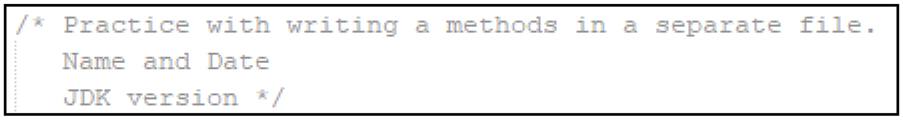
Writing Calculations Exercise

Our goal is to write a random method that will take in 2 integers to create a random number. We will use this method to generate 2 random numbers and then we will prompt the user for a mathematical operation (addition, subtraction, multiplication, or division). The equations will be handled by having separate methods for each operaor. We will display the answer to the equation.

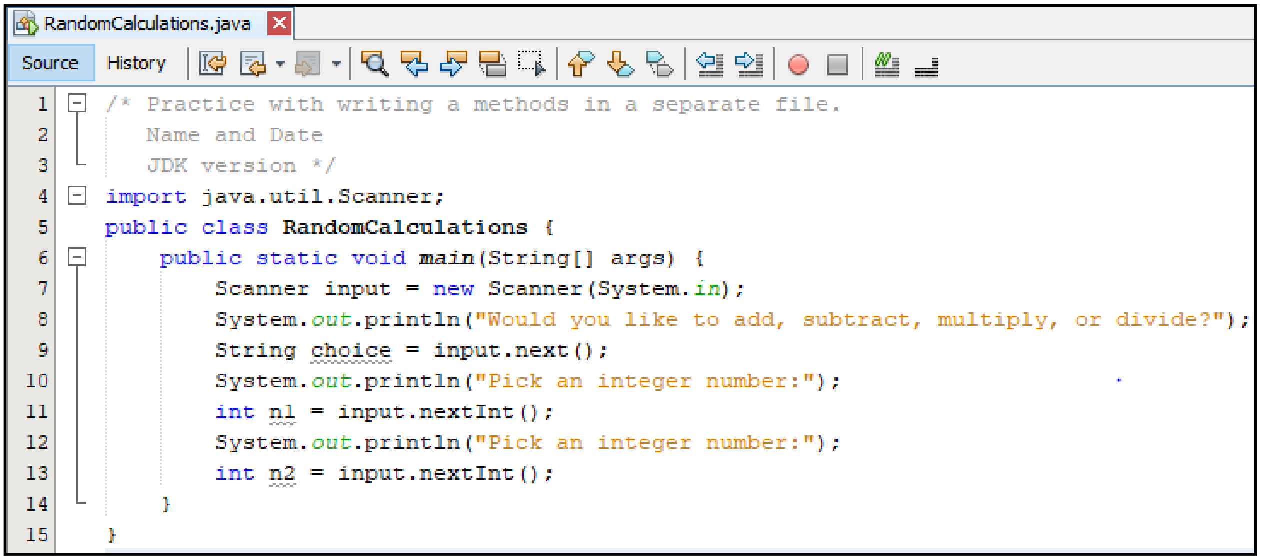
1. Get into NetBeans and create a new Java project called **RandomCalculations**. Be sure to save this project to the appropriate exercise folder. Do not create a Package statement.

2. Add comments to your program as follows:



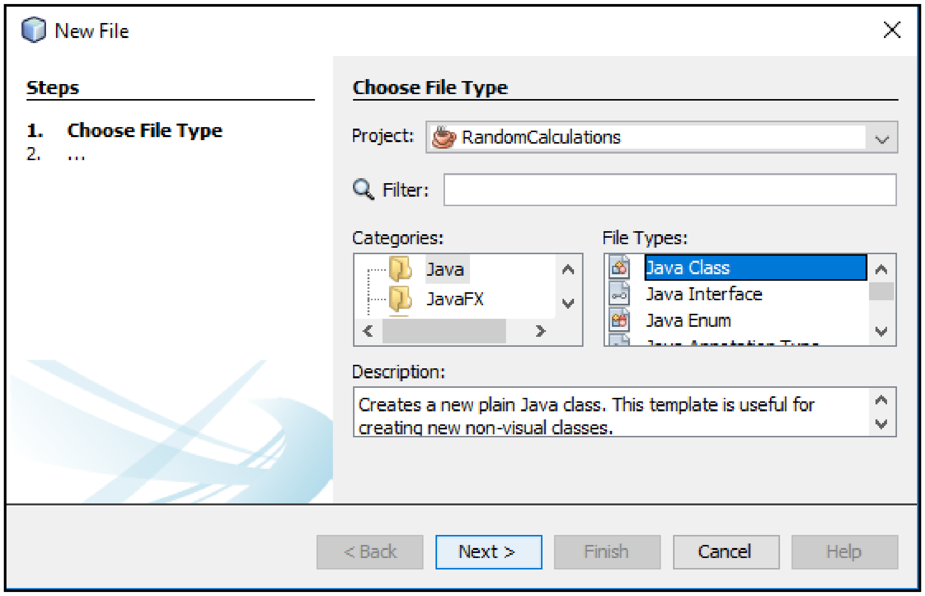
```
/* Practice with writing a methods in a separate file.
   Name and Date
   JDK version */
```

3. Let’s take in the data from the user. We will need to set up our Scanner and prompt the user for what operation they would like to use for the equation and then what numbers they would like to use for the upper and lower bounds of our generated random numbers.



```
/* Practice with writing a methods in a separate file.
   Name and Date
   JDK version */
import java.util.Scanner;
public class RandomCalculations {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Would you like to add, subtract, multiply, or divide?");
        String choice = input.next();
        System.out.println("Pick an integer number:");
        int n1 = input.nextInt();
        System.out.println("Pick an integer number:");
        int n2 = input.nextInt();
    }
}
```

4. The random number generation will be done in a separate method in a new class file. Let’s create the new file. We could write our methods in this file or we could separate them into a new class to make it more reusable. Choose **File**, **New File**, **Java Class**, and click **Next**.



Choose File Type

Project: RandomCalculations

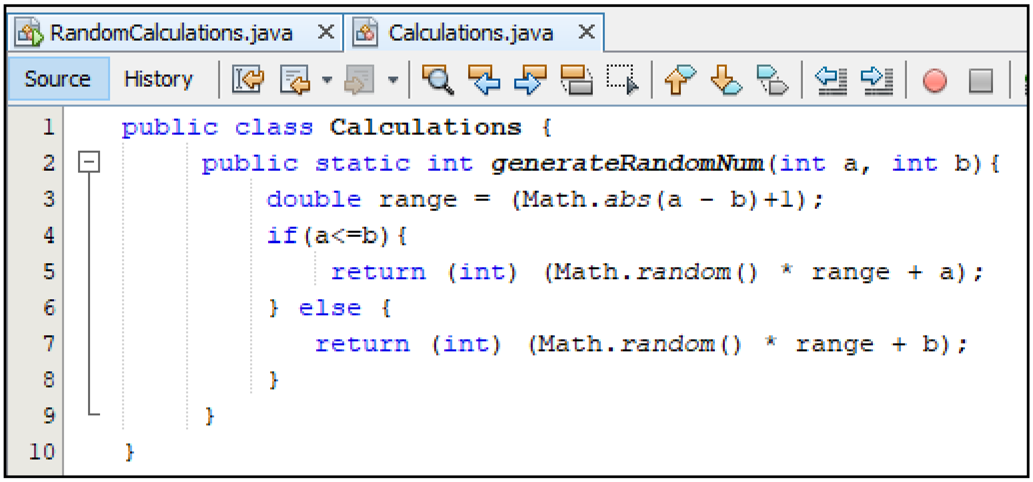
Filter:

Description:

Creates a new plain Java class. This template is useful for creating new non-visual classes.

5. Name this class file **Calculations** and click **Finish**.

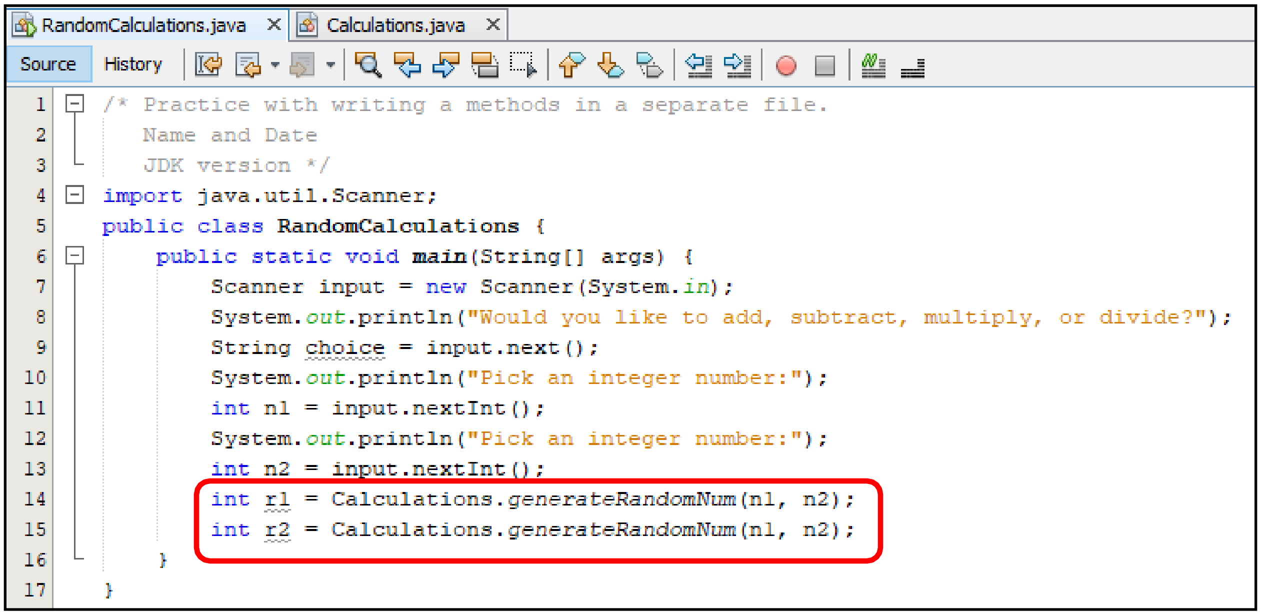
6. Add the following code to the **Calculations** class. The **generateRandomNum** method will take in 2 integers and return a random integer number. Inside the method, we will subtract the 2 in coming integers and use the absolute value with this equation to ensure that the range is not a negative number. We will add 1 to this result to get the range and save this range into a variable named **range**. Because we are not sure whether the first integer **a** or second integer **b** is greater, we will need to create a conditional to make this determination. If the integer **a** is less than or equal to integer **b**, then we will add that number to our range, otherwise we will use **b** to add to our range. We will use Java's random method to generate a random number between 0 and 1 and then we will multiply that times our range and add the smaller of the 2 integers. We need to cast this number to an integer to get rid of the decimal part of the generated number since we want to use integers in our equations.



```
public class Calculations {
    public static int generateRandomNum(int a, int b){
        double range = (Math.abs(a - b)+1);
        if(a<=b){
            return (int) (Math.random() * range + a);
        } else {
            return (int) (Math.random() * range + b);
        }
    }
}
```

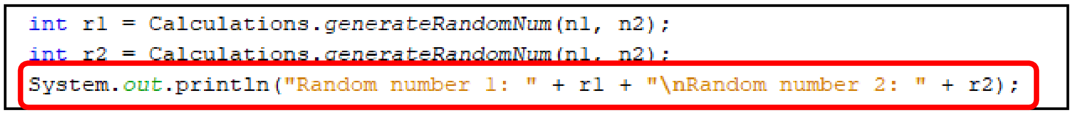
7. If you run the program nothing will happen, because even though we have created a **generateRandomNum** method, we have not called it.

8. We need to go back to the **RandomCalculations** class which contains our main method and call our new method from here. When we call our **generateRandomNum** method we will need to use the name of the class before the name of the method since our method is located in another class. Also, we will need to save the result from this method into a variable to be used later.



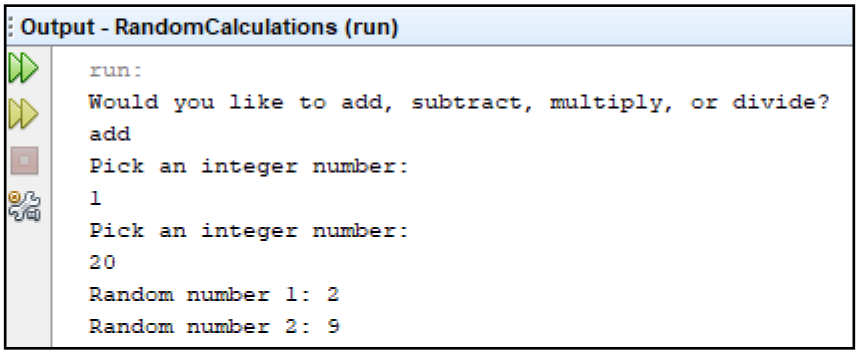
```
/* Practice with writing a methods in a separate file.
   Name and Date
   JDK version */
import java.util.Scanner;
public class RandomCalculations {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Would you like to add, subtract, multiply, or divide?");
        String choice = input.next();
        System.out.println("Pick an integer number:");
        int n1 = input.nextInt();
        System.out.println("Pick an integer number:");
        int n2 = input.nextInt();
        int r1 = Calculations.generateRandomNum(n1, n2);
        int r2 = Calculations.generateRandomNum(n1, n2);
    }
}
```

9. We won't know if this method worked correctly if we run the program since the random numbers our printed yet. Let's put a println statement in the program for testing purposes.



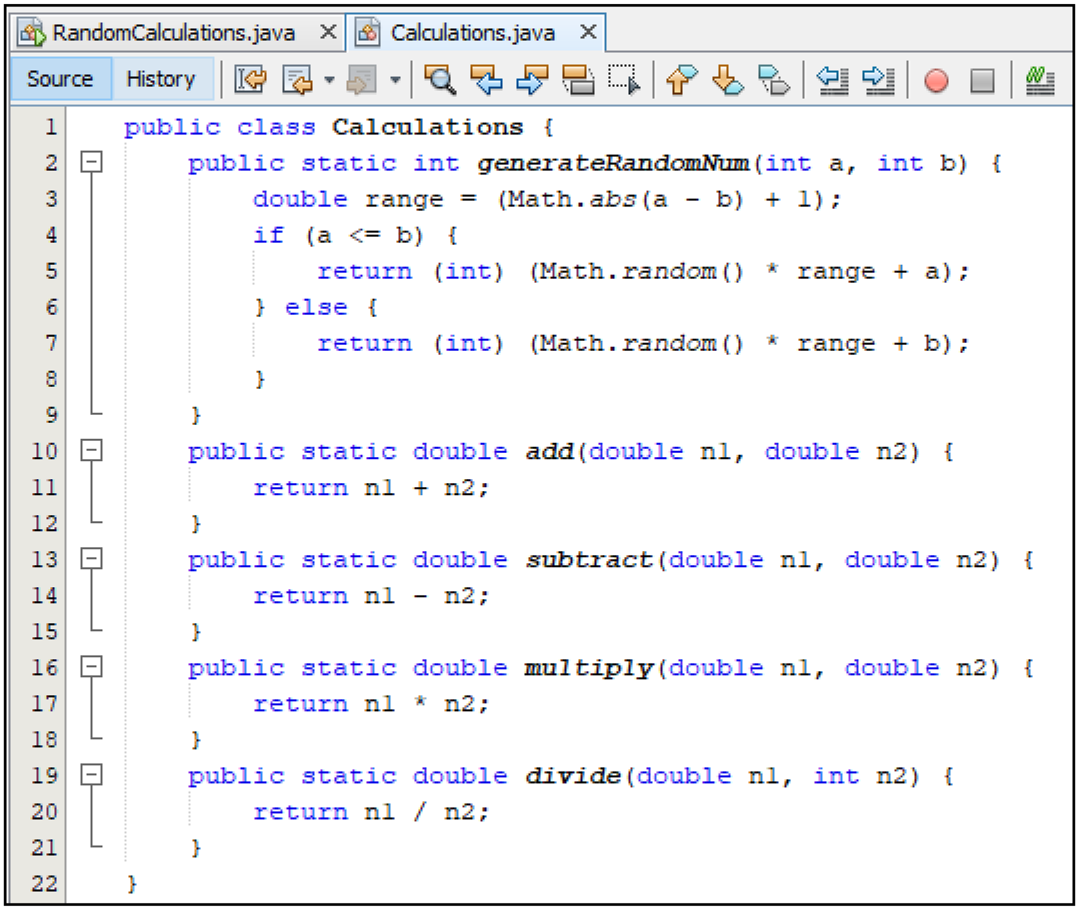
```
int r1 = Calculations.generateRandomNum(n1, n2);
int r2 = Calculations.generateRandomNum(n1, n2);
System.out.println("Random number 1: " + r1 + "\nRandom number 2: " + r2);
```

10. Run your program a couple of times using different integers to test whether the random numbers that are generated are in the range.
Example:



```
run:
Would you like to add, subtract, multiply, or divide?
add
Pick an integer number:
1
Pick an integer number:
20
Random number 1: 2
Random number 2: 9
```

11. Let’s add our **add**, **subtract**, **multiply**, and **divide** methods to our **Calculations** class file. Each of these methods will need to take in 2 numbers. Even though our random numbers will be integers, it would be nice to allow our new methods to take in doubles just in case we want to use them in the future. Using doubles as the parameter types for our divide method will also help to eliminate fix integer division issues.

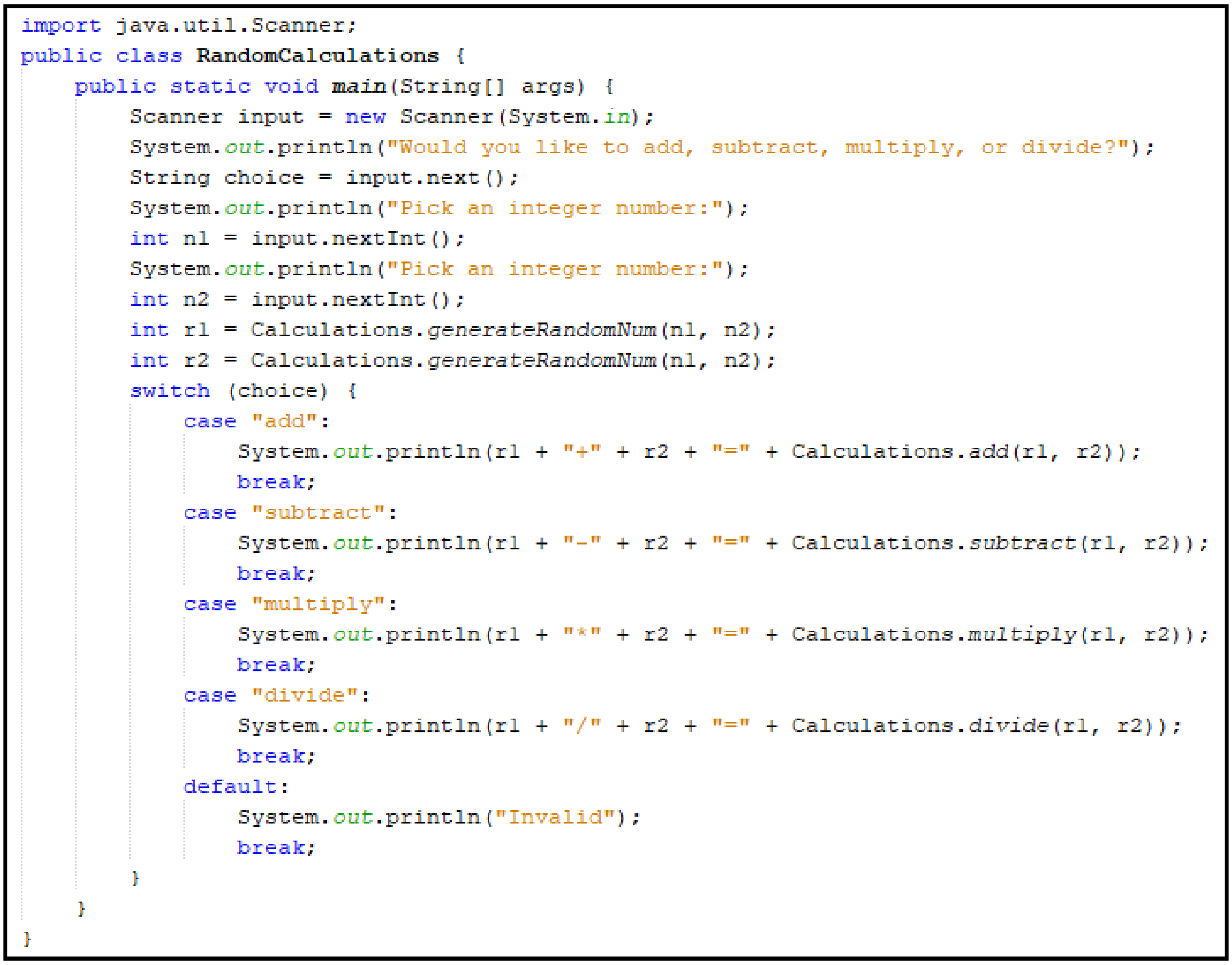


```java
public class Calculations {
    public static int generateRandomNum(int a, int b) {
        double range = (Math.abs(a - b) + 1);
        if (a <= b) {
            return (int) (Math.random() * range + a);
        } else {
            return (int) (Math.random() * range + b);
        }
    }
    public static double add(double n1, double n2) {
        return n1 + n2;
    }
    public static double subtract(double n1, double n2) {
        return n1 - n2;
    }
    public static double multiply(double n1, double n2) {
        return n1 * n2;
    }
    public static double divide(double n1, int n2) {
        return n1 / n2;
    }
}
```

12. Now, we need to go back to the **RandomCalculations** class and call our new methods. We aren’t going to call all of our new methods, since we only want to call the method of the operation that the user chose. Before we code our switch statement to decide which operator to use, let’s erase the following testing line.

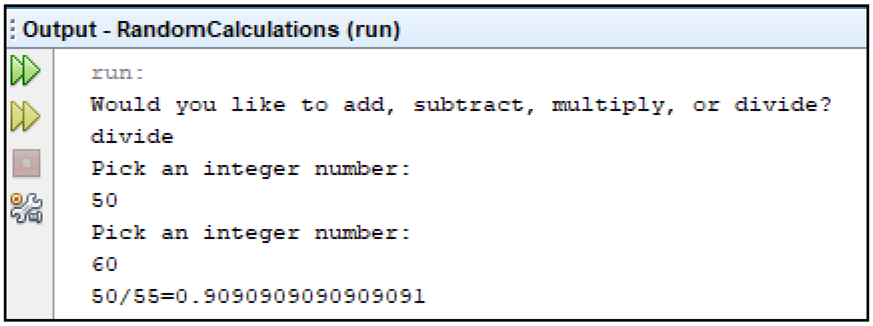
System.out.println("Random number 1: " + r1 + "\nRandom number 2: " + r2);

13. Next, we will add our switch statement. Our switch will determine which operation the user chose and then it will call the correct method from the **Calcuations** file using our random number and will print the result.



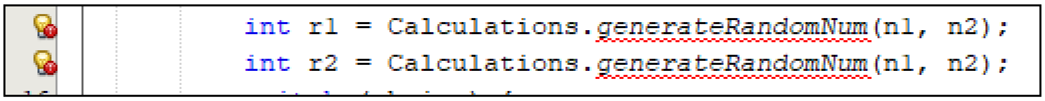
```java
import java.util.Scanner;
public class RandomCalculations {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Would you like to add, subtract, multiply, or divide?");
        String choice = input.next();
        System.out.println("Pick an integer number:");
        int n1 = input.nextInt();
        System.out.println("Pick an integer number:");
        int n2 = input.nextInt();
        int r1 = Calculations.generateRandomNum(n1, n2);
        int r2 = Calculations.generateRandomNum(n1, n2);
        switch (choice) {
            case "add":
                System.out.println(r1 + "+" + r2 + "=" + Calculations.add(r1, r2));
                break;
            case "subtract":
                System.out.println(r1 + "-" + r2 + "=" + Calculations.subtract(r1, r2));
                break;
            case "multiply":
                System.out.println(r1 + "*" + r2 + "=" + Calculations.multiply(r1, r2));
                break;
            case "divide":
                System.out.println(r1 + "/" + r2 + "=" + Calculations.divide(r1, r2));
                break;
            default:
                System.out.println("Invalid");
                break;
        }
    }
}
```

14. Now test the program to see if it works. Run it multiple times, choosing different integers and different operations.



```
run:
Would you like to add, subtract, multiply, or divide?
divide
Pick an integer number:
50
Pick an integer number:
60
50/55=0.9090909090909091
```

15. Let’s do some experimenting. What if we made the **generateRandomNum** method in the **Calculations** class **private**? This should give you an error in your program. Making this method private means that it would have to be called from the class that it is located in. Since it is not located in the same file that you called it from, you get an error. Change this method back to **public**.



```
        int r1 = Calculations.generateRandomNum(n1, n2);
        int r2 = Calculations.generateRandomNum(n1, n2);
```

16. Run your program to test that it still works. Close the program and exit NetBeans.