Sean Blanton

Introduction to Java

Project 03

Sep 16 2015

**Jar file is executable (5pts)**

-Evidenced in screenshot

**Programming Guidelines are followed (5pts)**

-Required comments present in class comment block

-Indentations consistent

-Block bracing consistent

**Report with screenshots is created (25pts)**

-This document

**Planet class is created and meets specs (30pts)**

-CelestialBody.java

-Planet.java

-Name field access via CelestialBody Parent Class

-Planet::moonsCount private field

-Two parameterized Planet constructors covering provision of full CelestialBody data or not

-Setters and Getters for all fields

-toString method overridden

**Solar System class is created and meets specs (30pts)**

-SolarSystem.java

-planetsArray field

-Additional private fields for stars and name of Solar System

-Single parameterized constructor

-AddPlanet and GetPlanet methods for Inserting and retrieving Planets

-Sol uses DefinePlanet method, Rigil Kent uses AddPlanet method

-GetPlanetCount(), GetMoonCount() methods

-toString method overridden

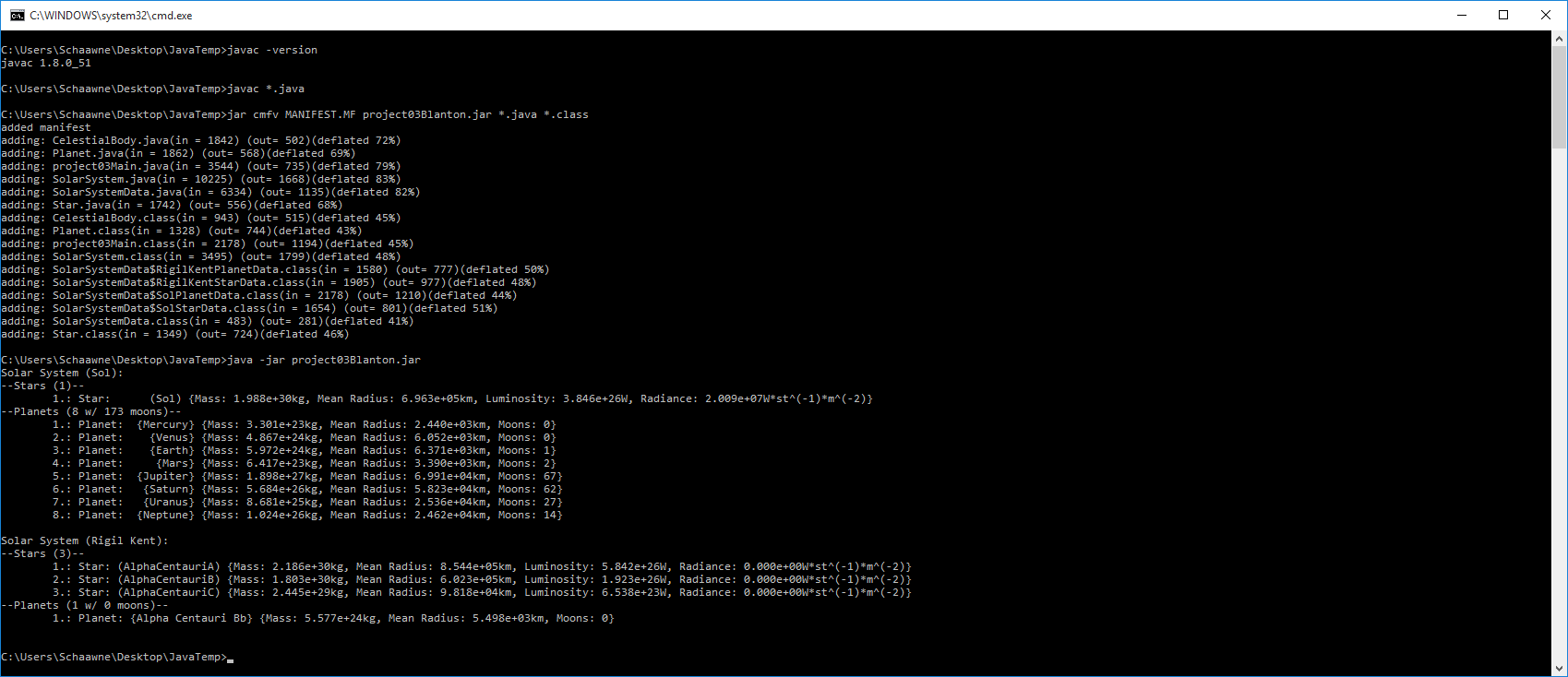
**Driver class is created and meets specs (30pts)**

-Contains main method

-main instantiates both Sol and Rigil Kent SolarSystem objects

-Populates Sol and RK from data encoded in Data class/enums

-Uses SolarSystem toString override to print to console



# 

# Summary

Created a basic Solar System set of classes in java. Used intelliJ IDEA community IDE. Created additional classes for CelestialBody, Star, and SolarSystemData to practice inheritance and enumeration. Manually compiled and packed source for release to provide concise screenshots of functionality. Setup exception handling for bad parameter inputs that would drive array creation. Main method does not trigger. Discovered a typo with Star toString override where radiancce was being substituted for radius and resolved. Attempting to use javadoc comments as practice but no requirement to submit generated documentation.