3.22 LAB: Smallest number

Write a program whose inputs are three integers, and whose output is the smallest of the three values.

Ex: If the input is:

```
7 15 3
```

the output is:

```
3
```

Solution:

```
import java.util.Scanner;

public class LabProgram {
   public static void main(String[] args) {
      Scanner scnr = new Scanner(System.in);
      int num1;
      int num2;
      int num3;

      num1 = scnr.nextInt();
      num2 = scnr.nextInt();
      num3 = scnr.nextInt();

      if ((num1 <= num2) && (num1 <= num3)) {    // num1 is smaller than both num2 and num3,
so is smallest
         System.out.println(num1);
      }
      else if ((num2 <= num1) && (num2 <= num3)) { // num2 is smaller than both num1 and
num3, so is smallest
         System.out.println(num2);
      }
      else { // Neither num1 nor num2 are smallest, so num3 must be smallest
         System.out.println(num3);
      }
   }
}
```

3.23 LAB: Interstate highway numbers

Primary U.S. interstate highways are numbered 1-99. Odd numbers (like the 5 or 95) go north/south, and evens (like the 10 or 90) go east/west. Auxiliary highways are numbered 100-999, and service the primary highway indicated by the rightmost two digits. Thus, I-405 services I-5, and I-290 services I-90. Note: 200 is not a valid auxiliary highway because 00 is not a valid primary highway number.

Given a highway number, indicate whether it is a primary or auxiliary highway. If auxiliary, indicate what primary highway it serves. Also indicate if the (primary) highway runs north/south or east/west.

Ex: If the input is:

```
90
```

the output is:

```
I-90 is primary, going east/west.
```

Ex: If the input is:

```
290
```

the output is:

```
I-290 is auxiliary, serving I-90, going east/west.
```

Ex: If the input is:

```
0
```

or any number not between 1 and 999, the output is:

```
0 is not a valid interstate highway number.
```

Ex: If the input is:

```
200
```

the output is:

```
200 is not a valid interstate highway number.
```

Solution:

```
import java.util.Scanner;

public class LabProgram {
  public static void main(String[] args) {
    Scanner scnr = new Scanner(System.in);
    int highwayNumber;
    int primaryNumber;

    highwayNumber = scnr.nextInt();

    // Output whether the highway is auxiliary (serving which primary), or primary
    if ((highwayNumber < 1) || (highwayNumber > 999) || ((highwayNumber % 100) == 0)) { //
Invalid
      System.out.println(highwayNumber + " is not a valid interstate highway number.");
    }
    else { // Valid
      if (highwayNumber > 99) {
        System.out.print("I-" + highwayNumber + " is auxiliary");
        // Get the primary
        primaryNumber = highwayNumber % 100; // Gets the rightmost 2 digits
        System.out.print(", serving I-" + primaryNumber);
      }
      else { // Must be 1-99
        primaryNumber = highwayNumber;
        System.out.print("I-" + primaryNumber + " is primary");
      }

      // Ready now to output the direction.
      if ((primaryNumber % 2) == 0) { // Even
        System.out.println(", going east/west.");
      }
      else { // Odd
        System.out.println(", going north/south.");
      }
    }
  }
}
```

3.24 LAB: Exact change

Write a program with total change amount in pennies as an integer input, and output the change using the fewest coins, one coin type per line. The coin types are Dollars, Quarters, Dimes, Nickels, and Pennies. Use singular and plural coin names as appropriate, like 1 Penny vs. 2 Pennies.

Ex: If the input is: `0`

the output is: `No change`

Ex: If the input is: `45`

the output is:
```
1 Quarter
2 Dimes
```

Solution:

```
import java.util.Scanner;

public class LabProgram {
   public static void main(String[] args) {
      Scanner scnr = new Scanner(System.in);
      int inputVal;
      int numDollars;
      int numQuarters;
      int numDimes;
      int numNickels;
      int numPennies;

      inputVal = scnr.nextInt();

      if (inputVal <= 0) {
         System.out.println("No change");
      } // Could return at this point

      numDollars = inputVal / 100;
      inputVal = inputVal - (numDollars * 100);

      numQuarters = inputVal / 25;
      inputVal = inputVal - (numQuarters * 25);

      numDimes = inputVal / 10;
      inputVal = inputVal - (numDimes * 10);

      numNickels = inputVal / 5;
      inputVal = inputVal - (numNickels * 5);
      numPennies = inputVal;

      if (numDollars > 0) {
         System.out.print(numDollars);
         if (numDollars == 1) {
            System.out.println(" Dollar");
         }
         else {
            System.out.println(" Dollars");
         }
      }

      if (numQuarters > 0) {
         System.out.print(numQuarters);
         if (numQuarters == 1) {
            System.out.println(" Quarter");
         }
         else {
            System.out.println(" Quarters");
         }
      }

      if (numDimes > 0) {
         System.out.print(numDimes);
         if (numDimes == 1) {
            System.out.println(" Dime");
         }
         else {
            System.out.println(" Dimes");
         }
      }

      if (numNickels > 0) {
         System.out.print(numNickels);
         if (numNickels == 1) {
            System.out.println(" Nickel");
         }
         else {
            System.out.println(" Nickels");
         }
      }

      if (numPennies > 0) {
         System.out.print(numPennies);
         if (numPennies == 1) {
            System.out.println(" Penny");
         }
         else {
            System.out.println(" Pennies");
         }
      }
   }
}
```

3.25 LAB: Leap year

A year in the modern Gregorian Calendar consists of 365 days. In reality, the earth takes longer to rotate around the sun. To account for the difference in time, every 4 years, a leap year takes place. A leap year is when a year has 366 days: An extra day, February 29th. The requirements for a given year to be a leap year are:

1) The year must be divisible by 4

2) If the year is a century year (1700, 1800, etc.), the year must be evenly divisible by 400; therefore, both 1700 and 1800 are not leap years

Some example leap years are 1600, 1712, and 2016.

Write a program that takes in a year and determines whether that year is a leap year.

Ex: If the input is:

```
1712
```

the output is:

```
1712 - leap year
```

Ex: If the input is:

```
1913
```

the output is:

```
1913 - not a leap year
```

Solution:

```
import java.util.Scanner;

public class LabProgram {
   public static void main(String[] args) {
      Scanner scnr = new Scanner(System.in);
      int inputYear;
      boolean isLeapYear;

      isLeapYear = false;
      inputYear = scnr.nextInt();

      if ((inputYear % 4) == 0) { // inputYear is divisible by 4
         if ((inputYear % 100) == 0) { // inputYear is divisible by 100 (century year)
            if ((inputYear % 400) == 0) { // inputYear is divisible by 400
               isLeapYear = true;
            }
            else {                        // inputYear is not divisible by 400
               isLeapYear = false;
            }
         }
         else {                        // inputYear is not divisible by 100
            isLeapYear = true;
         }
      }
      else {                        // inputYear is not divisible by 4
         isLeapYear = false;
      }

      if (isLeapYear) {
         System.out.println(inputYear + " - leap year");
      }
      else {
         System.out.println(inputYear + " - not a leap year");
      }
   }
}
```