CPSC 224 Greyson Knippel

10/12/2025

hw-4

1. ***After reading the "Pre-Requisite Introduction":***
   - ***(a) What points do you think Martin is making regarding professionalism?***
     I think what Martin is saying with this introduction with respect to professionalism is that you can never master professionalism, or that it at least takes a very long time to master. He's also saying that disregarding professionalism is most likely not the best way to deal with your decisions in life, especially job decisions. He wants you to learn about his journey in professionalism and how we can learn from the foolish mistakes he made.
   - ***(b) Relate what he is saying to your own life experiences.***
     I can totally relate to Martin's experiences here, especially since I'm a 19 year old, being the same age as Martin in these introductions. I've never been fired from a job I've had, but I have certainly done some foolish things and disrespected people in the workforce. I actually had a circumstance similar to this at one of my old jobs where I quit because of the terrible pay, like this. But although I didn't have another job lined up before I quit, I didn't need to because I was heading back to school. Reading this excerpt will certainly be interesting for me since I'll be hearing things that will very likely happen to me when I grow up.

2. ***In the beginning of Chapter 1 under "Be careful what you ask for", Martin claims that a critical difference between being a professional and a non-professional concerns taking responsibility. In his example, he uses money as a proxy for real consequences where you accept ownership for outcomes, not just contrition (like saying you are sorry), and being willing to bear the cost of mistakes, whatever form that cost takes. Examples are time (e.g., staying late to fix the issue), reputation (e.g., admitting to others that you broke something), trust (e.g., re-earning confidence from teammates), or opportunity (e.g., forgoing new work until the problem is corrected). Think of a recent time in school or at work when you took responsibility for something that went wrong, not by apologizing, but by taking real action to fix it or prevent it from happening again.***

   - ***(a) Explain the situation and what it cost you (time, pride, effort, etc.)?***
     This situation has happened countless times to me mostly in the workplace. I make a mistake, it costs us, and I had the initiative to just simply fix the problem, and learn

from my mistakes. Of course it costs me pride and effort, but that's all expected and acceptable for me.

(b) Looking back now, how did the experience make you feel and what did you learn?
These experiences make me feel good honestly. I'm not a guy who likes to complain or listen to people complain, so fixing mistakes or correcting wrongdoings that I do without making it public or feeling sorry for myself gives me a dopamine rush. I've learned over the years that people also don't like to listen to your complaining or feeling sorry for yourself, so it's best to move on and learn.

(c) What do you think the advantages are of viewing professionalism in this way (for you personally)?
I think you definitely get a lot of respect & confidence for yourself, since you've proven that you can solve mistakes and problems by yourself. It gives a good mental reward to solving problems, while actually making the company better. This confidence that you gain in professionalism makes you a better person overall in life, allowing you to handle more situations effectively.

3. After reading the section First, Do No Harm (i.e., up to but not including Work Ethic), answer the following.

(a) Provide short summaries in your own words of what Martin means by "Do No Harm to Function" and "Do No Harm to Structure" and relate these ideas back to professionalism.
Do no harm to function basically means that our software should work as intended, by completely knowing what it does. It should be tested thoroughly, to the point where nothing is wrong about it. Unit tests should cover the entire program, and the QA team shouldn't find anything that would cause an error, or harm the function of the software.

Do no harm to structure is essentially saying that the structure of code is extremely important. Changes should regularly be made to your code, and if you cant make changes, your design is bad. It goes back to testing. You cant be afraid of making changes to your code if your structure is good and you have proper testing to make sure everything works quickly!

Relating both of these topics to professionalism, implementing these techniques will minimize your mistakes and will minimize the amount of times you will say "sorry" and have other people pick up the slack for you.

(b) Thinking about how you write code for your classes, assignments, etc., list at least three (but preferrably more) concrete and specific things you can do today as part of your process to avoid doing harm to function and structure.

Write unit tests for 100% of your code!!!

Use regression testing

Follow the iterative process

Structure your code so that it is extremely easy to make changes later. (use OOP)

Make your code easy to read and understand (coding is a collaborative process!)

(c) Generalize Martin's notions of doing no harm beyond just writing code to develop useful software for users, i.e., beyond just verification and the structure of code. Come up with your own similar notions for validation and usability that Martin might agree with.

Martin's notion of doing no harm again can improve your professionalism greatly. Taking the time to minimize any mistakes doesn't put blame on you and people's notion of you will be greatly increased.

some notions of validation and usability that martin might agree with are testing the code with real customers or customers that will use the product. Using real customers greatly increases your chances of having a perfect usability score. If the customer knows what they want, you know what you want to do for the code. And having them use it validates that the code as well as situations that can happen in the code.

4. After reading the section Work Ethic, answer the following questions.

(a) In your own words, summarize Martin's main points in this section regarding his notion of a "work ethic".

Martin's notion of work ethic revolves around a ton of things. He goes over professionalism, knowing a bunch of stuff revolving around your field, practicing and learning new things related to your field, collaborating with others and working alone, knowing that your employer's problem is your problem, and finally knowing when you are wrong, or having a little humility.

(b) Especially in computer science and engineering, many view college as the moment you enter a profession. Evaluate yourself according to Martin's notion of work ethic under the assumption that school is a job for you (as opposed to purely training for a future job).

Id say I'd do a decent job filling out Martin's work ethics according to my time here at Gonzaga. I try to be professional, I fix my mistakes, I work by myself and with others, I continuously practice and learn new things related to computer science by researching and doing my own projects, I've learned new programming languages, and I've definitely developed some humility since going to college.

(c) What things (e.g., habits, activities, etc.) can you improve, change, or develop now in your own work process to help boost your "work ethic" according to Martin's view. (Note that the items in "Know Your Field" are just his suggestions for when he wrote the book; today this list would have many of these items, but also many more!)

I think some things I definitely need to improve on are my collaboration skills, and practicing my skills outside of school. Sure, I learn new things, but I'm not really practicing any specific coding problem, etc. I think I should dedicate more hours out of my week to solely practice these programming skills (array skills, memory management, loops, classes, etc.)

5. ✅
6. ✅
7. For the overall design of TwoDicePig on this homework, I mainly structured the game class differently input an array list of players instead of just players. Most of the code I changed/added was related to the array list of players (computers or humans)

   I had to also change most of the methods in the TwoDicePig.java class. I added new methods to simply get input from the user; either in string or ints. I also changed the display score method a lot to accompany for the different stats I added to each player, and if the game was in tournament mode/double trouble mode.

   The methods I changed the most were the play turns, game setup, and main methods. I had to add a parameter in the play turn methods (game) to get more information out of the new game attributes of the game. I changed the main while loop to accompany for the double trouble mode as well. All of this is basically the same for the computer play turn method. I did have to add a new method to TwoDicePig.java, which was the find lead opponent, since it worked differently for finding the opponent in the array list. Basically

this new method finds the player with the most score, and sets them as the lead player. If there are no players with a higher score than the current player, the next player in the index simply gets chosen. And if there is only one player in the game, then a dummy player is created for the opponent.

The main changes I made in the game setup method, were asking the user for the game modifications and settings before starting the game. I had to write code to ask for player names, their player type, the winning score, tournament mode, and double trouble mode.

Finally the main method. I changed up this one a lot to accommodate for the tournament mode feature. Here I check for if the tournament is won, resetting for the new game (because you have to play multiple games for tournaments) and quitting the game manually too.

For the other classes, I only added simple functions for the game and player class, not anything special. Most of them are getters and setters for the new variables/stats added. Again, I mainly had to change the original simple get player and current player methods due to the array list being used.

8. ✅
9. Some challenges I had developing this homework project were modifying the older methods and accommodating for all the new variables and the while loops I had to change, it's definitely a lot to keep track of. I also had a ton of trouble making the tournament mode, due to it changing a LOT of the main program, at least for me.

   In my opinion, adding the new stuff in the requirements wasn't challenging, it was changing the old methods that had me struggling. It made me realize how important it is to write good, organized code in the first place in case you need to make any changes to it.

   And to overcome these challenges, I looked at the main scope of the feature I wanted to add. I didn't write code right away, but I envisioned the requirements, the variables, and how it would affect other parts of the program. I'd say that really helped me with this assignment.
10. ✅