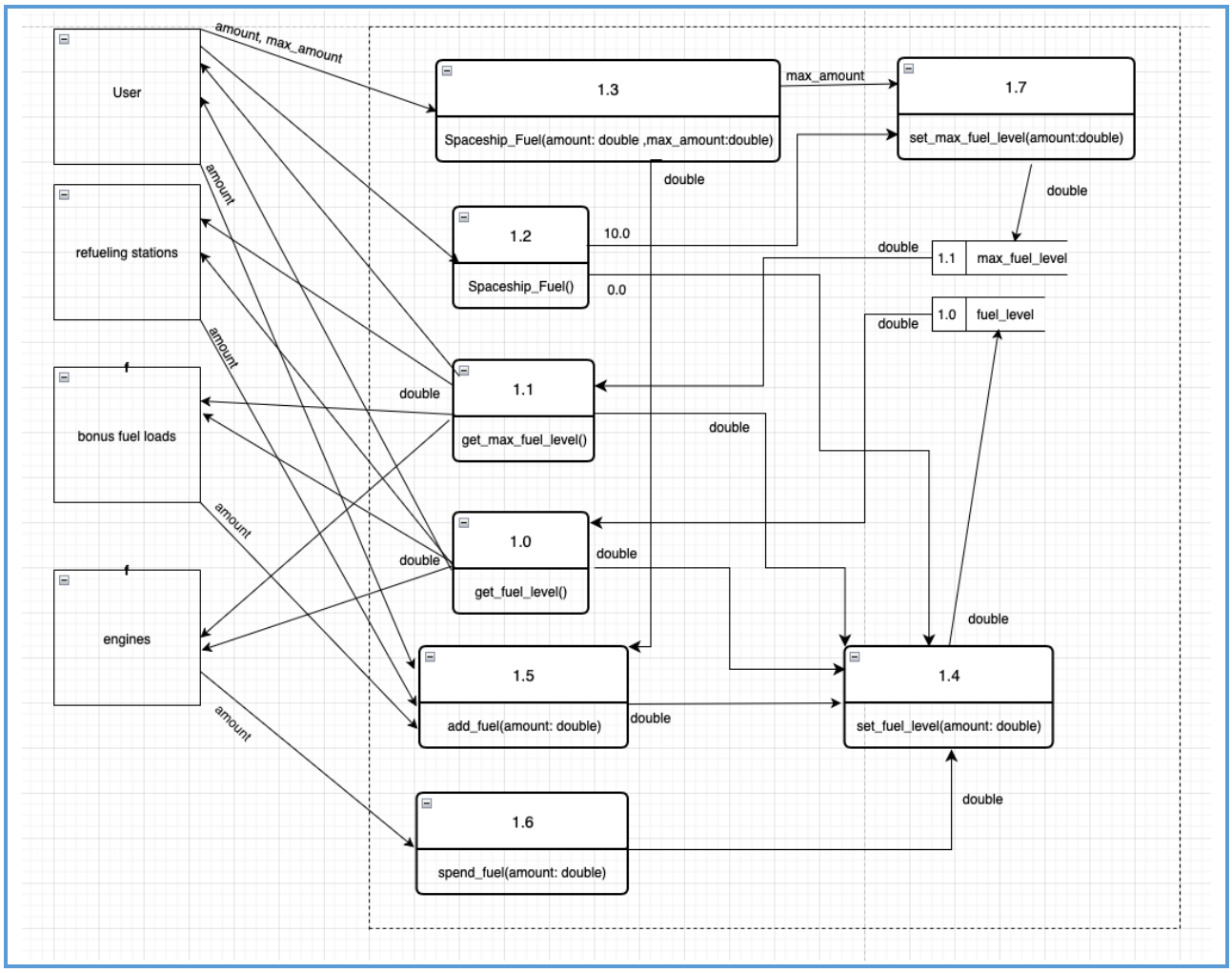
> A class is designed to represent the fuel amount in a spaceship. A variety of interfaces can adjust the fuel level, including refueling stations, bonus fuel loads, and the engines that consume fuel the more they are used.

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)

2. 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.



3. 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)
```