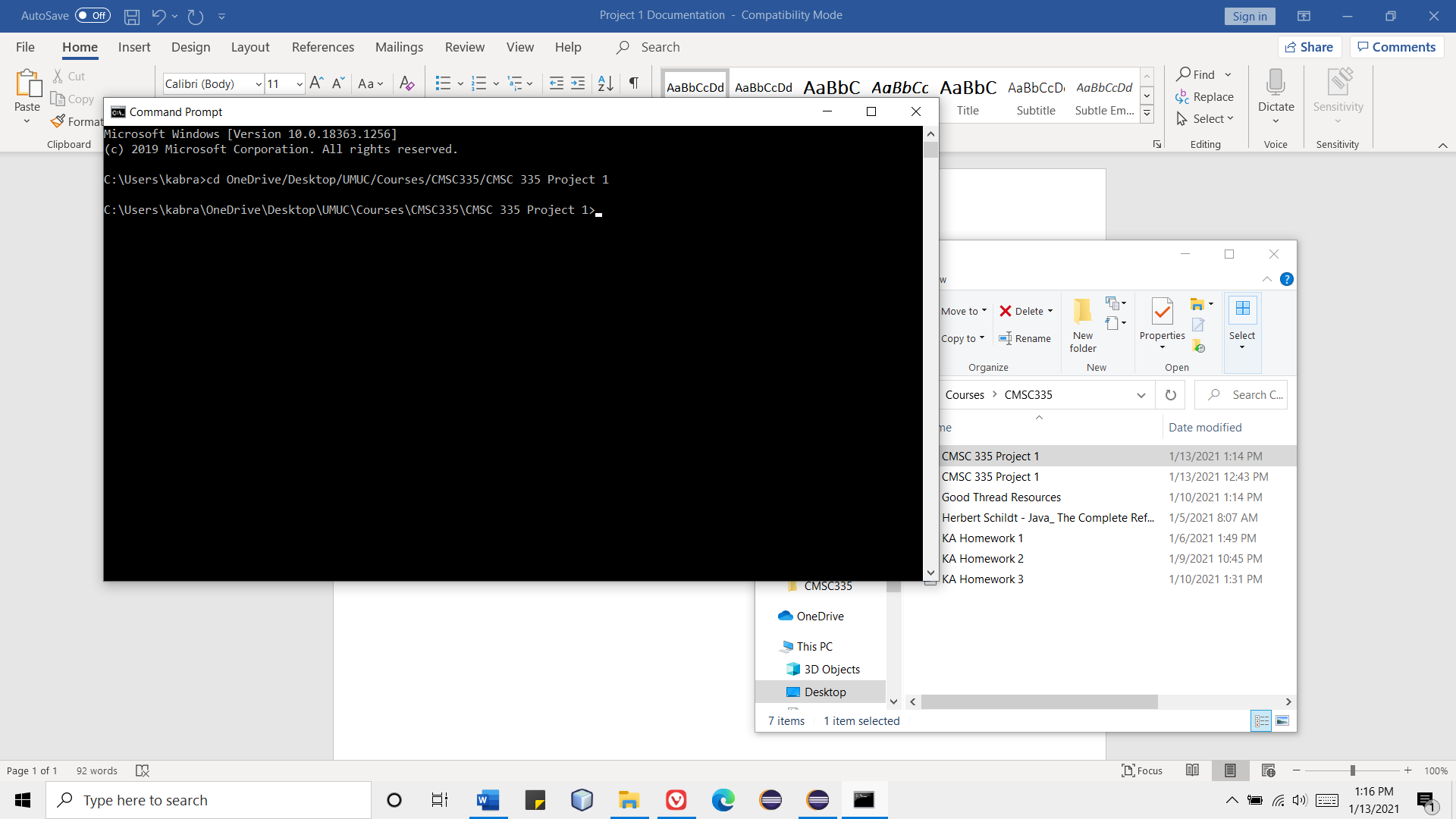
Kevin Abrahams

CMSC335 Project 1

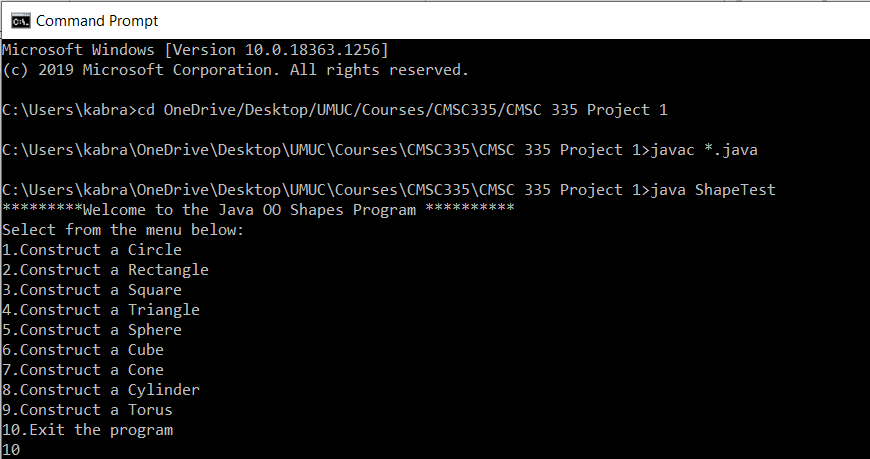
1/26/21

**User’s Guide (How Start the Program)**

To start this program, the program should be launched with the ShapeTest.java file. This program can be executed either through an IDE, such as eclipse in this case, or through the command line. These directions illustrate how to compile the program through the command line. To do so, navigate to the current directory of the program using the change directory command called “cd”. Ensure all java files are stored in the same folder, called CMSC 335 Project 1. Change to the directory of this folder.

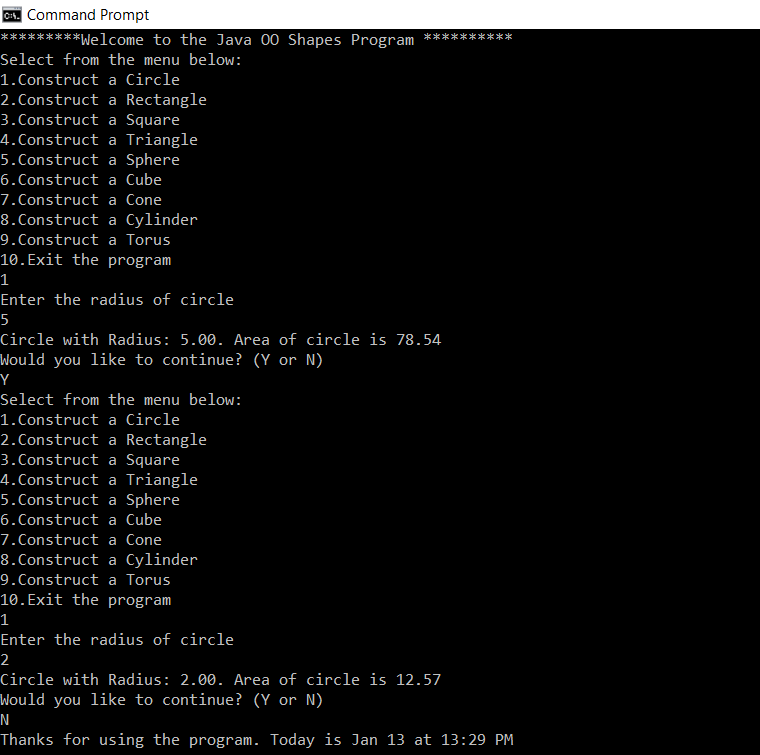


Then, after you have changed to the current directory of the CMSC 335 Project 1 folder, compile all java files in that directory using the command javac \*.java. Then, run the program containing the main method, ShapeTest.java, using the command java ShapeTest. The program then launches subsequently.

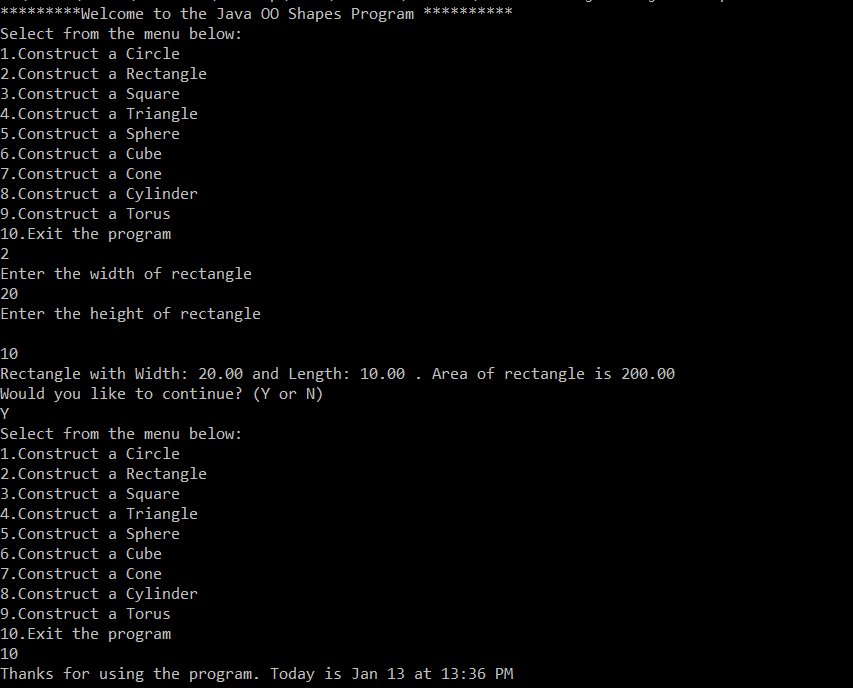


**Test Cases**

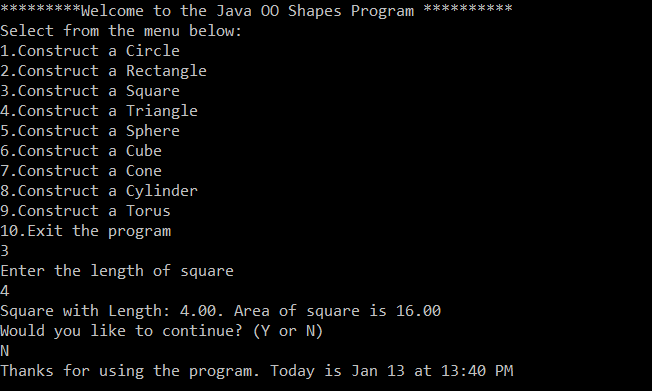
1. The first test involves selecting the first option which appears in the program. When the program launches, the user types and enters 1. The user then enters the information about the circle, typing in a radius of 5. Once the user enters the radius, they are shown more information about the shape, the circle’s area, and they are prompted if they want to continue. Since the user entered a y to designate yes, they are returned to the original starting menu. The user then moves on to enter a second circle with a radius of 2 and then terminates the program by selecting N.



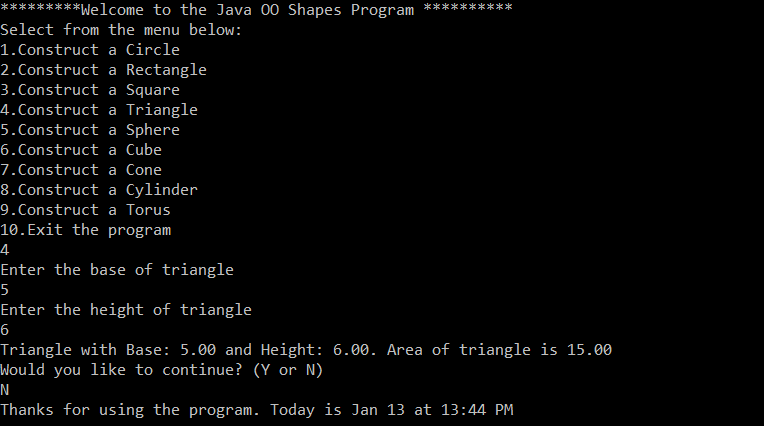
1. The second test involves constructing a rectangle. The user first selects option 2 by typing and entering the digit 2 as input. The user then enters information about the rectangle, first its width as 20 and then its height as 10. The user is then shown information about the rectangle, its area, and is asked if they want to continue. The user says yes by entering y and then shuts down the program by entering 10.



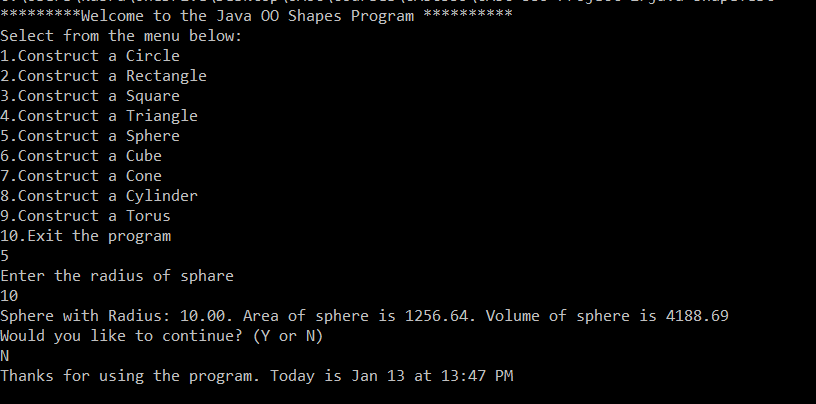
1. The third test involves constructing a square. The user makes this selection in choosing a square by typing and entering a 3 in the program as input. Upon doing so, the user is prompted for information about the square which they want to create, its length. The user does so by entering a 4, and then the area of the square is displayed. The user enters an N to signify that they do not wish to continue, and the program closes.



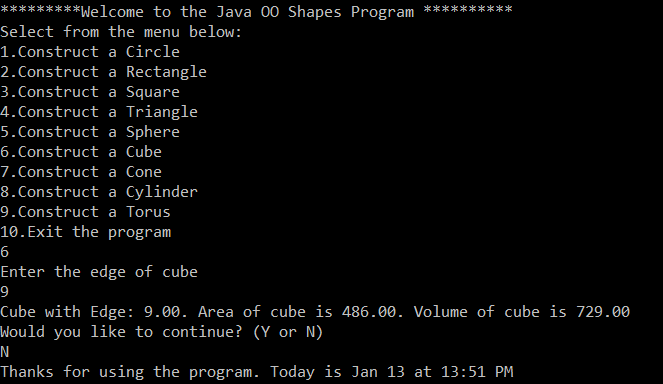
1. The fourth test involves constructing a triangle. To start, the user enters a 4 once the program is run and the menu is displayed. The user enters 5 once prompted to enter in the base of the triangle and a 6 once prompted for the height. The area of the triangle is then displayed as 15 which is the correct, expected result. The user enters N to discontinue the program.



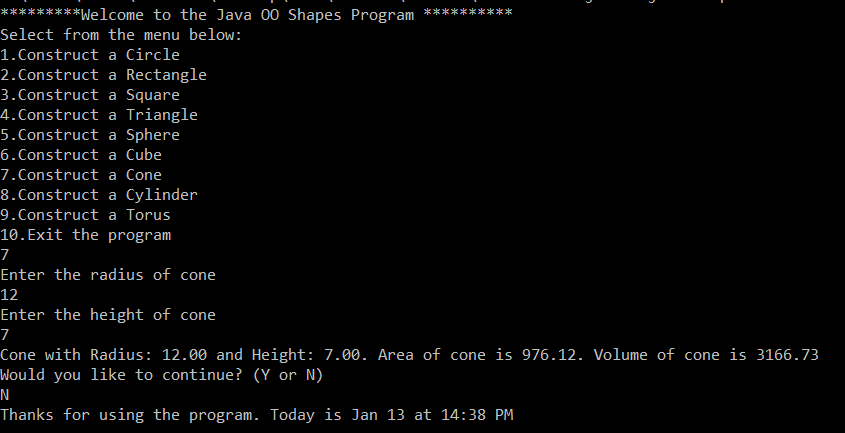
1. The fifth test involves constructing a sphere, the first three-dimensional shape to test in the program. The same process is involved. The user begins by entering a 5 at the menu. The user is then prompted to enter in information about the sphere. The user enters a 10 for the radius and then the area and volume for the sphere are shown. The user enters N to discontinue the program.



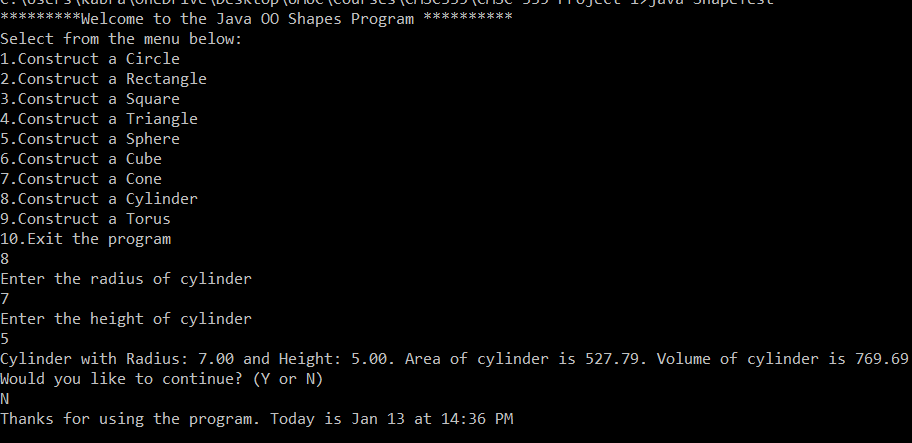
1. The sixth test involves constructing a cube, the second three-dimensional shape to test in the program. The user chooses to make a cube by entering a 6 at the starting menu. The user is then prompted to enter information regarding the cube, the length of its edges. The user enters a 9 and the corresponding data of the cube is shown, its area as 486.00 and its volume as 729.00. The user discontinues the program by entering N.



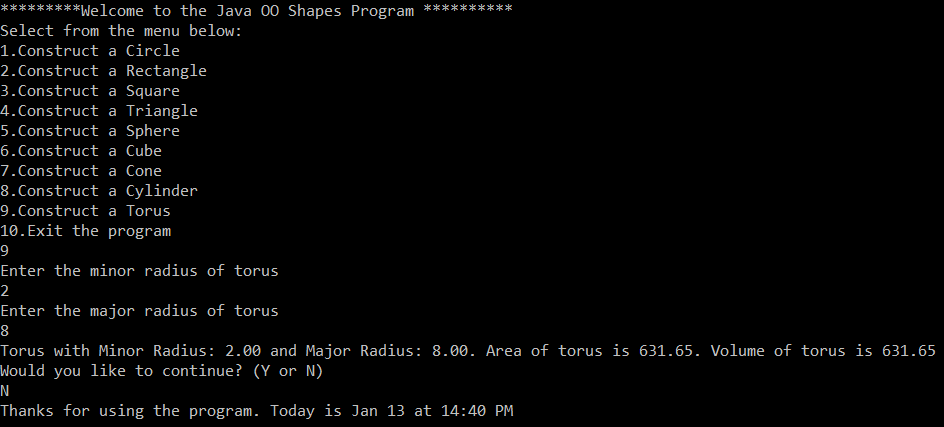
1. The seventh test involves constructing a cone. The user begins by entering a 7 to signify their choice. The user then enters information about the cone, a 12 for its radius and a 7 for its height. Corresponding information about the cone is then shown, including 976.12 for its area and 3166.73 for its volume. These results are mathematically sound. The user terminates the program by issuing an N as input.



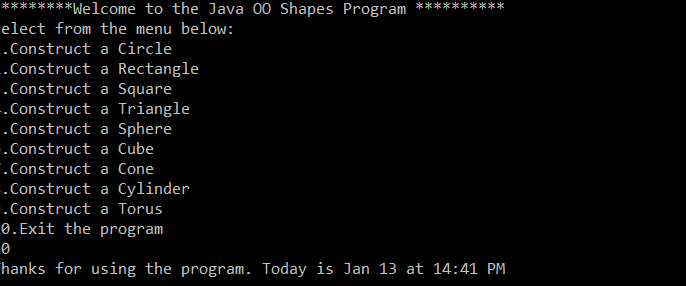
1. The eighth run of the program involves constructing a cylinder. The user chooses 8 when first prompted after the initial starting menu is shown. The user then begins to enter information about the cylinder to create. The user enters a 7 for the radius of the cylinder and a 5 for its height. The area and volume of the cylinder is then shown accurately as 527.79 and 769.69 respectively. The program then successfully ends after the user enters N.



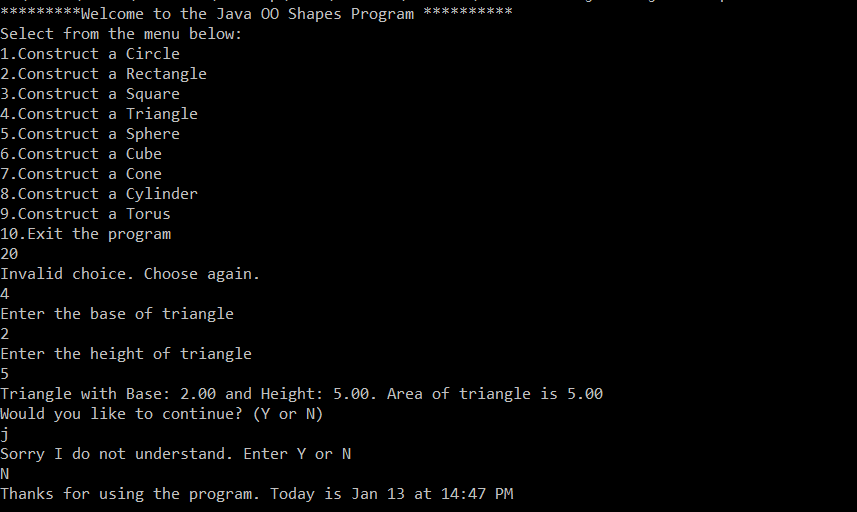
1. The ninth test involves constructing a torus. The user starts the program by entering a 9 to make the torus. The user then enters information about the torus, including a 2 for the minor radius and an 8 for the major radius. Information about the torus is then shown, including its area and volume as approximately 631.65.



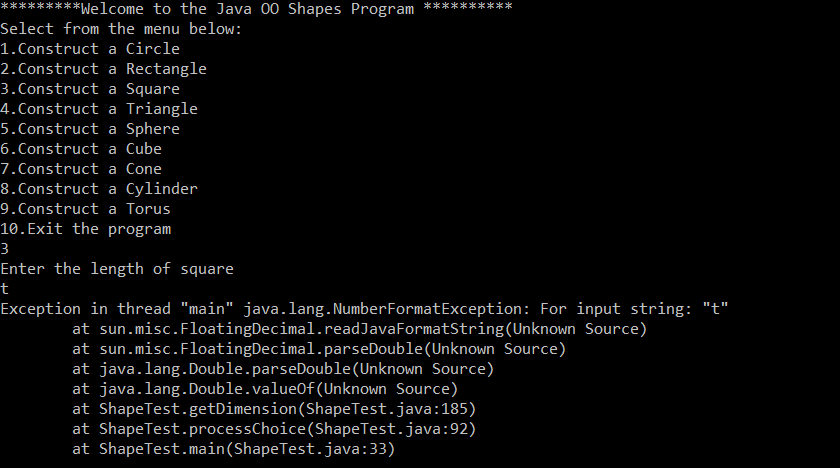
1. The tenth test involves first choosing 10 when the menu is first shown. The program closes immediately, as is expected.



1. The eleventh test involves choosing an invalid menu option. The user enters in 20 when the menu is first shown, and an error message indicating that improper selection has occurred by issuing the statement “Invalid choice. Choose again.” The menu then continues to be redisplayed until the user finally makes a proper, valid selection. The user then enters a 4 to break out of this loop and enters valid information about a triangle. When prompted to whether or not to return to the menu or terminate the program, the user enters an improper selection, “j”, causing an error message to appear onscreen. The user finally makes a correct choice by entering N, causing the program to complete execution.



1. This test case shows what happens when the user enters invalid input while entering information regarding a shape, such as a square. The user chooses to make a square by entering 3. When asked to enter the length of the square, the user types in a non-digit datatype, causing an exception to be thrown by the compiler at run-time.



**Lessons Learned**

This program was interesting because it built off from my previous programming knowledge with java such as exploring the ideas of loops and classes. This programming exercise was useful in gaining better knowledge and practical experience working with classes and knowing how to create them. I’ve built classes before in java, but this programming project gave me better experience with creating an intricate class hierarchy inundated with numerous class relationships such as inheritance and composition. I have a better sense of abstract classes, such as Shape, TwoDimensionalShape, and ThreeDiomensionalShape, due to having to use them in some of my parent classes, whose generality prohibits their instantiation and thus renders their full and complete implementation reliant and dependent upon their descenders and inheritors. In other words, these classes rely on the classes which inherit from them to fully complete their implementation. Thus, I have learned that abstract classes are incomplete and thus are useful in ensuring that a class not be instantiated. It makes sense for why these classes and some of the methods created in this program are declared as abstract because they are not concrete or meant to be used directly used or created in a separate java program. It was interesting overriding methods, including those declared and defined in the preceding, overarching classes, such as the computational methods of each of the shapes’ areas and volumes. It made sense for why these methods were declared as abstract and for why they were put in the parent classes because each specific class of shape had this behavioral state information in common yet had a slight modifiable difference in their implementations of it. For instance, the rectangle has a method to calculate area just like a triangle, and a cylinder has a method to calculate volume just like a sphere, but the manner such is done in varies. Thus, the methods are declared as abstract in the parent classes because the child classes must complete their own implementation of them. I have a better working knowledge of the constituent pieces of a class and the structure which a class consists and is made of, such as instance variables and methods. I have better understanding of how these components connect and intertwine, especially in the form of inheritance relationships.

**UML Diagram**

