Lab5: Money Using Arrays

Goal: Create a data abstraction to represent Money. Use the functions you create appropriately in other functions.

Required: Functions, Conditionals, Loops, Arrays

IMPORTANT: YOU MAY ASSUME ALL DOLLARS AND CENTS WILL BE >= 0. YOU MAY ASSUME CENTS WILL BE [0,99], **EXCEPT FOR IN createMoney().** YOU MAY ASSUME MONEY WILL ALWAYS BE POSITIVE AS ARGUMENTS AND AFTER COMPUTATION.

TEST YOUR CODE AS YOU GO!!! DO NOT USE DOUBLES IN THE ENTIRE LAB.

Explanation: Money will be stored as 2 integers, 1 for dollars and 1 for cents. The 2 integers will be represented as an array of 2 integers.

Ex: $10.24 will be represented as the array {10, 24} because there are 10 dollars and 24 cents.

Ex: $3.00 will be represented as the array {3, 0} because there are 3 dollars and 0 cents.

Create **CONSTRUCTOR FUNCTIONS** (Creating data according to your abstraction)

1. int[] createMoney(int dollars, int cents); // Given the number of dollars and cents, construct a Money abstraction. If given >99 cents, convert it to dollars.

```
//Insert Code with comments here:
```

2. int[] copyMoney(int[] money); // Given a money, create a separate copy of it.

```
//Insert Code with comments here:
```

Create **ACCESSOR FUNCTIONS** (Using information from data without changing it)

3. int dollars(int[] money); // Returns the number of dollars in the amount. Ex, dollars($12.34) => 12. $12.34 is "money," ie, {12,34}

```
//Insert Code with comments here:
```

4. int cents(int[] money); // Returns the number of cents in the amount. Ex, cents($12.34) => 34, NOT .34!!!

```
//Insert Code with comments here:
```

5. String moneyToString(int[] money); // Returns a nice looking string. Ex, "$6.25", "$0.21", "$4.01", "$2.00". MAKE SURE TO CONSIDER ALL EXAMPLES!

```
//Insert Code with comments here:
```

6. **String moneyToText(int[] money);** // Returns the Money as words. Ex,{123,51} => "one hundred and twenty three dollars and fifty one cents." YOU MAY ASSUME money <$1000.

```
//Insert Code with comments here:
```

Create **CHECKING FUNCTIONS**.

7. boolean isGreaterThan(int[] m1, int[] m2); // Returns True if m1 > m2.

//Insert Code with comments here:

8. boolean isEqual(int[] m1, int[] m2); // Returns True if m1 == m2.

//Insert Code with comments here:

Create **ADDING FUNCTIONS**.

Consider the examples:

$3.50 + $4.25 => $7.75

$10.99 + $11.99 => 22.98 What is the maximum sum of 2 cent values?

9. void adder(int[] m1, int[] m2); // Make m1 the sum of both m1 and m2. Leave m2 untouched.

//Insert Code with comments here:

10. int[] add(int[] m1, int[] m2); // Return the sum of both m1 and m2. m1 and m2 untouched.

//Insert Code with comments here:

11. void subber(int[] m1, int[] m2); // Make m1 the difference of m1 - m2. Leave m2 untouched.

//Insert Code with comments here:

12. int[] sub(int[] m1, int[] m2); // Return the difference between m1 - m2. m1 and m2 untouched.

//Insert Code with comments here:

Create **CALCULATION FUNCTIONS**.

13. int[] payWith20(int[] owe);
// If you owe $5.12 and pay with $20.00, your change should be $14.88
// If you owe $3.91 and pay with $20.00, your change should be $16.09
// You may assume you always pay with $20, and you always owe <= $20.

//Insert Code with comments here:

14. **int[] applyInterest(int[] balance, int interest);** // Interest is stored as an int, so 5 represents 5%. Do not use doubles.
// Ex, m = {3, 25} and interest = 5, representing 5%. Before starting the problem, we can calculate 5% of $3.25 = $ 0.1625. You can just chop the .0025 to get $0.16 interest. Adding it to your starting balance, you get 3.41.

//Insert Code with comments here:

TEST CODE

```
public static void main(String[] args){
    // createMoney()
    int[] a = createMoney(4,115);
    System.out.println("5 15: " + a[0] + " " + a[1]);
    // copyMoney()
    int[] b = copyMoney(a);
    a[1] = 50;
    System.out.println("5 50: " + a[0] + " " + a[1]);
    System.out.println("5 15: " + b[0] + " " + b[1]);
    // dollars()
    System.out.println("Dollars: 5: " + dollars(a));
    // cents()
    System.out.println("Cents: 50: " + cents(a));
    // moneyToString()
    System.out.println("$5.50: " + moneyToString(a));
    int[] c = createMoney(1,2);
    System.out.println("$1.02: " + moneyToString(c));
    // moneyToText()
    System.out.println("five dollars and fifty cents: " + moneyToText(a));
    // isGreater()
    System.out.println("isGreater: true: " + isGreaterThan(a,b));
    // isEqual()
    System.out.println("isEqual: false: " + isEqual(a,b));
    // adder()
    a = createMoney(1,10);
    b = createMoney(2,20);
    adder(a,b);
    System.out.println("$3.30: " + moneyToString(a));
    System.out.println("$2.20: " + moneyToString(b));
    // add()
    b = add(a,b);
    System.out.println("$3.30: " + moneyToString(a));
    System.out.println("$5.50: " + moneyToString(b));
```