.slide 1

So, the assignment will unsurprisingly be-

.slide 2

implementing Monte Carlo Tree Search in chess. You will be doing this in Unity, which I'm guessing you're all familiar with. But, on the off chance you aren't, don't worry, you don't have to do any modelling or anything like that, you are just going to have to program, the environment has been all set up for you.

Your task will specifically be to adjust the MCTSSearch and MCTSNode classes so that the program correctly performs MCTS.

.slide 3

This is going to be a pretty difficult assignment for many of you, which is why I decided to divide it into multiple smaller assignments so that you have to actually spend some time thinking about it throughout the semester instead of just trying to get it done the day before the deadline.

.slide 4

Starting today, you will get one assignment per week until the end of the semester, with the deadline being next week. Worry not however, I expect that these small assignments will be quite easy for you for the simple reason that they will not be required to work correctly. They will be required to *compile*, but that's it. I'll get to why in a second.

The small assignments will be the different parts of the MCTS algorithm – every week, you will be tasked with programming one small part of it and, at the end, you will have to put it all together. The reason why the individual parts will not be required to work is that I didn't provide you with any tests for the individual parts, so you have no way of knowing whether they work or not (although, you are of course free to come up with your own tests). And the reason why I didn't do so is twofold – first, some of the parts would be difficult or even impossible to test without the rest of the algorithm as they can be programmed in various ways, and second, the point of this assignment isn't just to create a correct implementation of MCTS but also to learn to debug it because that is something akin to what you may encounter in practice as AI programmers.

So, the plan is as follows: today, you will be tasked with implementing the node of the search tree that the algorithm will use. Then, the selection step, the expansion step, the simulation step, the backpropagation step. The deadline for all of these will be the lecture after they are assigned. And, then, finally, you will be tasked with putting it all together. The final deadline will be the end of February, which should be something like two weeks or a week and a half after the end of the exam period. This deadline is fixed and will not be extended without sufficient reason. As for the other deadlines, you can miss 3 of them – more than that will result in failure. Also, if you miss a deadline, you have to deliver what you were supposed to program by the next deadline.

The reasoning behind this schema is that, as this is a complicated algorithm, I am hoping that having the assignment spaced out like this will make you think about it throughout the semester so that, by the end of it, you will have a good enough understanding of it to complete the assignment without too much

trouble. I will also try to provide you with feedback if I'll notice that you've misunderstood something about how the algorithm is supposed to work.

To be clear though, you can feel free to skip ahead and finish the whole assignment sooner.

.slide 5

.slide 6

Ok, what follows is a description of the project you'll be working with, the API, etc. It would be sort of pointless to go through that now, as you most likely wouldn't remember any of it, so you can take a look at it in your own time. Let's skip to…

.slide 7 – 37

.slide 38

the End Notes.

.slide 39

There's some general debugging advice here which you should read before trying to put the whole algorithm together, but what I want to mention here is this-

.slide 40

Your program will be evaluated using 15 tests where it will be supposed to checkmate its opponent. The program will have a limited number of simulations at its disposal, and if it won't be able to find the correct solution within that limit, the test will fail. I tried to set the limits reasonably, but, since this is a stochastic algorithm and our implementations may differ, it can happen, that your solution will not pass all the tests despite being correct. You therefore *don't need to pass all the tests* before sending me the solution.

What's more, as I've already mentioned, this is going to be a difficult assignment for most of you. It is therefore perfectly ok to contact me when you'll get stuck or when you'll need something explained more thoroughly. I created this assignment expecting that to happen. However, also be aware that if you all start messaging me three days before the deadline, I probably won't be able to help you all, so try not to wait for the final deadline before working on the final part of the assignment.

.slide 41

Once you finish, send your solution to me on Discord or through e-mail. *Do not send me the entire solution*, just the files that you changed. And, of course, if you discover some bugs or things that could be done better, let me know. And that's it.