PROJECT LOG

Content

Day 1 **Page 1-2**

Day 2 **Page 2**

Day 3 **Page 3**

Day 1

Plans Looking at the project in general, we notice we'll have to distribute the workload carefully so as to be able to complete the task. Today we will be focusing mainly on grasping exactly what the skeleton code provided does, particularly the Chunks and ControllerImp classes, so as to know things such as what kind of return types we will want throughout the code, how the output is going to work, and to understand the logger function and the status class fully. Today we should also decide what new classes we will have to make, using our second coursework as well as the provided sample solution, and what variables.

Achievements Today we managed to look through most of the code and understand it, although a couple of things such as how to use the Displacement class and the equals function that is found in a few classes are still not clear. However, I did manage to understand how logging works, which took a few tries at google searches and YouTube videos, as the concept of storing information about the program and then printing it depending on the error/warning/info/fine/finer/finest level that has been chosen was slightly abstract to begin with. We also created the Enum class to describe what mode the user is in, which we placed inside the ControllerImp class for readability.

Reflection We did not manage to do everything we planned to today, seeing as figuring out the code took a big part of our work day. It was not particularly clear to understand how exactly it all works together, or has to eventually, and I am not sure I have fully grasped it yet. I am still quite confused on how to use the Displacement class as well as not knowing exactly how the location for the user will be input (I have realised we are not writing a full App but only parts, so it is not fully realistic in terms of locating a user

and filling variables with coordinates) as it seems to be a very inflexible method of using the code. After looking at the Tests I have also realised I will need to learn more about writing them so as to make sure the code fulfils all the requirements for the coursework, as the tests given do not check for everything.

Day 2

Plans After working on the assignment for a couple of days (although not consequently as have other assignments due for similar dates, which has impacted progress slightly), including the first day of the log, we have made some progress in the form of creating the Location, Waypoint, Leg and Tour classes as well as writing some methods in the Create Tour mode. The classes are not finalized but we have populated them with variables and some constructor methods. This has been done through following of the code, starting at the Create Tour block of the ControllerImp class and seeing what we needed to have in the other classes. Today we hope to make more progress in these classes and perhaps finish the Create Tour block.

Achievements Today we worked for more than five hours on building up the classes as well as writing the first couple of methods available in Create Tour, as well as the constructor method for the Tour class. We also read over the Tests provided and have found some of the loopholes we will need to address once the code is more complete. After some more thinking, we finally understood how logging will work, and we have decided what kind of logging will occur at each importance level, for example, general information about what the code is executing at the moment will be kept at a fine level.

Reflection Our progress was much slower paced than anticipated, as we had not taken necessary details such as logging and producing errors, which took some time to add to the code we already had. It took us a few tries to clear up the code we had written - at first, we wrote many comments so as to keep track of the decisions we were making, but we have managed to finalise a couple of methods. Another reflection is that working with a partner on this project is particularly helpful, as having the other teammate to discuss our doubts gave solutions faster, and together we managed to solve most of them, placing us on the right track for the rest of the coursework. We now have a much wider and deeper view of how our program is going to work, which is very helpful as it means we can write more code.

Day 3

Plans It seems likely that this will be the last log we complete as we have not managed to advance with our code as much as we would've liked. The Chunks class is incomplete, and the appropriate Tests have not been created, and it seems unlikely we will be able to finish that before the deadline. We have worked for 8 non-consecutive days, which is fewer than previously intended but other deadlines and courseworks have come in the way. We have nearly finished the Create Tour mode of the ControllerImp with only some details such as the adding of leg to go. We will try to add the appropriate chunks to the getOutput method so as to receive some kind of output and hopefully not get a 0 for the code part of the assignment.

Achievements We finally decided on a Location class rather than using just Displacement, which we were ambiguous about at first, which has made several operations easier. The Create mode was finally completed after adding the few remaining things such as making sure the current tour becomes blank after being added to the tours list, as well as adding the leg direction aspect to the Leg class. We fixed a recurring problem that we had been receiving, due to a misunderstanding about declaring and initialising ArrayLists, as we had created, or so we thought, an ArrayList for the waypoints but we had not initialised it. After looking up information on the subject we came to know that in fact, when you only declase an ArrayList, you are merely creating a pointer to the memory location the ArrayList will be stored at. This mistake gave us errors when we tried to access it from the ControllerImp class's addWaypoint method.

Reflection The most problematic part of today's work was adding the addLeg method because we were not fully sure about how to use it. The controller does not have the option to use this method, meaning it is called implicitly, and so it was very confusing to know whether we should place it at the beginning or at the end of a waypoint so that the program would run smoothly. We also had to check many things for Waypoints, such as making sure a Waypoint is allowed to be one taking into account its radius and other Waypoints' locations. We found a problem in the case of a waypoint being close to a previous waypoint that was not the last. To check for these factors we created the isFurtherToThan and isAllowedIn methods.