

Project 1: Orioles Baseball

Due: Sunday 09/23, 11:00 PM
Open/Close Policy: Open



[Click here to see the full roster for the Orioles!](#)

1 Objective

This project will allow you to practice variables, strings, input/output facilities, conditional statements, logical operators, and the Eclipse IDE.





2 Overview

Professional baseball is interesting and exciting; this project... is not. But we have to start somewhere!

You will write a very simple application that asks the user to identify which Orioles players wear certain jersey numbers. (Or it might ask what jersey numbers do certain players wear.)

IMPORTANT: The program is limited to the four players in the table below. Your program will only be interested in the players' last names and corresponding jersey numbers.

This project is considered “open”. Please visit the course web page for information regarding the open/closed policy for projects of this course.

				
Name:	Adam Jones	Trey Mancini	Caleb Joseph	Chris Davis
Jersey Number:	10	16	36	19
Position:	Center Fielder	Designated Hitter	Catcher	First Baseman

3 Specifications

Your program knows four things: Jones is number 10, Mancini is number 16, Joseph is number 36, and Davis is number 19. It does not know anything about any other players on the team, so please only bother to make it work for these 4 players.

The program begins by prompting the user with:

Type 1 to enter a number or 2 to enter a name:

The user then enters either “1” or “2”.

3.1 Processing an Entry of 1

The program prompts the user with:

Enter player number:

The user must enter either 10, 16, 36, or 19. If any other number is entered, the program will print out “Invalid choice.” and will terminate with no further output.

Assuming that the user has entered one of the four valid numbers, the program prompts the user with:

Which player wears number n on his jersey?

[In the prompt above, the “n” must actually be either 10, 16, 36, or 19, depending on what the user had selected.]

The user will enter a String. If the String entered represents the correct player name for the number selected, the program will output “Correct!” and terminate. But if the user

has entered the wrong String, then the program will output “Incorrect!” and terminate. [Note that the player names are last name only!]

3.2 Processing an Entry of 2

The program prompts the user with:

Choose a name:

The user must enter either “Jones”, “Mancini”, “Joseph” or “Davis”. If any other String is entered, the program will print out “Invalid choice.” and will terminate with no further output.

Assuming that the user has entered one of the four valid names, the program prompts the user with:

What number does name wear?

[In the prompt above, the “name” must actually be either “Jones”, “Mancini”, “Joseph”, or “Davis”, depending on what the user had selected.]

The user will enter a number. If the value entered represents the correct number for the player selected, the program will output “Correct!” and terminate. But if the user has entered the wrong value, then the program will output “Incorrect!” and terminate.

4 Getting Started

In order to help you get started, we have defined an `OriolesBaseball` class with a `main()` method in a file named **`OriolesBaseball.java`**. This file can be retrieved by checking out the project from the CVS repository. Remember that you must have set up your repository in order to check out and submit projects. Information on how to do this can be found in the [Managing Projects](#) section of the Eclipse tutorial. Follow the instructions given there to check out project **131Fall18Proj1**. After checking out the project, when you switch back to the Java perspective, you will see the above files in the “Package Explorer” window, and you will be able to start modifying them.

If you write the project from scratch, without checking out the “131Fall18Proj1” files from your CVS repository, you will not be able to submit your work.

5 Requirements

- **You may not use any “return” statements in this project!** (If you don’t know what a return statement is, ignore this instruction.)
- Your program must use the single class `OriolesBaseball`.
- You must use “named constants” for any literal values that will not change during program execution.

- You must use meaningful variable names and good indentation.
- Input and output operations must be implemented as demonstrated in class. (i.e. use the Scanner class for input, and use System.out.print or System.out.println for output.)
- You do not need to worry about the user entering the wrong types of values. If an integer is expected, assume the user enters an integer; if a String is expected, assume the user enters a String.
- You may assume that the user enters either “1” or “2” (and nothing else) after being asked whether they want to choose a player name or a jersey number.
- You should not get creative and modify the project specifications. For example, do not add extra behavior or output.
- We use an automated system to grade projects and you will not pass our tests if you do not follow the project specifications precisely. **In particular, you must spell things exactly as we have spelled them and use punctuation exactly as we have.** One missing or incorrect punctuation mark (a period, exclamation point, comma, or colon) could cause you to fail all of the tests. If you spell something wrong (“nubmer” or “incorect” for example) you will fail many of our tests. **PROOFREAD CAREFULLY.** You do NOT need to worry about how many spaces to put between words in your output – it doesn’t matter. Also, whether or not you capitalize letters doesn’t matter in your output.
- Just in case you know what an array is – you may not use arrays (or any Java collections of any kind) while implementing this project.
- Your program must terminate without using anything like System.exit(). (Don’t worry if you don’t know what that is.)
- **Be sure to only use ONE Scanner for your project. If you use more than one Scanner to read from the input stream, you will fail our tests!**
- Please be sure that the Scanner is closed when your project terminates.

6 Sample Runs

The following examples show how your program should behave. Note that items in *italics* represent the things that are entered by the user. Keep in mind these are just examples and not the only scenarios that your program is expected to handle.

Example 1:

Type 1 to enter a number or 2 to enter a name: 1

Enter player number: 16
Which player wears number 16 on his jersey? Mancini
Correct!

Example 2:
Type 1 to enter a number or 2 to enter a name: 2
Choose a name: Teli
Invalid choice.

Example 3:
Type 1 to enter a number or 2 to enter a name: 2
Choose a name: Jones
What number does Jones wear? 99
Incorrect!

7 Submission

Submit your project from Eclipse by right-clicking the project folder and selecting “Submit Project”. You may submit as many times as you want – we only grade your most recent submission. After you have submitted your project, you should visit the [submit server](#). There you can obtain limited feedback about how well your project is performing. (Choose “release test” to see detailed information.) The number of times you can run our tests on your project (before the due date) is limited. **The earlier you begin working on the project, the more opportunities you will have to see how your project performs on our tests before the due date!**

8 Grading

There are seven release tests which will be run on your project. Together, these tests will determine 90% of the grade on the project. The remaining 10% will be based on your use of correct programming style.