

```
// Die.java
// Carl Owenby
// October 3, 2005
// Javadoc comments omitted for demonstration

import java.util.Random;

public class Die
{
    // FIELDS
    private Random rand; // Create an object variable (object reference) of type
Random
    private int sides; // Create a variable of type int

    // CONSTRUCTORS
    public Die(int s)
    {
        // initialise instance variables
        sides = s; // Initialize the number of sides to the value passed in the
parameter s
        rand = new Random(); // Create a new object of type Random
    }

    public Die()
    {
        // If no parameter is passed, create a die with 6 sidea
        this(6); // call constructor shown above with parameter 6 (sides)
    }

    // METHODS
    public int getSides() // ACCESSOR METHOD
    {
        return sides;
    }

    public void setSides(int s) // MUTATOR METHOD
    {
        sides = s;
    }

    public int roll()
    {
        // Roll the die
        return 1 + rand.nextInt(sides);
        // If the number of sides is 6
        // rand.nextInt(sides) returns a random number between 0 and 5
        // adding 1 returns a value between 1 and 6 (inclusive)
    }
}
```

```
// DieTester.java
// Carl Owenby
// October 3, 2005
// Javadoc comments omitted for demonstration

public class DieTester
{
    static int o1, o2, o3, o4, o5, o6; // variables to count the number of
occurrences for the roll of dOne
    static int t1, t2, t3, t4, t5, t6; // variables to count the number of
occurrences for the roll of dTwo
    static int s2, s3, s4, s5, s6, s7, s8, s9, s10, s11, s12; // Number of
occurrences for sum of rolls one and two

    public static void main(String[] args)
    {
        final int SIDES = 6; // NAMED CONSTANT

        // The following code creates the object variable and the new object in
separate statements
        Die d8, d12; // Create two object variables of type Die
        d8 = new Die(8); // parameter 8 creates a die with 8 sides
        System.out.println("Create new object d12");
        d12 = new Die(); // no parameter creates a die with the default 6 sides
        System.out.printf("d12 has %d sides by default since no parameter was
passed\n", d12.getSides());
        d12.setSides(12);
        System.out.printf("d12 now has %d sides after setSides method (mutator)
called\n\n", d12.getSides());

        // The following code creates the object variable and the new object in a
single statement
        Die dOne = new Die(SIDES); // Create a new object of type Die
        Die dTwo = new Die(SIDES); // Create another new object of type Die

        System.out.println("Printing the numbers that are rolled");
        while (s7 < 6)
        {
            int firstRoll = dOne.roll(); // Roll the first die
            int secondRoll = dTwo.roll(); // Roll the second die
            countOccurrences(firstRoll, secondRoll);
            int sum = firstRoll + secondRoll; // Add the value of the first and
second rolls
            System.out.printf("%2d ", sum); // Print the number rolled (2-12)
        }
        System.out.printf("\n\ns7 = %d after first while loop executes\n", s7);
        int sum = s2 + s3 + s4 + s5 + s6 + s7 + s8 + s9 + s10 + s11 + s12;
        System.out.printf("Rolled %d 7's in %d tries\n", s7, sum);

        // while loop that uses a boolean variable as the condition for looping
        boolean done = false;
        int i = 0;
        while (!done)
        {
            int firstRoll = dOne.roll();
            int secondRoll = dTwo.roll();
            countOccurrences(firstRoll, secondRoll);
            i++;
            if (i >= 100000)
                done = true; // exit condition
        }
    }
}
```

```

    }
    System.out.printf("\ni = %d after second while loop executes\n", i);

    // while loop that uses a relational operator in the condition
    i = 0;
    while (i < 100000)
    {
        // i = 99999 as it loops for the last time
        int firstRoll = dOne.roll();
        int secondRoll = dTwo.roll();
        countOccurrences(firstRoll, secondRoll);
        i++;
        // i = 100000 as it exits the loop
        // loop executes 100000 times
    }
    System.out.printf("\ni = %d after third while loop executes\n", i);

    // do loop that uses a relational operator in the condition
    // always executes the body of the loop at least once
    // the condition is tested at the end of the loop
    i = 0;
    do
    {
        // i = 99999 as it loops for the last time
        int firstRoll = dOne.roll();
        int secondRoll = dTwo.roll();
        countOccurrences(firstRoll, secondRoll);
        i++;
        // i = 100000 as it exits the loop
        // loop executes 100000 times
    }
    while (i < 100000);
    System.out.printf("\ni = %d after do loop executes\n", i);

    // for loop - Note initialization, condition and update handled outside the
block
    for (int j = 1; j <= 100000; j++)
    {
        int firstRoll = dOne.roll();
        int secondRoll = dTwo.roll();
        countOccurrences(firstRoll, secondRoll);
    }
    System.out.printf("\nj does not exist outside the for loop, it is out-of-
scope\n\n");

    int oSum = o1 + o2 + o3 + o4 + o5 + o6;
    int tSum = t1 + t2 + t3 + t4 + t5 + t6;
    System.out.printf("Occurrences %7d%7d%7d%7d%7d%7d Total \n", 1, 2, 3, 4, 5,
6);
    System.out.printf("First Roll %7d%7d%7d%7d%7d%7d\n", o1, o2, o3, o4, o5,
o6, oSum);
    System.out.printf("Second Roll %7d%7d%7d%7d%7d%7d\n\n", t1, t2, t3, t4, t5,
t6, tSum);

    System.out.printf("Percentage %7d%7d%7d%7d%7d%7d\n", 1, 2, 3, 4, 5, 6);
    System.out.printf("First Roll %7.3f%7.3f%7.3f%7.3f%7.3f%7.3f\n",
100.0*o1/oSum, 100.0*o2/oSum, 100.0*o3/oSum, 100.0*o4/oSum, 100.0*o5/oSum,
100.0*o6/oSum);

```

```

        System.out.printf("Second Roll %7.3f%7.3f%7.3f%7.3f%7.3f%7.3f\n",
100.0*t1/tSum, 100.0*t2/tSum, 100.0*t3/tSum, 100.0*t4/tSum, 100.0*t5/tSum,
100.0*t6/tSum);
        System.out.printf("Expected %7.3f%7.3f%7.3f%7.3f%7.3f%7.3f\n\n", 100.0/6,
100.0/6, 100.0/6, 100.0/6, 100.0/6);

        int sSum = s2 + s3 + s4 + s5 + s6 + s7 + s8 + s9 + s10 + s11 + s12;
        System.out.printf("Sum %6d%6d%6d%6d%6d%6d%6d%6d%6d%6d\n", 2, 3, 4,
5, 6, 7, 8, 9, 10, 11, 12);
        System.out.printf("Occurences %6d%6d%6d%6d%6d%6d%6d%6d%6d%6d\n", s2, s3,
s4, s5, s6, s7, s8, s9, s10, s11, s12);

        System.out.printf("Percentage %6.2f%6.2f%6.2f%6.2f%6.2f", 100.0*s2/sSum,
100.0*s3/sSum, 100.0*s4/sSum, 100.0*s5/sSum, 100.0*s6/sSum);
        System.out.printf("%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f\n", 100.0*s7/sSum,
100.0*s8/sSum, 100.0*s9/sSum, 100.0*s10/sSum, 100.0*s11/sSum, 100.0*s12/sSum);

        System.out.printf("Expected
%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f%6.2f\n", 100.0*1/36, 100.0*2/36,
100.0*3/36, 100.0*4/36, 100.0*5/36, 100.0*6/36, 100.0*5/36, 100.0*4/36, 100.0*3/36,
100.0*2/36, 100.0*1/36);

        System.out.println("\nNow counting to one billion...");
        i = 0;
        do
        {
            i++;
            if (i % 100000000 == 0)
                System.out.printf("i = %4d million\n", i / 1000000);
        }
        while (i <= 1000000000);
        System.out.printf("DONE!!!\ni = %d", i);
    }

    public static void countOccurrences(int r1, int r2)
    {
        // Count the occurrences of roll one USING IF STATEMENTS
        if (r1 == 1)
            o1++;
        else if (r1 == 2)
            o2++;
        else if (r1 == 3)
            o3++;
        else if (r1 == 4)
            o4++;
        else if (r1 == 5)
            o5++;
        else if (r1 == 6)
            o6++;
        else
            System.out.println("Error: parameter r1 out of expected range");

        // Count the occurrences of roll two USING SWITCH STATEMENT
        switch (r2)
        {
            // Note break statements required otherwise execution will continue to
            next case (fall-through)
            case 1: t1++; break;
            case 2: t2++; break;
            case 3: t3++; break;

```

```
        case 4: t4++; break;
        case 5: t5++; break;
        case 6: t6++; break;
        default: System.out.println("Error: parameter r2 out of expected range");
    }

    // Count the occurrences for the sum r1 + r2 USING SWITCH STATEMENT
    switch (r1 + r2)
    {
        case 2: s2++; break;
        case 3: s3++; break;
        case 4: s4++; break;
        case 5: s5++; break;
        case 6: s6++; break;
        case 7: s7++; break;
        case 8: s8++; break;
        case 9: s9++; break;
        case 10: s10++; break;
        case 11: s11++; break;
        case 12: s12++; break;
        default: System.out.println("Error: sum r1 + r2 out of expected range");
    }
}
```

```
Create new object dl2
dl2 has 6 sides by default since no parameter was passed
dl2 now has 12 sides after setSides method (mutator) called
```

```
Printing the numbers that are rolled
```

```
8 5 3 4 11 8 12 6 8 7 6 9 8 7 12 9 7 6 7 8 8 6 5 3 7 9 9 7
```

```
s7 = 6 after first while loop executes
Rolled 6 7's in 28 tries
```

```
i = 100000 after second while loop executes
```

```
i = 100000 after third while loop executes
```

```
i = 100000 after do loop executes
```

```
j does not exist outside the for loop, it is out-of-scope
```

Occurences	1	2	3	4	5	6	Total
First Roll	66514	66589	66805	66377	66628	67115	400028
Second Roll	66385	66386	66625	66954	67023	66655	400028

Percentage	1	2	3	4	5	6
First Roll	16.627	16.646	16.700	16.593	16.656	16.778
Second Roll	16.595	16.595	16.655	16.737	16.755	16.663
Expected	16.667	16.667	16.667	16.667	16.667	16.667

Sum	2	3	4	5	6	7	8	9	10	11	12
Occurences	11019	22240	32949	44421	55819	66239	55816	44652	33181	22418	11274
Percentage	2.75	5.56	8.24	11.10	13.95	16.56	13.95	11.16	8.29	5.60	2.82
Expected	2.78	5.56	8.33	11.11	13.89	16.67	13.89	11.11	8.33	5.56	2.78

```
Now counting to one billion...
```

```
i = 100 million
i = 200 million
i = 300 million
i = 400 million
i = 500 million
i = 600 million
i = 700 million
i = 800 million
i = 900 million
i = 1000 million
DONE!!!
i = 1000000001
```