**Name- Khushi Nitinkumar Patel**

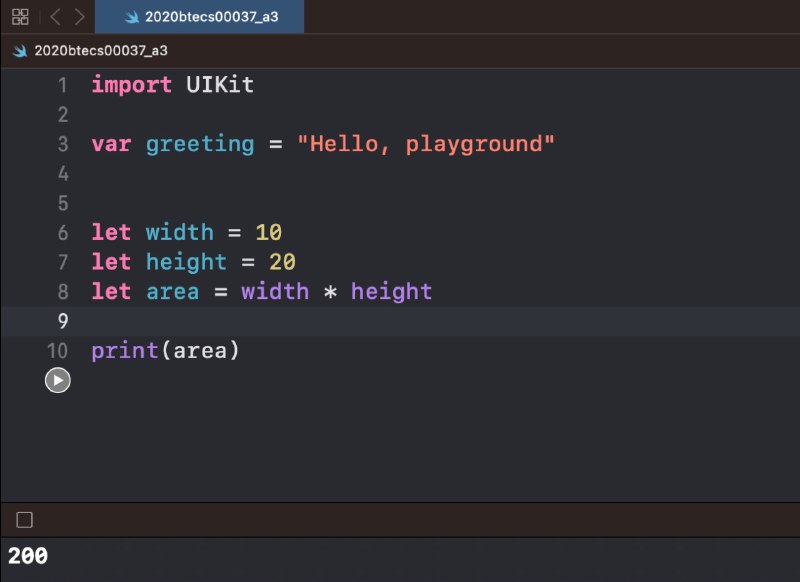
**PRN-2020BTECS00037**

**Batch- T2**

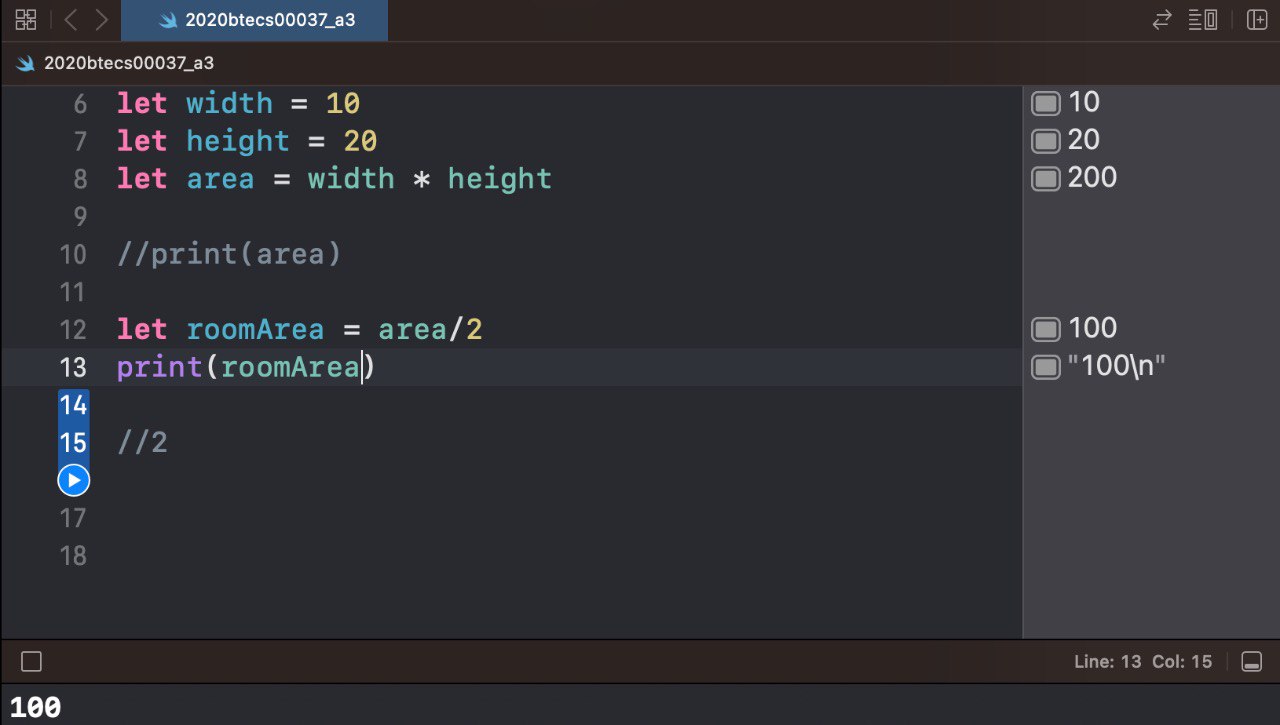
**Assignment no – 3**

## Exercise - Basic Arithmetic

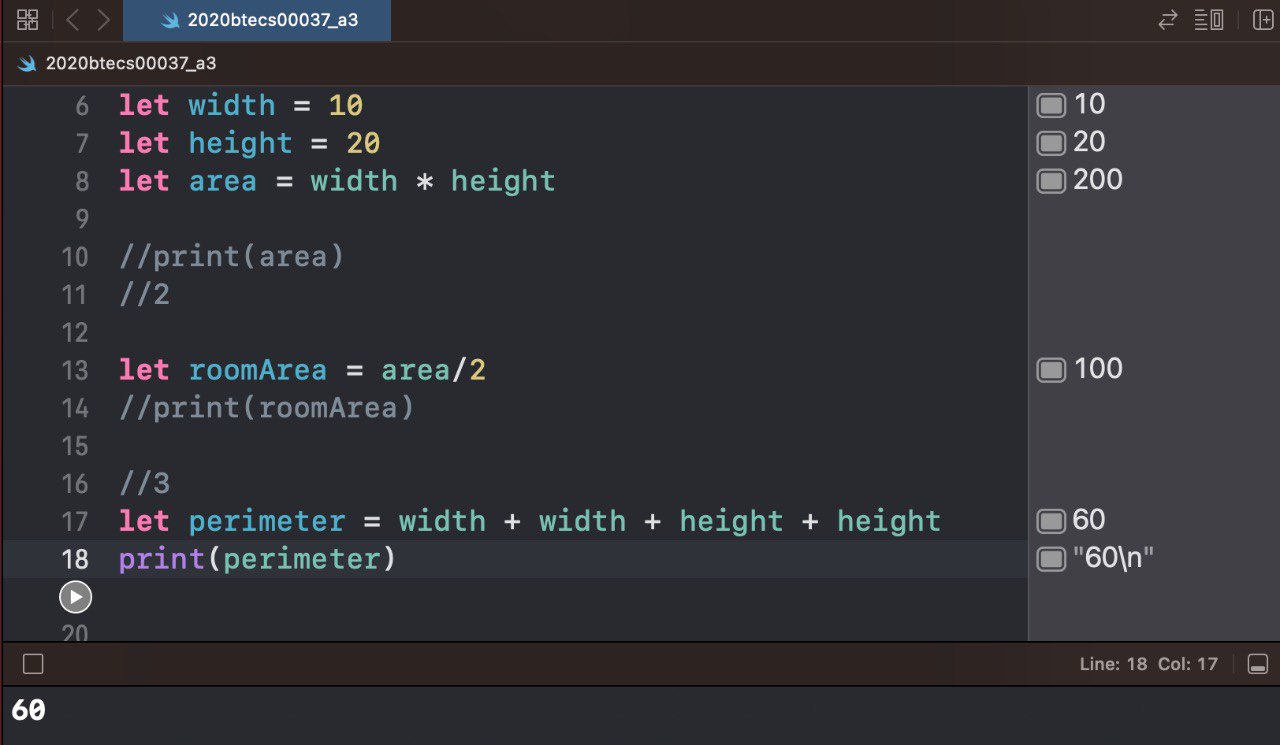
1. You decide to build a shed and want to know beforehand the area of your yard that it will take up. Create two constants, `width` and `height`, with values of 10 and 20, respectively. Create an `area` constant that is the result of multiplying the two previous constants together, and print out the result.



1. You decide that you'll divide your shed into two rooms. You want to know if dividing it equally will leave enough room for some of your larger storage items. Create a `roomArea` constant that is the result of dividing `area` in half. Print out the result.

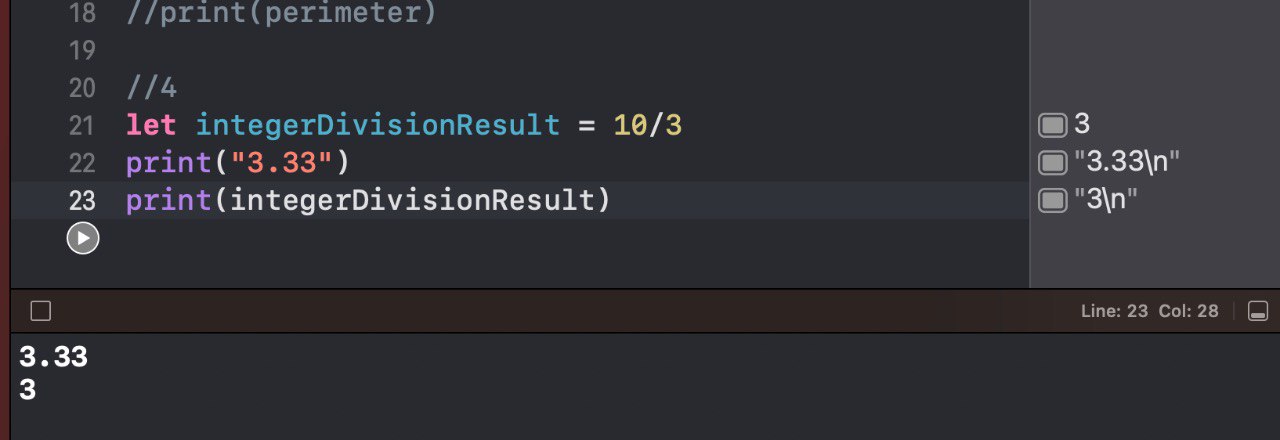


1. Create a `perimeter` constant whose value equals `width` plus `width` plus `height` plus `height`, then print out the result.



1. Print what you would expect the result of integer division of 10 divided by 3 to be. Create a constant,

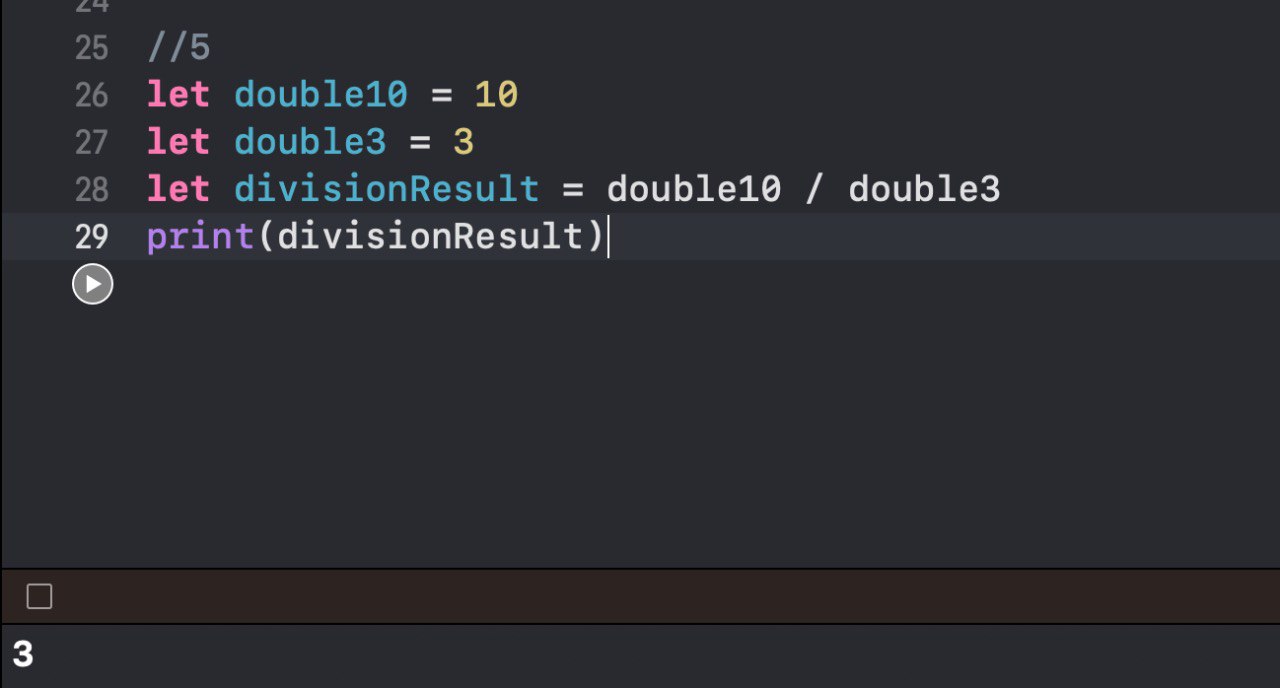
`integerDivisionResult` that is the result of 10 divided by 3, and print the value.



1. Now create two constants, `double10` and `double3`, set to 10 and 3, and declare their types as

`Double` values. Declare a final constant `divisionResult` equal to the result of `double10` divided by

`double3`. Print the value of `divisionResult`. How does this differ from the value when using integer division?

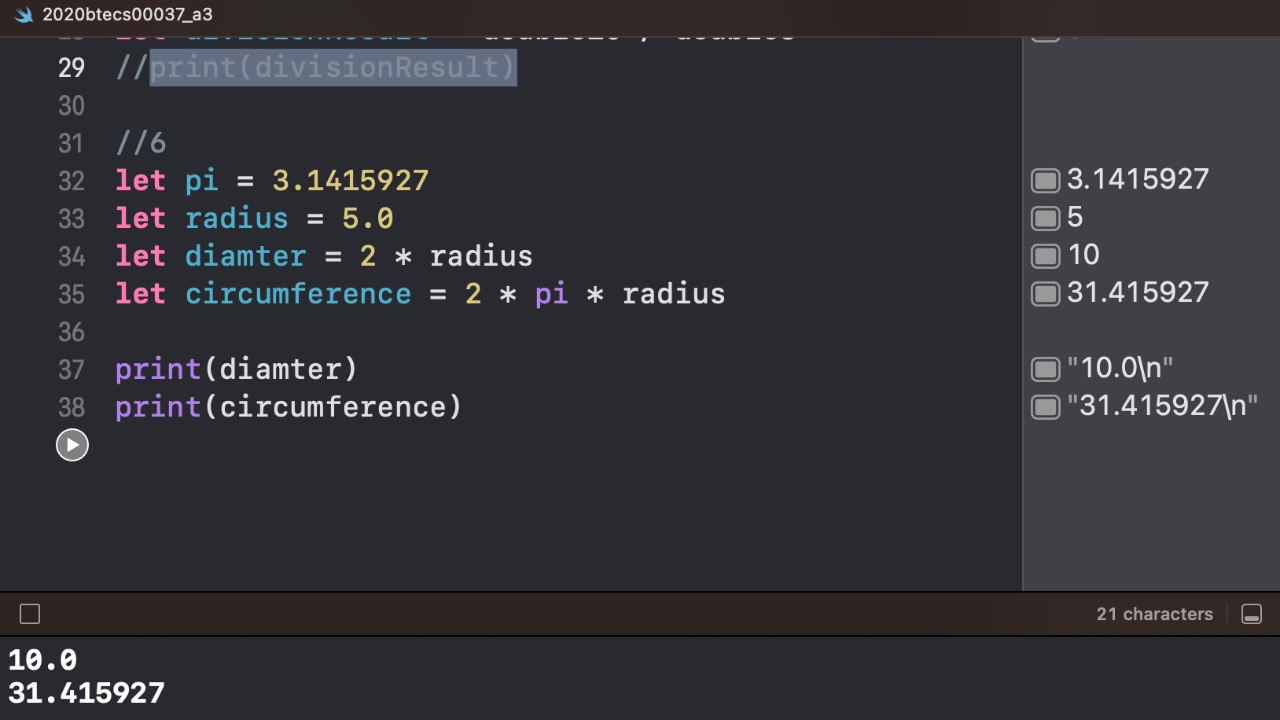


1. Given the value pi (3.1415927), create a `radius` constant with a value of 5.0, then calculate the diameter and circumference of the circle using the following equations, and print the results:

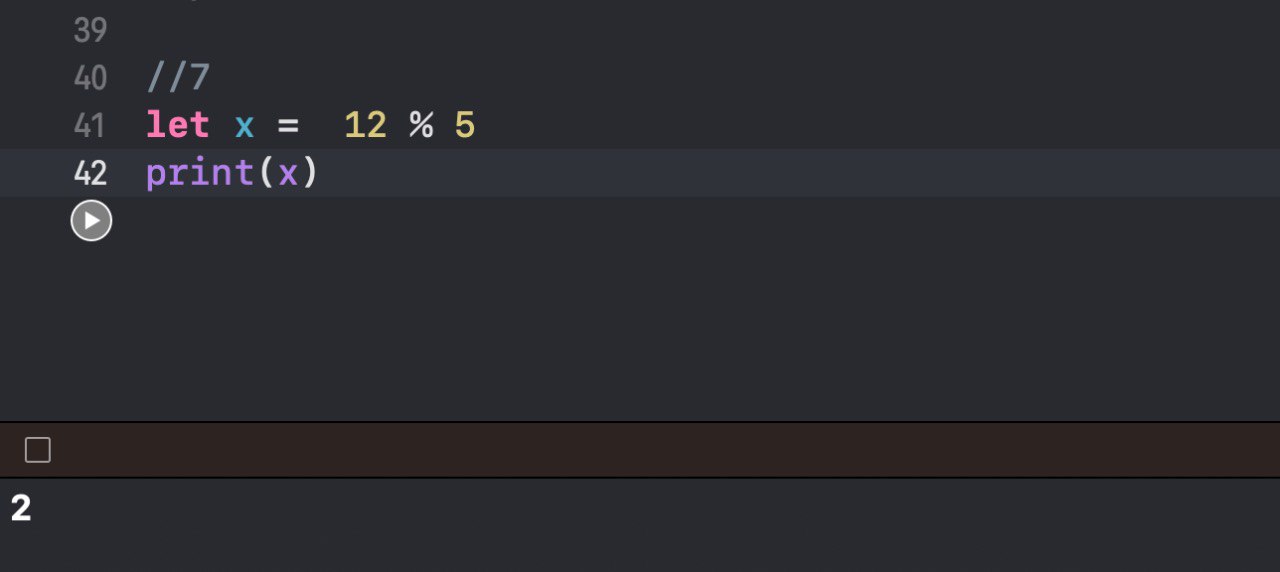
//diameter = 2 \* radius\*

//circumference = 2 \* pi \* radius.\*

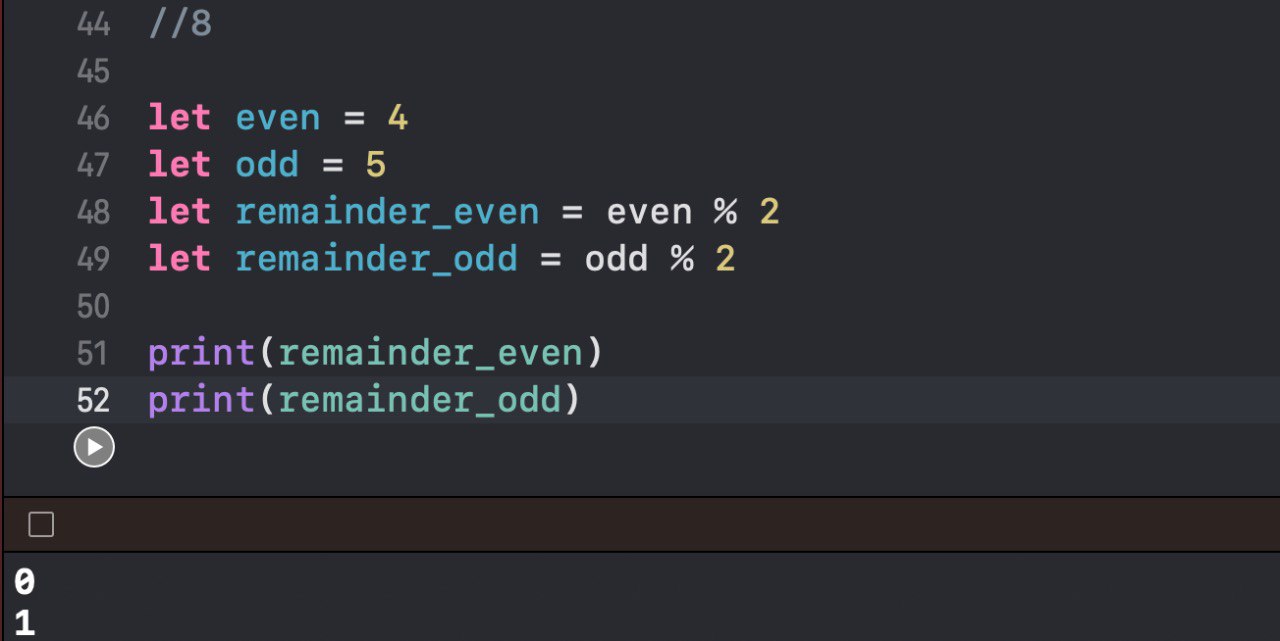
//let pi = 3.1415927



1. Create an integer constant. Using the modulus operator, set its value to the remainder of 12 divided by 5.

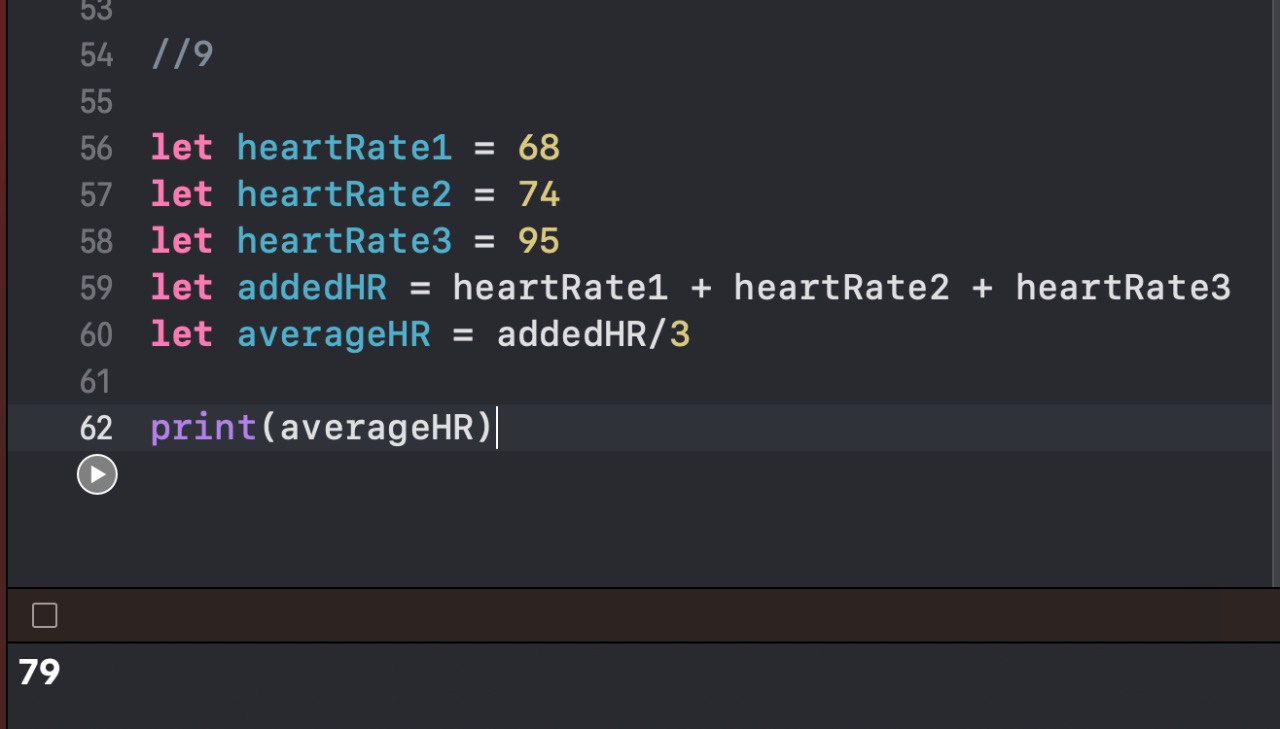


1. Create two integer constants, `even` and `odd` and set them to any even integer and any odd integer, respectively. For each, print the remainder of dividing the value by 2. Looking at the results, how do you think you could use the remainder operator to determine if an integer is even or odd?



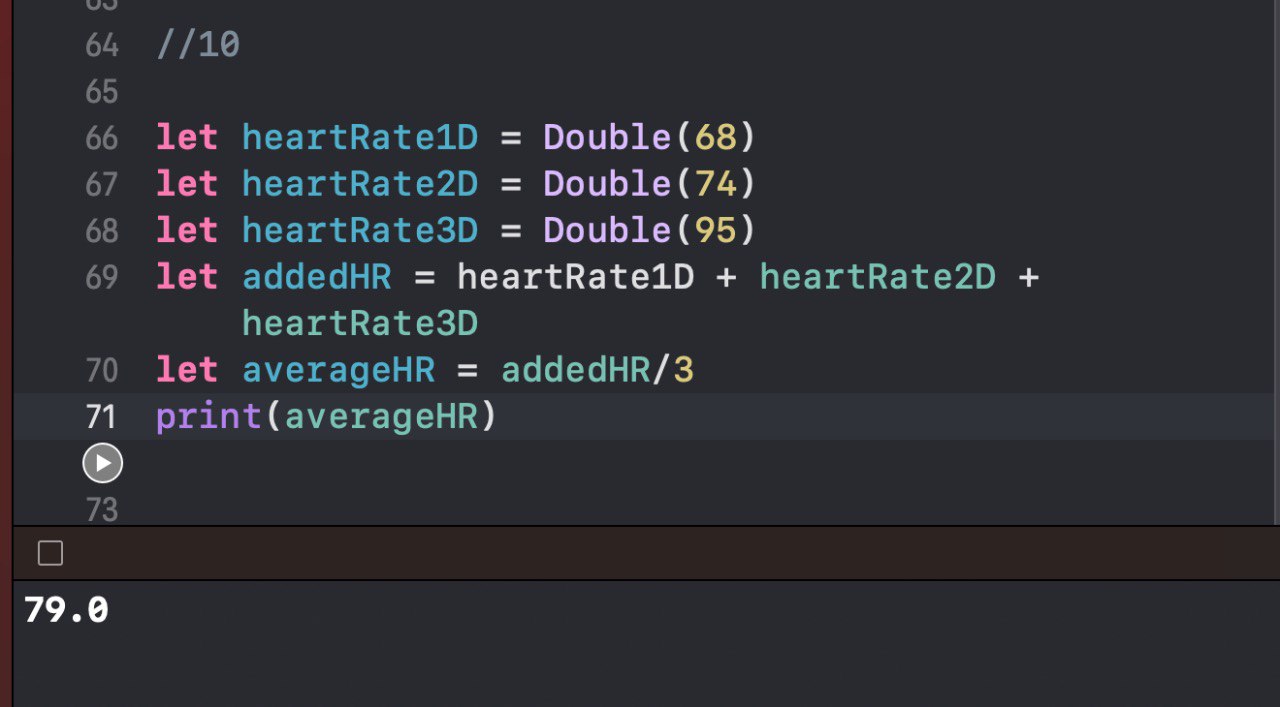
## App Exercise - Fitness Calculations

1. Your fitness tracker keeps track of users' heart rate, but you might also want to display their average heart rate over the last hour. Create three constants, `heartRate1`, `heartRate2`, and `heartRate3`. Give each constant a different value between 60 and 100. Create a constant `addedHR` equal to the sum of all three heart rates. Now create a constant called `averageHR` that equals `addedHR` divided by 3 to get the average. Print the result.



1. Now create three more constants, `heartRate1D`, `heartRate2D`, and `heartRate3D`, equal to the same values as `heartRate1`, `heartRate2`, and `heartRate3`. These new constants should be of type

`Double`. Create a constant `addedHRD` equal to the sum of all three heart rates. Create a constant called `averageHRD` that equals the `addedHRD` divided by 3 to get the average of your new heart rate constants. Print the result. Does this differ from your previous average? Why or why not?

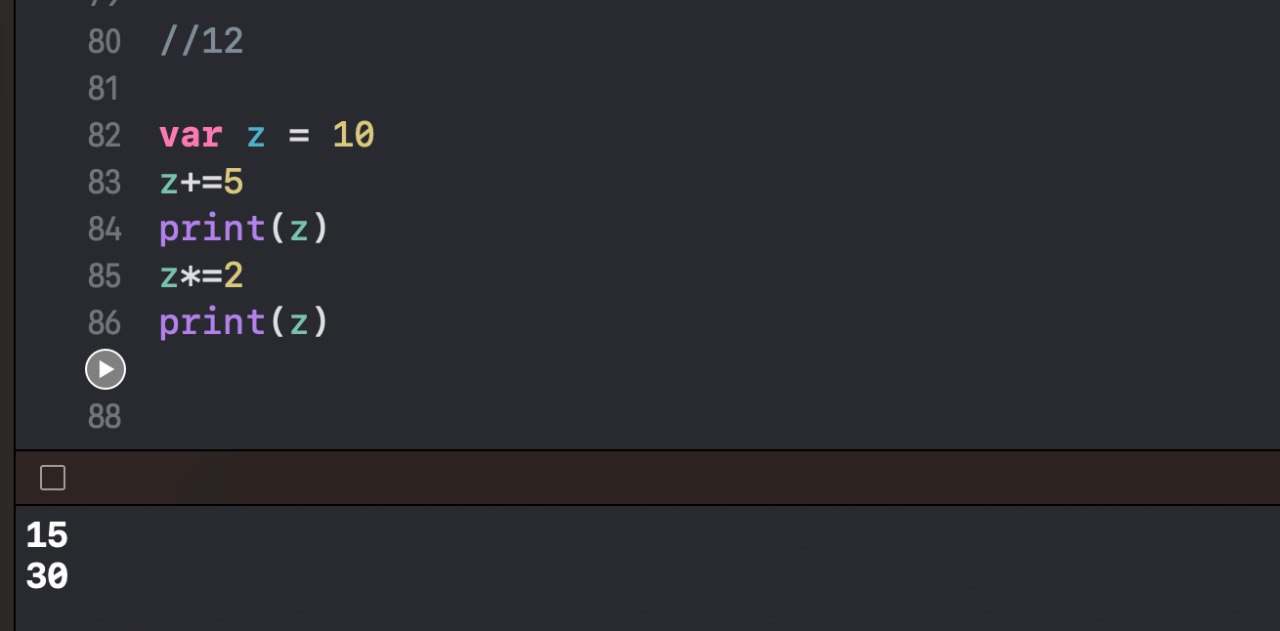


1. Imagine that partway through the day a user has taken 3,467 steps out of the 10,000 step goal. Create constants `steps` and `goal`. Both will need to be of type `Double` so that you can perform accurate calculations. `steps` should be assigned the value 3,467, and `goal` should be assigned 10,000. Create a constant `percentOfGoal` that equals an expression that evaluates to the percent of the goal that has been achieved so far.



## Exercise - Compound Assignment

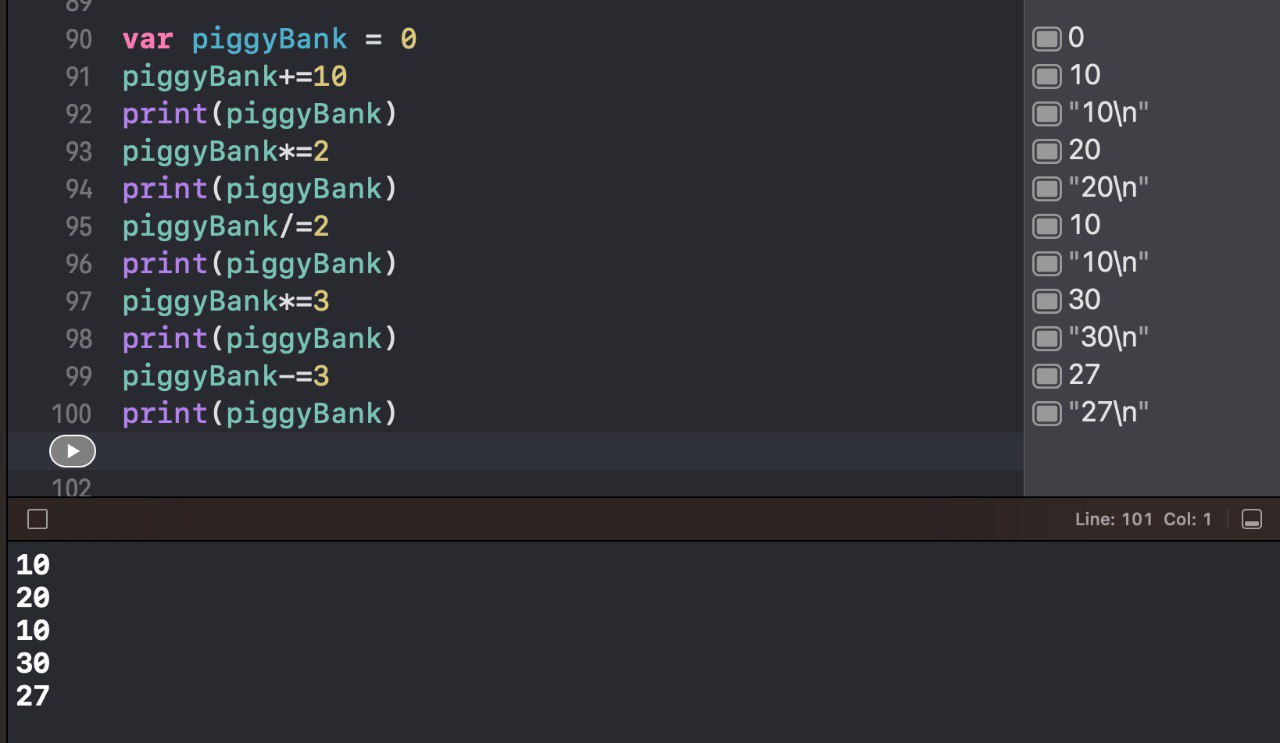
1. Declare a variable whose value begins at 10. Using addition, update the value to 15 using the compound assignment operator. Using multiplication, update the value to 30 using compound assignment. Print out the variable's value after each assignment.



1. Create a variable called `piggyBank` that begins at 0. You will use this to keep track of money you earn and spend. For each point below, use the right compound assignment operator to update the balance in your piggy bank.

* Your neighbor gives you 10 dollars for mowing her lawn
* You earn 20 more dollars throughout the week doing odd jobs
* You spend half your money on dinner and a movie
* You triple what's left in your piggy bank by washing windows
* You spend 3 dollars at a convenience store

Print the balance of your piggy bank after each step.



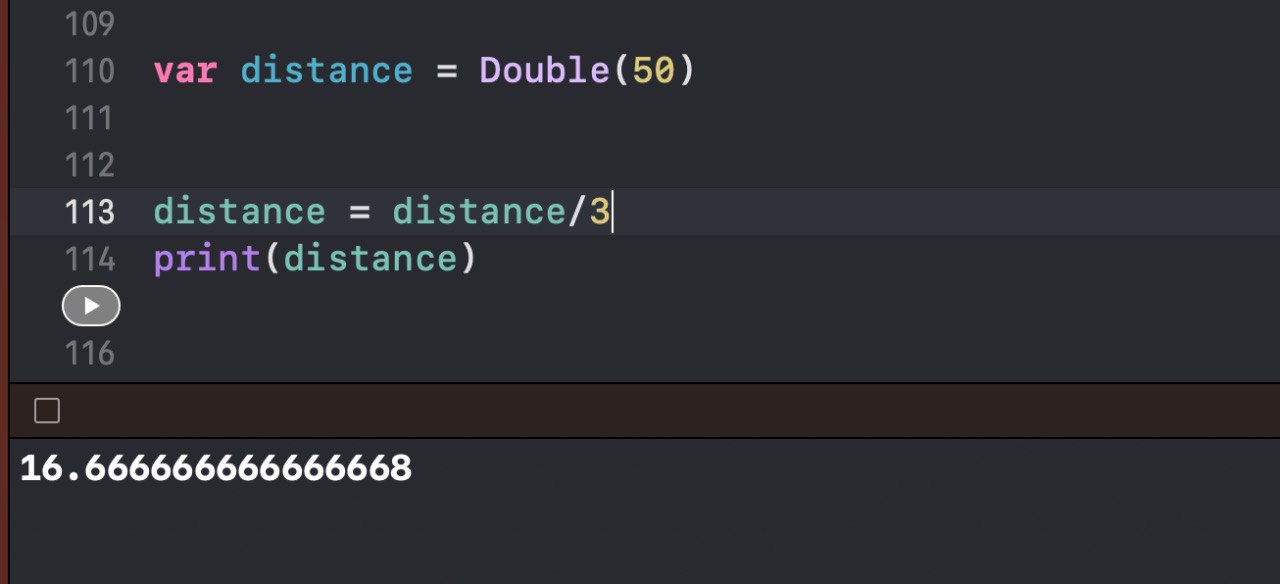
## App Exercise - Counting

1. The most basic feature of your fitness tracking app is counting steps. Create a variable `steps` and set it equal to 0. Then increment its value by 1 to simulate a user taking a step.



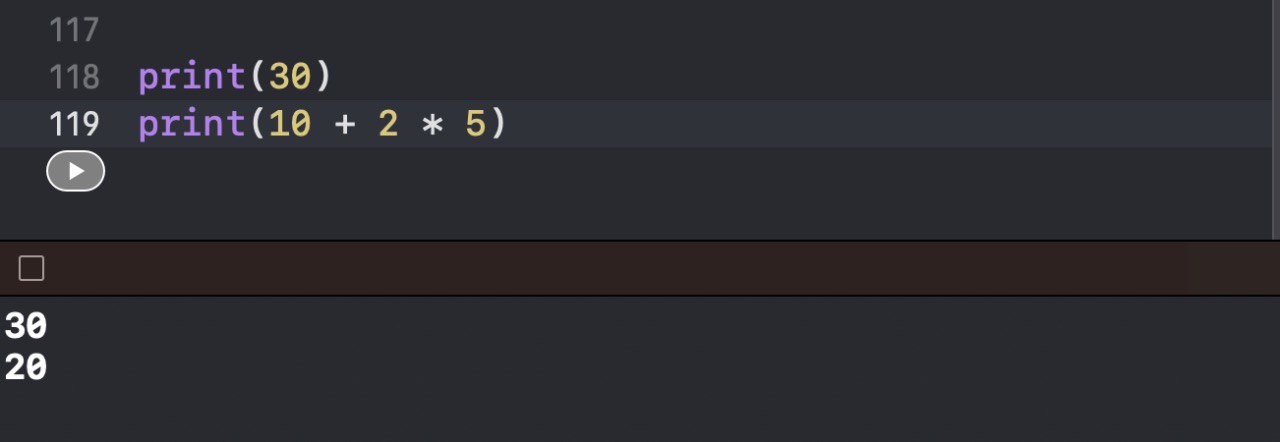
1. In addition to tracking steps, your fitness tracking app tracks distance traveled. Create a variable

`distance` of type `Double` and set it equal to 50. This will represent the user having traveled 50 feet. You decide, however, to display the distance in meters. 1 meter is approximately equal to 3 feet. Use a compound assignment operator to convert `distance` to meters. Print the result.

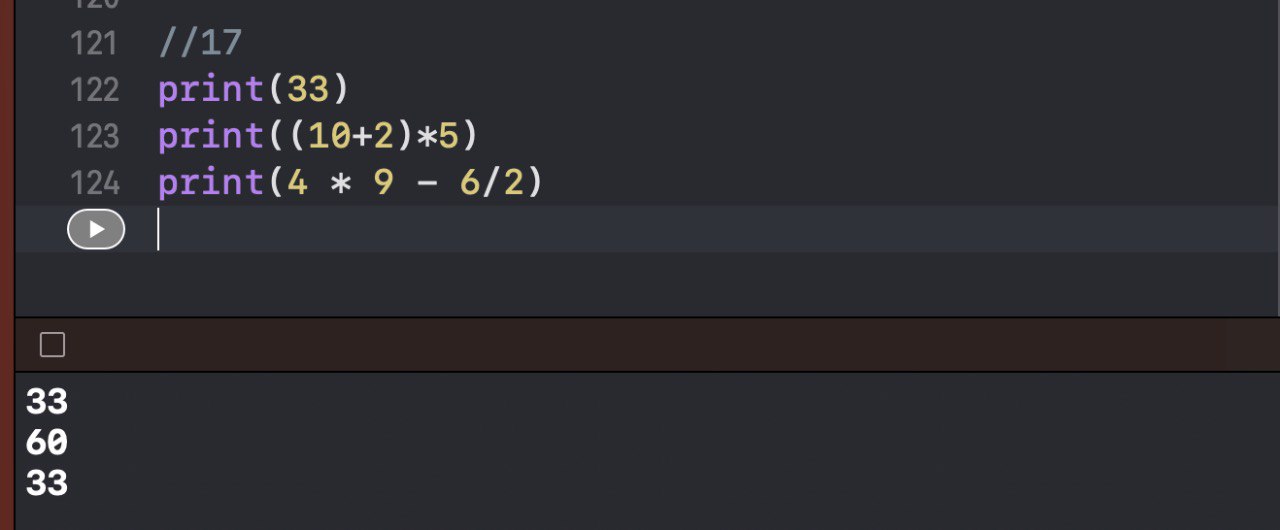


## Exercise - Order of Operations

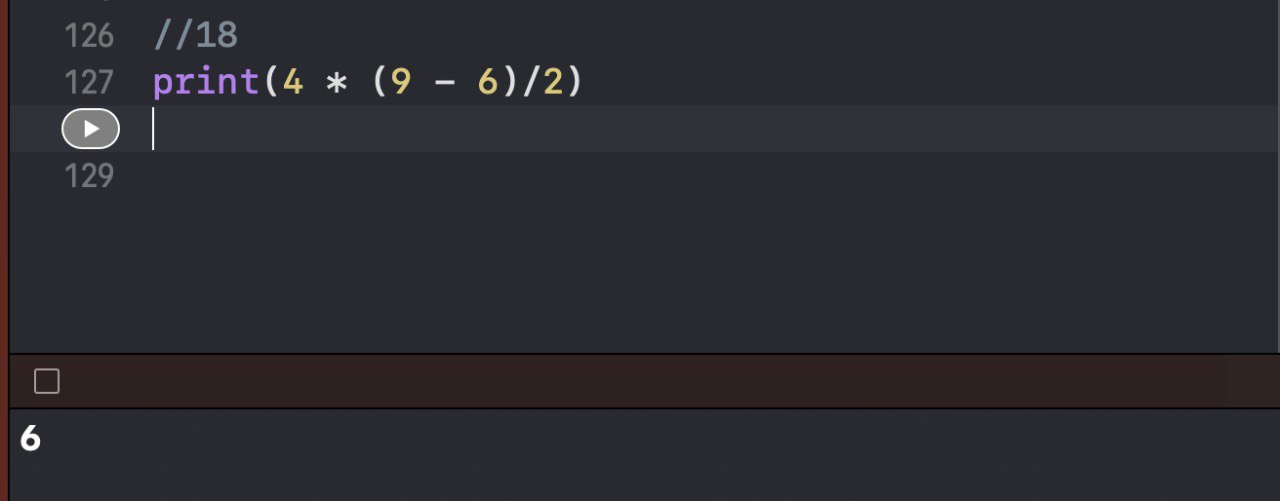
1. Print out what you think 10 + 2 \* 5 evaluates to. Then print out the actual expression (i.e. `print(10 + 2 \* 5)`)



1. In a separate `print` statement, add in the necessary parentheses so that addition takes place before multiplication. Print out what you think 4 \* 9 - 6 / 2 evaluates to. Then print out the actual expression.

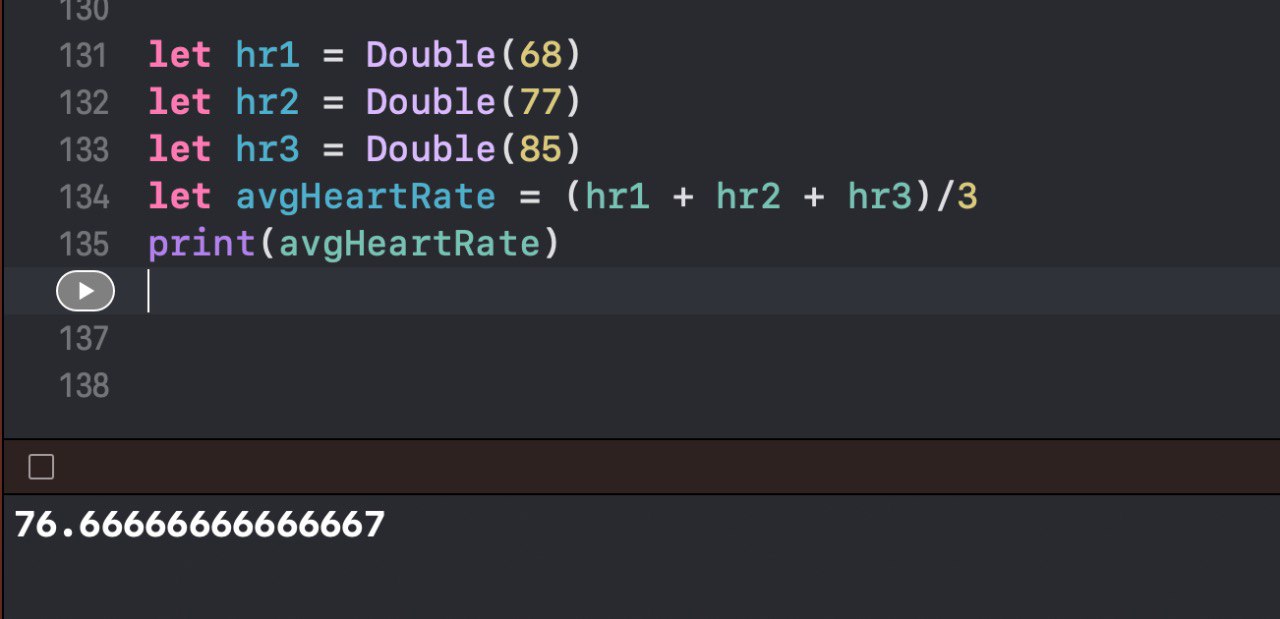


1. In a separate `print` statement, add in the necessary parentheses so that the subtraction is prioritized over the multiplication and division.

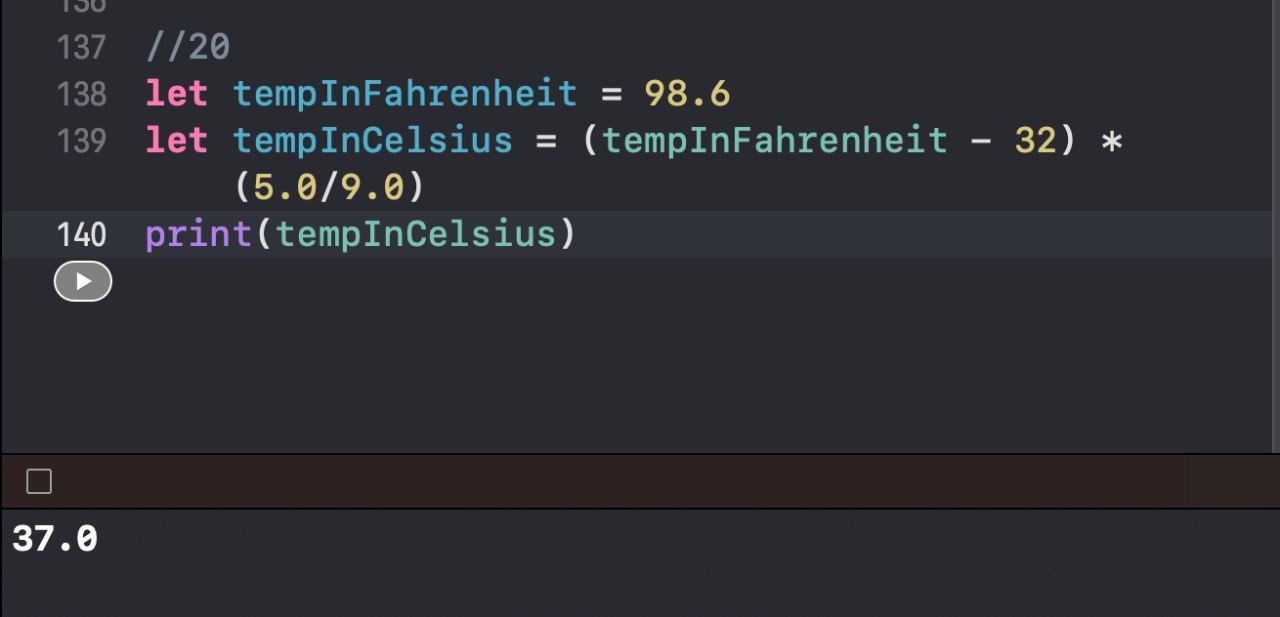


## App Exercise - Complex Fitness Calculations

1. If you completed the Fitness Calculations exercise, you calculated an average heart rate to display to the user. However, using proper order of operations you can do this in fewer steps. Create three separate heart rate constants, all of type `Double`, with values between 60 and 100. Then create a constant equal to the average heart rate. If you use correct order of operations you can do the heart calculation in one line.

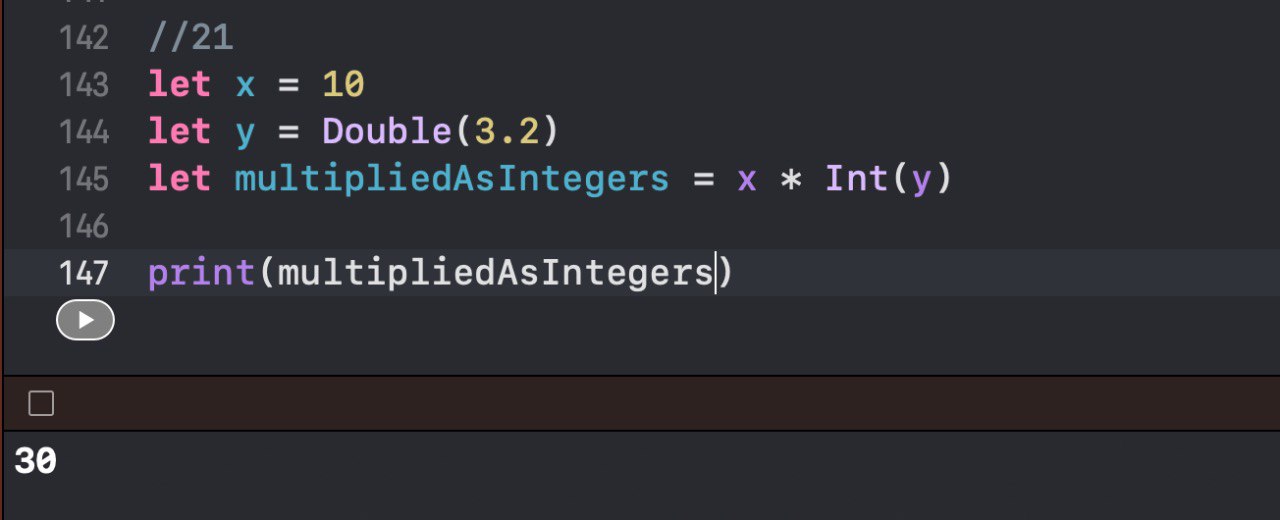


1. One feature you might want to give users is to display their current body temperature. Create a constant `tempInFahrenheit` equal to 98.6. You may want to also show the temperature in celsius. You can convert fahrenheit to celsius by taking `tempInFahrenheit` and subtracting 32, then multiplying the result by (5.0/9.0). Create a constant `tempInCelsius` that calculates in one line the temperature in celsius.



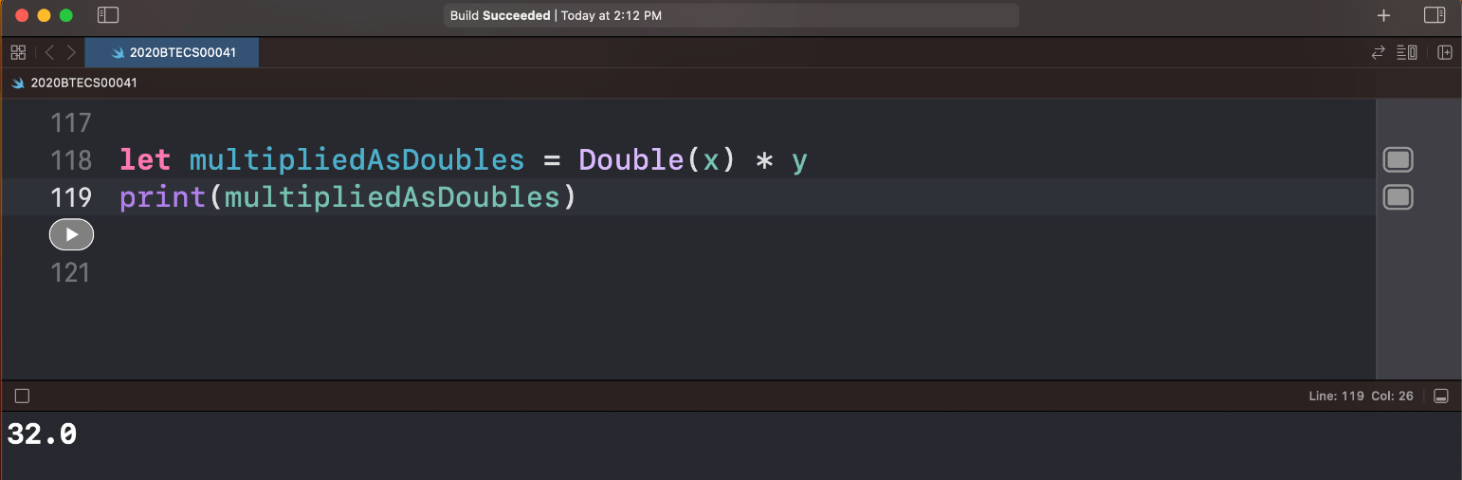
## Exercise - Numeric Type Conversion

1. Create an integer constant `x` with a value of 10, and a double constant `y` with a value of 3.2. Create a constant `multipliedAsIntegers` equal to `x` times `y`. Does this compile? If not, fix it by converting your `Double` to an `Int` in the mathematical expression. Print the result.

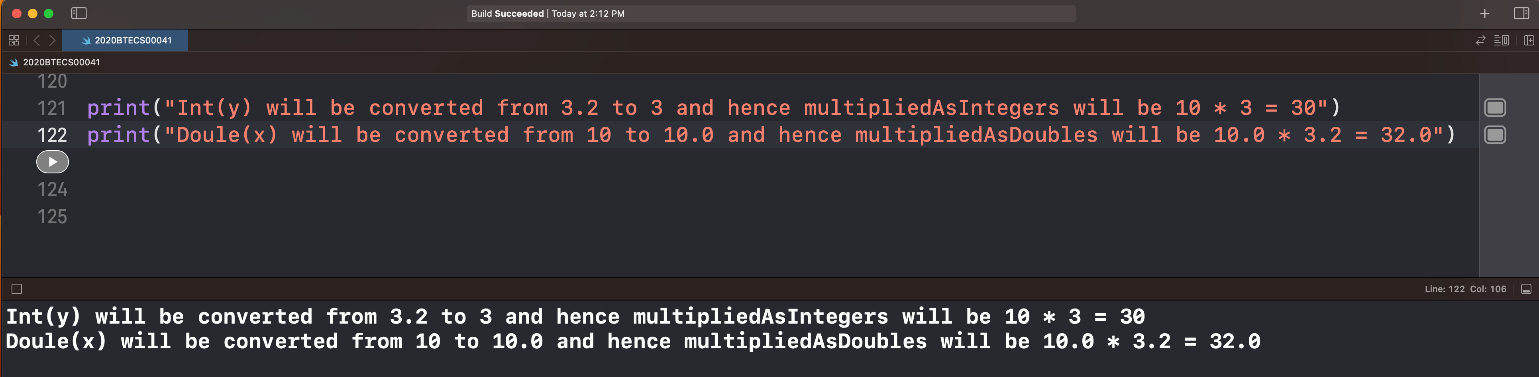


1. Create a constant `multipliedAsDoubles` equal to `x` times `y`, but this time convert the `Int` to a

`Double` in the expression. Print the result.



1. Are the values of `multipliedAsIntegers` and `multipliedAsDoubles` different? Print a statement to the console explaining why.



## App Exercise - Converting Types

1. If you completed the Fitness Calculations exercise, you calculated the percent of the daily step goal that a user has achieved. However, you did this by having `steps` be of type `Double`. But you can't really track a partial step, so `steps` should probably be of type `Int`. Go ahead and declare `steps` as type `Int` and give it a value between 500 and 6,000. Then declare `goal` as type `Int` and set it equal to 10,000.



1. Now create a constant `percentOfGoal` of type `Double` that equals the percent of the goal that has been reached so far. You'll need to convert your constants of type `Int` to be of type `Double` in your calculation.



\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*