

## Exercises

3. Suppose that `a` and `b` are `boolean` values. Show that the expression `(!(a && b) && (a || b)) || ((a && b) || !(a || b))` is equivalent to `true`.

`(False And True)`

*Or*

`(True && False)`

`False`

*Or*

`True`

**`True`**

4. Suppose that `a` and `b` are `int` values. Simplify the following expression: `(!(a < b) && !(a > b))`

same as `(a == b)`, only if values are equals the condition will be `true`

6. Why does `10/3` give 3 and not 3.33333333?

Java interprets an integer operation and gives an integer result.

7. What do each of the following print?

- `System.out.println(2 + "bc");` prints: 2bc it concatenates number 2 and string "bc" giving the string 2bc
- `System.out.println(2 + 3 + "bc");` prints: 5bc First it does an integer operation giving 5, then it concatenates with string "bc" giving the string 5bc
- `System.out.println((2+3) + "bc");` prints: 5bc prints: 5bc it concatenates the result of 2+3 encapsulated into (), and string "bc" giving the string 5bc
- `System.out.println("bc" + (2+3));` prints: bc5 The string "bc" concatenates with the result of integers between () giving an string result as "bc5"

- e. `System.out.println("bc" + 2 + 3);` prints: bc23 Integers are treated as strings due those are being added to the string "bc"

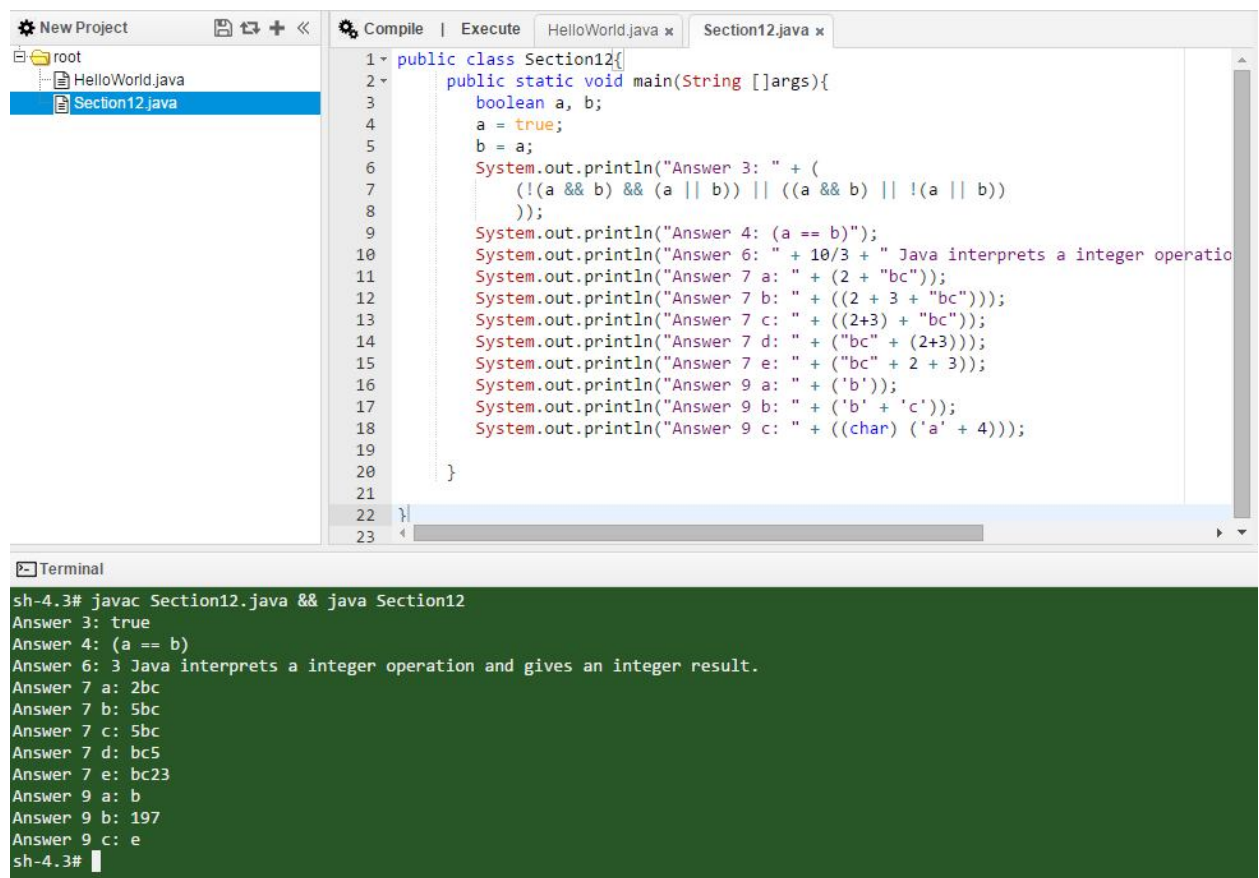
Explain each outcome.

9. What do each of the following print?

- a. `System.out.println('b');` Only print character b
- b. `System.out.println('b' + 'c');` Compiler interprets character operations as ascii decimal number , b equals to 98 and c 99 = 197
- c. `System.out.println((char) ('a' + 4));` a on ascii is 97 plus 4, and the (char) converts 101 to character = e on ascii code.

Explain each outcome.

A java program to test exercises 3..9:



The screenshot shows an IDE with a project named 'root' containing two files: 'HelloWorld.java' and 'Section12.java'. The 'Section12.java' file is open, showing the following code:

```
1 public class Section12{
2     public static void main(String []args){
3         boolean a, b;
4         a = true;
5         b = a;
6         System.out.println("Answer 3: " + (
7             !(a && b) && (a || b) || ((a && b) || !(a || b))
8         ));
9
10        System.out.println("Answer 4: (a == b)");
11        System.out.println("Answer 6: " + 10/3 + " Java interprets a integer operation");
12        System.out.println("Answer 7 a: " + (2 + "bc"));
13        System.out.println("Answer 7 b: " + ((2 + 3 + "bc")));
14        System.out.println("Answer 7 c: " + ((2+3) + "bc"));
15        System.out.println("Answer 7 d: " + ("bc" + (2+3)));
16        System.out.println("Answer 7 e: " + ("bc" + 2 + 3));
17        System.out.println("Answer 9 a: " + ('b'));
18        System.out.println("Answer 9 b: " + ('b' + 'c'));
19        System.out.println("Answer 9 c: " + ((char) ('a' + 4)));
20    }
21 }
22 }
23 }
```

The terminal output shows the results of running the program:

```
sh-4.3# javac Section12.java && java Section12
Answer 3: true
Answer 4: (a == b)
Answer 6: 3 Java interprets a integer operation and gives an integer result.
Answer 7 a: 2bc
Answer 7 b: 5bc
Answer 7 c: 5bc
Answer 7 d: bc5
Answer 7 e: bc23
Answer 9 a: b
Answer 9 b: 197
Answer 9 c: e
sh-4.3#
```

10. Suppose that a variable a is declared as `int a = 2147483647` (or equivalently, `Integer.MAX_VALUE`). What do each of the following print?

- a. `System.out.println(a);` 2147483647

- b. `System.out.println(a + 1);`    - 2147483648
- c. `System.out.println(2 - a);`    - 2147483645
- d. `System.out.println(-2 - a);`    2147483647
- e. `System.out.println(2 * a);`    -2
- f. `System.out.println(4 * a);`    -4

Explain each outcome.

The screenshot shows the CodingGround IDE interface. The top bar includes the logo, the text "(JDK 1.7.0)", and tabs for "New Project", "Compile", "Execute", and several open files: "HelloWorld.java", "DayOfWeek.java", "Section12part2.java", "SumOfTwoDice.java", and "Section12part3.java". The left sidebar shows a project tree with files like "DayOfWeek.class", "DayOfWeek.java", "HelloWorld.java", "Newfile.java", "Section12part2.class", "Section12part2.java", "SumOfTwoDice.class", "SumOfTwoDice.java", and "Section12part3.java". The main editor displays the code for "Section12part2.java":

```
1
2
3 public class Section12part2 {
4     public static void main(String[] args) {
5         int a = Integer.MAX_VALUE;
6         System.out.println("answer 10 a: " + a);
7         System.out.println("answer 10 b: " + (a + 1));
8         System.out.println("answer 10 c: " + (2 - a));
9         System.out.println("answer 10 d: " + (-2 - a));
10        System.out.println("answer 10 e: " + (2 * a));
11        System.out.println("answer 10 f: " + (4 * a));
12    }
13
14 }
```

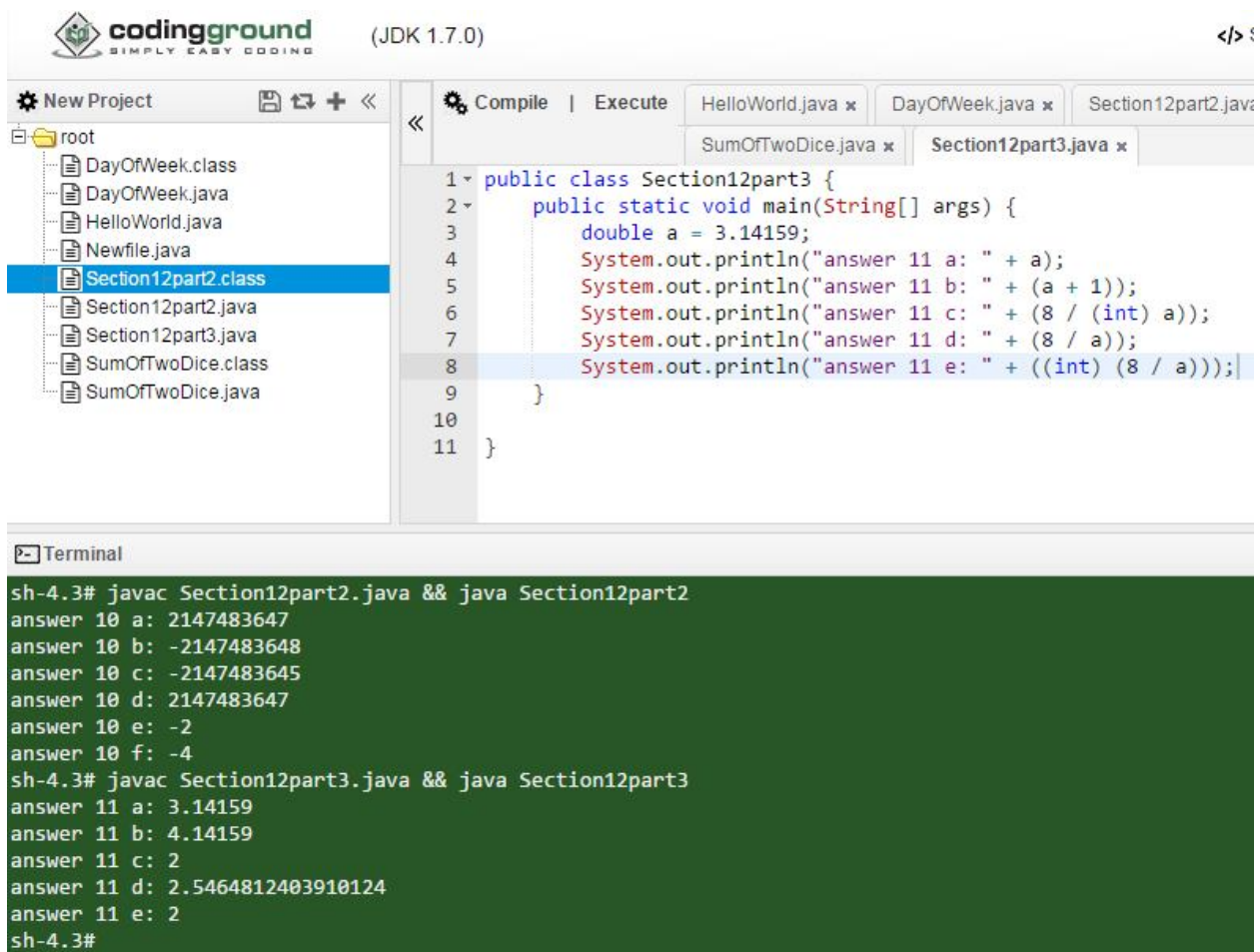
The bottom terminal window shows the execution output:

```
sh-4.3# javac Section12part2.java && java Section12part2
answer 10 a: 2147483647
answer 10 b: -2147483648
answer 10 c: -2147483645
answer 10 d: 2147483647
answer 10 e: -2
answer 10 f: -4
sh-4.3#
```

11. Suppose that a variable `a` is declared as `double a = 3.14159`. What do each of the following print?

- a. `System.out.println(a);` 3.14159 prints out assigned value
- b. `System.out.println(a + 1);` 4.14159 prints out a value plus 1
- c. `System.out.println(8 / (int) a);` 2 Divide 8 / converted a value to integer , it returns an integer due to an operation between integers.
- d. `System.out.println(8 / a);` 2.54648124039101 It returns a double due to an operation between doubles
- e. `System.out.println((int) (8 / a));` 2 It converts to integer the result of division 8/a

Explain each outcome.



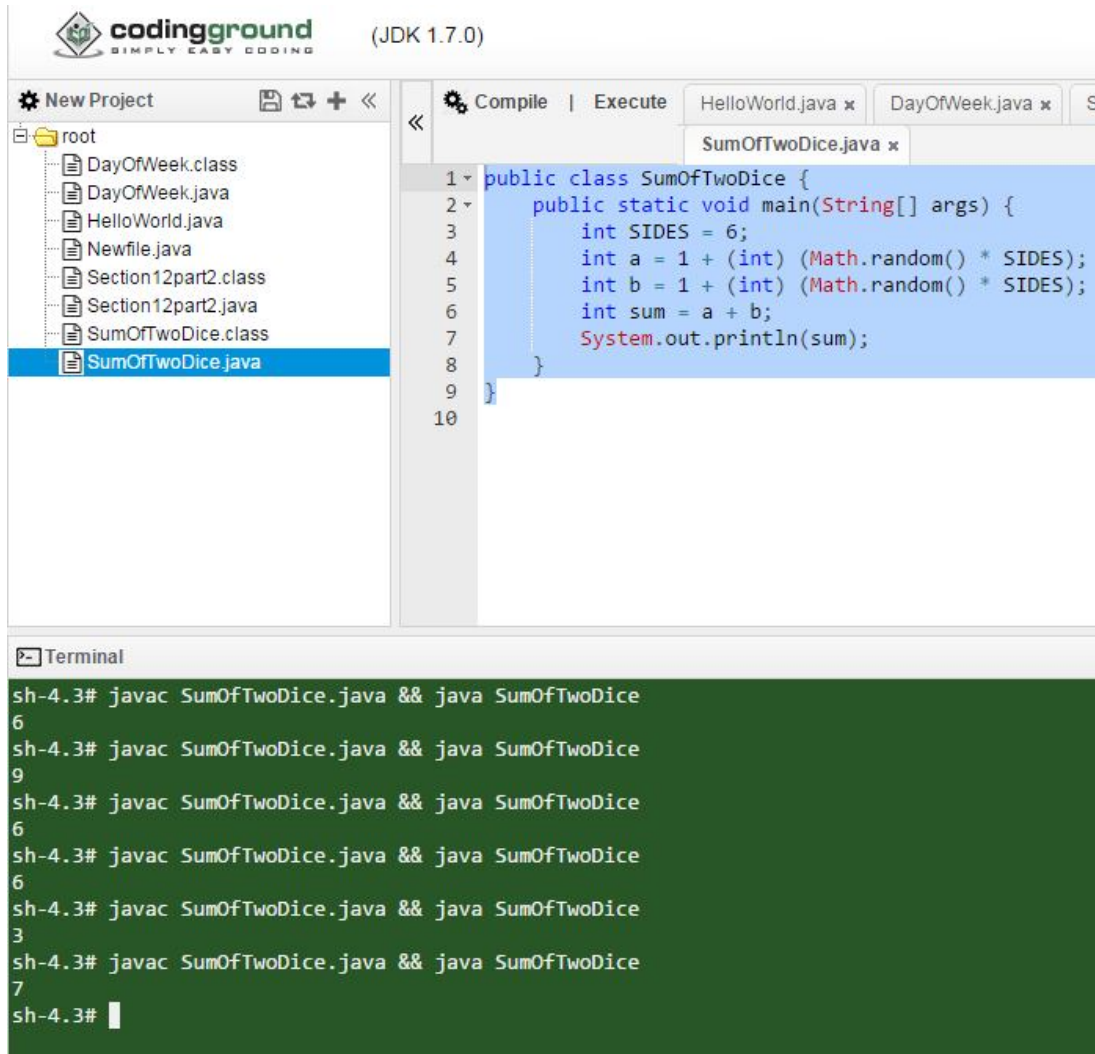
The screenshot shows the CodingGround IDE interface. The top bar includes the logo, version (JDK 1.7.0), and a code icon. The left sidebar shows a project structure with files like `DayOfWeek.class`, `DayOfWeek.java`, `HelloWorld.java`, `Newfile.java`, `Section12part2.class` (selected), `Section12part2.java`, `Section12part3.java`, `SumOfTwoDice.class`, and `SumOfTwoDice.java`. The main editor displays the code for `Section12part3.java`:

```
1 public class Section12part3 {
2     public static void main(String[] args) {
3         double a = 3.14159;
4         System.out.println("answer 11 a: " + a);
5         System.out.println("answer 11 b: " + (a + 1));
6         System.out.println("answer 11 c: " + (8 / (int) a));
7         System.out.println("answer 11 d: " + (8 / a));
8         System.out.println("answer 11 e: " + ((int) (8 / a)));
9     }
10 }
11 }
```

The bottom terminal window shows the execution results:

```
sh-4.3# javac Section12part2.java && java Section12part2
answer 10 a: 2147483647
answer 10 b: -2147483648
answer 10 c: -2147483645
answer 10 d: 2147483647
answer 10 e: -2
answer 10 f: -4
sh-4.3# javac Section12part3.java && java Section12part3
answer 11 a: 3.14159
answer 11 b: 4.14159
answer 11 c: 2
answer 11 d: 2.5464812403910124
answer 11 e: 2
sh-4.3#
```

20. Write a program SumOfTwoDice.java that prints the sum of two random integers between 1 and 6 (such as you might get when rolling dice).



The screenshot shows the CodingGround IDE interface. The top bar displays the logo and the text "(JDK 1.7.0)". The left sidebar shows a project structure with a "root" folder containing several files, including "SumOfTwoDice.java" which is selected. The main editor window shows the code for "SumOfTwoDice.java" with line numbers 1 through 10. The code is as follows:

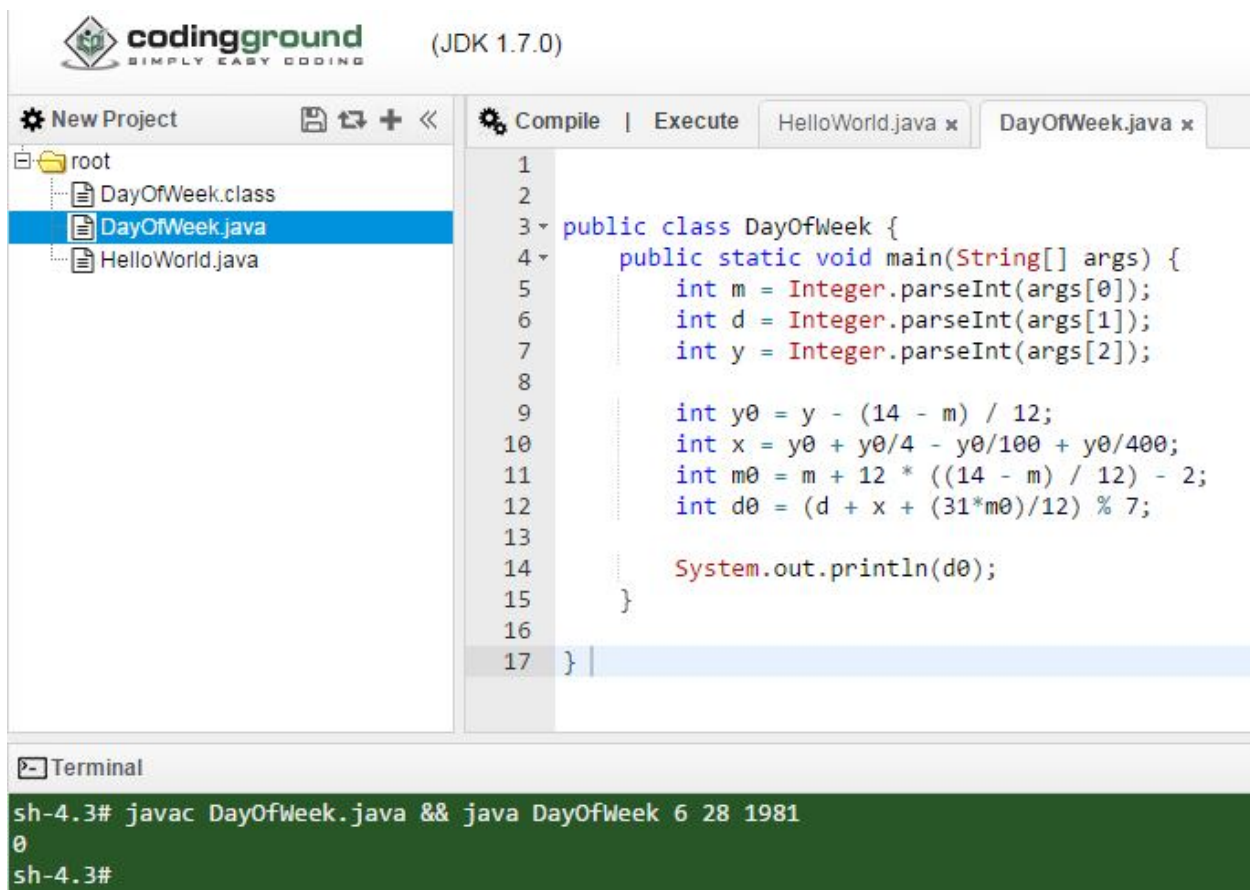
```
1 public class SumOfTwoDice {  
2     public static void main(String[] args) {  
3         int SIDES = 6;  
4         int a = 1 + (int) (Math.random() * SIDES);  
5         int b = 1 + (int) (Math.random() * SIDES);  
6         int sum = a + b;  
7         System.out.println(sum);  
8     }  
9 }  
10
```

The bottom panel shows the terminal output, which displays the results of running the program multiple times. The output shows the sum of two random integers between 1 and 6, such as 6, 9, 6, 6, 3, and 7.

```
sh-4.3# javac SumOfTwoDice.java && java SumOfTwoDice  
6  
sh-4.3# javac SumOfTwoDice.java && java SumOfTwoDice  
9  
sh-4.3# javac SumOfTwoDice.java && java SumOfTwoDice  
6  
sh-4.3# javac SumOfTwoDice.java && java SumOfTwoDice  
6  
sh-4.3# javac SumOfTwoDice.java && java SumOfTwoDice  
3  
sh-4.3# javac SumOfTwoDice.java && java SumOfTwoDice  
7  
sh-4.3#
```

## Creative Exercises

**29. Day of the week.** Write a program [DayOfWeek.java](#) that takes a date as input and prints the day of the week that date falls on. Your program should take three command-line arguments:  $m$  (month),  $d$  (day), and  $y$  (year). For  $m$  use 1 for January, 2 for February, and so forth. For output print 0 for Sunday, 1 for Monday, 2 for Tuesday, and so forth. Use the following [formulas](#), for the [Gregorian calendar](#):



The screenshot shows the codingground IDE with the following components:

- Header:** codingground SIMPLY EASY CODING (JDK 1.7.0)
- Left Panel (File Explorer):** Shows a project structure with a 'root' folder containing 'DayOfWeek.class', 'DayOfWeek.java' (selected), and 'HelloWorld.java'.
- Top Bar:** Includes 'New Project', 'Compile', 'Execute', and tabs for 'HelloWorld.java' and 'DayOfWeek.java'.
- Editor:** Displays the code for 'DayOfWeek.java' with line numbers 1 through 17. The code implements the algorithm for finding the day of the week based on the provided formulas.
- Terminal:** Shows the command 'sh-4.3# javac DayOfWeek.java && java DayOfWeek 6 28 1981' and the output '0', followed by the prompt 'sh-4.3#'.

```
1
2
3 public class DayOfWeek {
4     public static void main(String[] args) {
5         int m = Integer.parseInt(args[0]);
6         int d = Integer.parseInt(args[1]);
7         int y = Integer.parseInt(args[2]);
8
9         int y0 = y - (14 - m) / 12;
10        int x = y0 + y0/4 - y0/100 + y0/400;
11        int m0 = m + 12 * ((14 - m) / 12) - 2;
12        int d0 = (d + x + (31*m0)/12) % 7;
13
14        System.out.println(d0);
15    }
16
17 }
```

Terminal output:

```
sh-4.3# javac DayOfWeek.java && java DayOfWeek 6 28 1981
0
sh-4.3#
```