CS116
Lab3

Questions 1a & 1b
Pseudocode and Test Tables are in DateClient

Question 2

Pseudocode:

Zoning

- Instance variables
    - Private attributes -> Length (double), width (double)
- Class constants
    - Zoning lot area levels (double)
    - Height limit base amounts for each zone (double)
    - Percents for each zone (double)
- Default constructor
    - Set private attributes equa (height and width)l to 0
- Non Default constructor
    - Check and set private attributes equal to the height and width taken in as a parameter
- Accessors
    - Return values (getHeightLimit(), getWidth(), getLength(), etc), all public double methods
- Mutators
    - setLength() -> void method, length as a parameter, check if length >= 0
    - setWith() -> void method, width as a parameter, check if length >= 0
        - Else = 0
- public double heightLimit()
    - Use if statements to check what category the area falls into
    - Parameter: none, return: double
- public int unitCountLimit()
    - Use if statements to check what category the area falls into
    - Parameter: none, return: int
- public String toString()
    - Parameter: none, return String

ZoningApp

- Instantiate a Zoning object

    - Prompt the user to enter length and width
    - check for random area for each zone, check upper and lower limits for each zone
- Accessor Methods
    - call each of the GetMethods
- Mutator Methods
    - setLength(), setWidth() -> try length and width greater than 0, less than 0, and equal to 0
- toString
    - print out

Test Case	**Sample Data**	**Expected Result**	**Tested?**
Zone 1	Length: 0.0 Width: 0.0 Area: 0.0	Length: 0.0 Width: 0.0 Area: 0.0 Height Limit: 25.0 Unit Count Limit: 1	YES
Zone 1	Length: -14.0 Width: 120.0 Area: 0.0	Length: 0.0 Width: 120.0 Area: 0.0 Height Limit: 25.0 Unit Count Limit: 1	YES
Zone 1	Length: 125.0 Width: 11.0 Area: 1375.0	Length: 125.0 Width: 11.0 Area: 1375.0 Height Limit: 25.0 Unit Count Limit: 1	YES
Zone 1	Length: 200.0 Width: 10.0 Area: 2000.0	Length: 200.0 Width: 10.0 Area: 2000.0 Height Limit: 25.0 Unit Count Limit: 1	YES
Zone 2	Length: 15.0 Width: 140.0 Area: 2100.0	Length: 15.0 Width: 140.0 Area: 2100.0 Height Limit: 27.0 Unit Count Limit: 2	YES
Zone 2	Length: 25.0 Width: 100.0 Area: 2500.0	Length: 25.0 Width: 100.0 Area: 2500.0 Height Limit: 35.0 Unit Count Limit: 2	YES

Zone 3	Length: 125 Width: 25 Area: 3125.0	Length: 125.0 Width: 25.0 Area: 3125.0 Height Limit: 41.25 Unit Count Limit: 2	YES
Zone 3	Length: 100.0 Width: 35.0 Area: 3500.0	Length: 100.0 Width: 35.0 Area: 3500.0 Height Limit: 45.0 Unit Count Limit: 2	YES
Zone 4	Length: 50.0 Width: 80.0 Area: 4000.0	Length: 50.0 Width: 80.0 Area: 4000.0 Height Limit: 47.5 Unit Count Limit: 3	YES
Zone 4	Length: 50.0 Width: 100.0 Area: 5000.0	Length: 50.0 Width: 100.0 Area: 5000.0 Height Limit: 52.5 Unit Count Limit: 3	YES
Zone 5	Length: 50.0 Width: 165.0 Area: 8250.0	Length: 0.0 Width: 0.0 Area: 0.0 Height Limit: 60.625 Unit Count Limit: 4	YES

Question 3

Demonstration of setting breakpoints, stepping through code, and watching variables for DateClient.java

Command Line Debugging

```
Microsoft Windows [Version 10.0.17134.523]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Ifrah>cd CSLABS

C:\Users\Ifrah\CSLABS>javac -g DateClient.java

C:\Users\Ifrah\CSLABS>jdb DateClient
Initializing jdb ...
> stop in DateClient.main
Deferring breakpoint DateClient.main.
It will be set after the class is loaded.
> run
run DateClient
Set uncaught java.lang.Throwable
Set deferred uncaught java.lang.Throwable
>
VM Started: Set deferred breakpoint DateClient.main

Breakpoint hit: "thread=main", DateClient.main(), line=34 bci=0
34       Date date1 = new Date(2,1,-100);

main[1] list
30    {
31      public static void main( String [] args )
32      {
33         // add code to construct Data objects
34 =>    Date date1 = new Date(2,1,-100);
35       Date date2 = new Date(2,29,2006);
36       Date date3 = new Date(1,1,2000);
37       Date date4 = new Date(2,29,2004);
38       Date date5 = new Date(2,29,1900);
39       Date date6 = new Date(2,29,1600);
main[1] step
>
Step completed: "thread=main", Date.<init>(), line=21 bci=0
21      public Date( int mm, int dd, int yyyy)   {

main[1] locals
Method arguments:
mm = 2
dd = 1
yyyy = -100
Local variables:
main[1] print date1
com.sun.tools.example.debug.expr.ParseException: Name unknown: date1
 date1 = null
main[1] step
>
Step completed: "thread=main", Date.<init>(), line=5 bci=4
5      final int DEFAULTYEAR = 2000;
```

```
6
7       /** default constructor
8       *  sets month to 1, day to 1 and year to 2000
9       */
10       public Date( )   {
main[1] step
>
Step completed: "thread=main", Date.<init>(), line=22 bci=11
22         setDate( mm, dd, yyyy );

main[1] list
18       *
19       *  passes parameters to setDate method
20       */
21       public Date( int mm, int dd, int yyyy)   {
22 =>      setDate( mm, dd, yyyy );
23       }
24
25       /* accessor methods */
26       int getMonth( ) { return month; }
27       int getDay( )   { return day; }
main[1] locals
Method arguments:
mm = 2
dd = 1
yyyy = -100
Local variables:
main[1] print Date
com.sun.tools.example.debug.expr.ParseException: Name unknown: Date
 Date = null
main[1] step
>
Step completed: "thread=main", Date.setDate(), line=38 bci=0
38          if(yyyy>=0 && leapYear(yyyy) == true && mm==2 && dd==29)

main[1] step
>
Step completed: "thread=main", Date.setDate(), line=45 bci=42
45       else if(yyyy>=0 && leapYear(yyyy) == false && mm==2 && dd==29){

main[1] list
41               setMonth( mm );
42               setDay( dd );
43
44       }
45 =>    else if(yyyy>=0 && leapYear(yyyy) == false && mm==2 && dd==29){
46               mm=1;
47               setYear(yyyy);
48               setMonth( 3 );
49               setDay( 1 );
50       }
```

```
main[1] locals
Method arguments:
mm = 2
dd = 1
yyyy = -100
Local variables:
main[1] print yyyy
 yyyy = -100
main[1] step
>
Step completed: "thread=main", Date.setDate(), line=52 bci=85
52              if(yyyy<0){

main[1] list
48              setMonth( 3 );
49              setDay( 1 );
50      }
51      else{
52 =>           if(yyyy<0){
53                      yyyy=DEFAULTYEAR;
54              }
55              setYear(yyyy);
56              setMonth( mm );
57              setDay( dd );
main[1] step
>
Step completed: "thread=main", Date.setDate(), line=53 bci=89
53                      yyyy=DEFAULTYEAR;

main[1] list
49              setDay( 1 );
50      }
51      else{
52              if(yyyy<0){
53 =>                   yyyy=DEFAULTYEAR;
54              }
55              setYear(yyyy);
56              setMonth( mm );
57              setDay( dd );
58      }
main[1]  stop in Date.setDay
Set breakpoint Date.setDay
main[1] list
49              setDay( 1 );
50      }
51      else{
52              if(yyyy<0){
53 =>                   yyyy=DEFAULTYEAR;
54              }
55              setYear(yyyy);
56              setMonth( mm );
57              setDay( dd );
58      }
```

```
57              setDay( dd );
58      }
main[1] cont
>
Breakpoint hit: "thread=main", Date.setDay(), line=68 bci=0
68        int [] validDays = { 0,  31, 29, 31, 30,

main[1] print dd
 dd = 1
main[1] stop in Date.toString
Set breakpoint Date.toString
main[1] list
64      *  if dd is legal day for current month, sets day to dd
65      *  otherwise, sets day to 1
66      */
67      private void setDay( int dd )   {
68 =>     int [] validDays = { 0,  31, 29, 31, 30,
69                           31, 30, 31, 31, 30,
70                           31, 30, 31 };
71        day = ( dd >= 1 && dd <= validDays[month] ? dd : 1 );
72      }
73      /** setMonth
main[1] cont
>
Breakpoint hit: "thread=main", Date.setDay(), line=68 bci=0
68        int [] validDays = { 0,  31, 29, 31, 30,

main[1] list
64      *  if dd is legal day for current month, sets day to dd
65      *  otherwise, sets day to 1
66      */
67      private void setDay( int dd )   {
68 =>     int [] validDays = { 0,  31, 29, 31, 30,
69                           31, 30, 31, 31, 30,
70                           31, 30, 31 };
71        day = ( dd >= 1 && dd <= validDays[month] ? dd : 1 );
72      }
73      /** setMonth
main[1] cont
>
Breakpoint hit: "thread=main", Date.setDay(), line=68 bci=0
68        int [] validDays = { 0,  31, 29, 31, 30,

main[1] step
>
Step completed: "thread=main", Date.setDay(), line=71 bci=76
71        day = ( dd >= 1 && dd <= validDays[month] ? dd : 1 );

main[1] exit

C:\Users\Ifrah\CSLABS>
```