**Step 1  
  
Project Overview**

This project is a chance for you to combine and practice everything you learned in this section of the Nanodegree program. You will be making a class to represent a report card.

The goal is to design and create the structure of a ReportCard Java Class which would allow a school to store a student’s grades for a particular year. *This project will be just a java class, rather than a full Android app*.

**Why this project?**

In the most recent portion of the Nanodegree program, you learned about custom classes, which are critical to every object-oriented program you will build going forward. It is vital to be able to think about how to design objects which interact with each other and model real-world concepts.

**What will I learn?**

This project is about combining various ideas and skills we’ve been practicing throughout the course. They include:

* Designing a custom class
* Creating that class in Java code.
* Storing information in an array
* Looping through an array

**Subscription vs Free-Version student**

If you are subscribed to the paid Nanodegree program, coaches will be available in the Coaches Lounge to help guide you through the final project. You will receive feedback after submitting your project and a verified certificate after successfully finishing this project.

If you are enrolled in the free version of this course, the project rubric and all of the project details are available to you.

Either way, we would love to see a description and link to your app on the discussion forum.

**Step 2:**

**Build Your Project**

For this project, you'll be creating a single Java class file. The class file is intended to represent a report card.

You'll want to think about what information report cards have and what methods a report card might need. These will be different depending on the educational system you would like to model. For instance, some systems use a letter grading system where the grades are "A,B,C,D,F" and some use a numeric system where a grade would be represented as a percentage.

There are some things that a report card will always do and some things that a report card will never do, however. Report cards never teach classes, so a teach() method would go instead in a Teacher class. Report cards always show a student's grades, however, so your class should include some way to do that. It may be useful for you to read over [this page on the toString() method](https://classroom.udacity.com/nanodegrees/nd803/parts/8031345406/modules/674539489875460/lessons/7863766808/concepts/81887887670923)before starting your project.

Your project will be evaluated using the [Report Card project rubric](https://review.udacity.com/#!/rubrics/160/view).

**Step 3:**

**Prepare for Submission**

**Clean Your Build**

Before submitting, please follow the instructions for cleaning your project files. This removes some temporary files and greatly decreases the size of your project.

[Cleaning your project files](https://d17h27t6h515a5.cloudfront.net/topher/2016/June/5769c116_1000-files-tutorial/1000-files-tutorial.pdf)

**Review the Project Rubric**

Udacity reviewers will be reviewing your project based on the project rubric. All specifications must be met to pass the project.

[Project review Rubric](https://review.udacity.com/?_ga=1.159318650.1362034947.1463828731#!/rubrics/160/view)

**Final Submission Checklist**

Before submitting your project for evaluation, we recommend that you check that each of the following is true:

1. Your app compiles and runs as expected.
2. You are proud of your app and its output.
3. You completed this project according to instructions.
4. You cleaned the project using the instructions above.
5. You checked your project against the rubric.

**Zip your Project for Submission**

Be sure to zip your whole android project only after you have cleaned it.

## PROJECT SPECIFICATION

**ReportCard**

Class Design

| CRITERIA | MEETS SPECIFICATIONS |
| --- | --- |
| Constants | Any constants contained in the class are never modified, and all make sense as things which should be theoretically constant. |
| Variables | The class contains appropriate variables to track information which belongs on a report card. The class does not contain variables which should belong, for instance, to a student, a teacher, a school, or a textbook. |
| Constructor | The class contains at least one constructor which properly instantiates each variable in the class. |
| Getters | Any private variables which might be read off of a report card should have a getter method to access them. |
| Setters | Any private variables that should be set by an outside class (for instance, a teacher might set a student’s grade in their history class) should have a setter method. |
| The toString method | The class contains a toString method which returns the contents of the class in a human-readable string. |

Functionality

| CRITERIA | MEETS SPECIFICATIONS |
| --- | --- |
| Compilation Errors | The code compiles without errors |

Code Readability

| CRITERIA | MEETS SPECIFICATIONS |
| --- | --- |
| Format | The code is properly formatted i.e. there are no unnecessary blank lines; there are no unused variables or methods; there is no commented out code. The code also has proper indentation when defining variables and methods. |
| Naming conventions | All variables, methods, and resource IDs are descriptively named such that another developer reading the code can easily understand their function. |