# :

|  |  |  |
| --- | --- | --- |
| **Category** | **Mistake** | Penalty |
|  | **Part 0: Compatibility with the automated tests** |  |
| Your code doesn’t compile with the tests / starter code that the instructor provided to you.  It appears that you have a copy of the starter project from a different quarter. Your code works with that version, but does not work with the current version.  There’s two things I’d like to be clear about:   1. It’s great for you to get help from friends who know how to program, including friends who have taken this class in prior quarters. They can help you learn how to program and that is good. They can help you learn BUT they cannot do your work for you. 2. You need to do your work on your own. Borrowing a copy from a prior quarter is a form a plagiarism / academic dishonesty. As per the syllabus, the penalty for plagiarism is 20% for the first offense. | | -30 |
|  | **Part A: Using an Existing MyDate class** |  |
|  | You must start by prompting the user today’s date (month and day) | -16 |
|  | You must prompt the user for their birthday date (month and day) | -16 |
|  | You must print the number of day’s in the user’s birthday’s month | -16 |
|  | You must print the number of days between now and the user’s birthday date (in days) | -16 |
|  | If the user’s birthday is today, you must print a “Happy Birthday” message | -16 |
|  | EITHER print out “Happy Birthday” OR print out the number of days until the birthday, but not both | -6 |
|  | Use the .equals and .nextDay methods to determine if today is the user’s birthday, instead of duplicating that logic in main | -8 |
|  | Use the .equals method to determine if today is the user’s birthday, instead of duplicating that logic in main | -8 |
|  | By comparing the user input variables directly you’re duplicating the ‘equals’ logic here in main. Duplicating code is (generally) bad.  It would be better to find another way, such as checking if the count of days between today and the birthday is zero | -6 |
|  | Looks good. |  |
|  | **Part B: Creating your own MyDate class** |  |
|  | Remove the ‘today’ prefix from the instance variables. Putting that in front of all the variables isn’t really helpful and it’s also not accurate for the birthday object | -3 |
|  | public MyDate(int month, int day) |  |
|  | This class should not print anything to the user. We haven’t covered ‘exception handling’ yet, but that would be a great approach to use instead.  The big idea here is that if these methods print anything then before you call any of these methods you have to stop and ask yourself “Is it ok if the method prints something to the user?” first. In other words, these methods now have TWO important effects – whatever we actually want them to do, AND interacting with the user. | -5 |
|  | Looks good. |  |
|  | public int getMonth() |  |
|  | Looks good. |  |
|  | public int getDay() |  |
|  | Looks good. |  |
|  | public void setDate(int month, int day) |  |
|  | Looks good. |  |
|  | public String toString() |  |
|  | Looks good. |  |
|  | public boolean equals(MyDate d) |  |
|  | Looks good. |  |
|  | Implement this by comparing the month and day fields of each object, instead of comparing strings (comparing the numbers directly will be faster and more memory-efficient) | -6 |
|  | You did not implement this | -16 |
|  | public int daysInMonth() |  |
|  | Looks good. |  |
|  | A better way to do this is to have an array of integers, where slot 0 contains the number of days in month 1 (i.e., 31), slot 1 contains 28 (the number of days in Februrary), etc. |  |
|  | It would be substantially more efficient to create the array once per MyDate object (instead of once per time that daysInMonth is called) | -6 |
|  | Use the ‘fall-through’ capability of switch to list all the months that have 30 days, all the months that have 31 days, rather than listing each one individually | -6 |
|  | You did not implement this | -16 |
|  | public void nextDay() |  |
|  | Looks good. |  |
|  | The only real difference between the three cases is what number that you're comparing the **day** variable to.  **Make sure that you can re-write this so that you can do this without the switch** | -6 |
|  | Use daysInMonth to avoid the Giant Humongous Switch Statement | -6 |
|  | It would be better to compare the current month to **months.Length**, instead of comparing it to 12 directly |  |
|  | If month++ becomes 13 then you should reset it to 1 | -8 |
|  | You did not implement this | -16 |
|  | Style / Presentation |  |
|  | Looks good, overall |  |
| Identify the author of each file | You didn't put your name, etc, at the top of *EVERY*  source code file, including any files given to you. (The "etc" part includes the name of this class (“BIT 142”), and the year and quarter, and the assignment number, including the revision number) | -3 |
| Quality of presentation  (Make sure that what you've handed in looks good) | Code that you've commented out, because you're no longer using it, should be removed before handing in the assignment. | -6 |

**Grade (out of 150):**

Revision-Specific Grading Details:

**To calculate your grade:** add up all the (-1)'s and (-6)'s and (-X)'s, to get some negative number, then take that from the total to get your grade. For example: If the total points available for the assignment was 100, and you had the following penalties: -3 + -6 + -6 🡺 -15, so the grade would be 100 -15 = 85.

Why do you have to do this? Because this is only version 1, and so you won't really get your 'real grade' until you hand in the revision. Sometimes the grade on this first version appears really low (especially if you left out a whole section), and so I want to give people feedback, but try to avoid spooking people. Keep in mind that if you don't hand in a revision, this will be your final grade.

**Note**: Please note that if any of the above errors are duplicated within your code, you need to fix ALL INSTANCES of the error, even if it's not specifically listed above, in order to get the points on the revision.

**Note**: While the above list of errors is intended to guide your improvement of this homework, you should realize that a given error may occur in more places than have been specifically cited here. It is your responsibility to find all occurrences of a given error, and fix them all.

**Note**: Items that are 'greyed out' and *italicized* are there for informational purposes, to preemptively give you feedback for your revision, but these don't actually represent points that you've lost yet. When doing your revision make sure to pay attention to these items so you don’t lose these points in your revision. Example of a 'greyed out' item:

*You didn't do X. (-6)*