Exam QA:

- None of what I say supersedes exam instructions.
- The external examiner may see things differently
- I can't/don't say which subjects are in/are not in the exam
  - Exams follow a 20/80% rule: 20% of the subjects are part of 80% of the exams
  - I will talk about the 80%
  - I can give examples of what would typically be in an exam like this
  - I can say which topics would typically make for good exam questions
  - ..but think about the 20%!
- Check your exam registrations

Exam instructions

Like midterm A + B; spring 2023 exam: 24 questions/sub-questions in total (9 + 3 + 3).

- Part I: Right answer: **+3**, Wrong: **-1**, Random answer: **0** (average).
- Part II: **Write justifications** (same setup as introduction to intelligent systems)
  - No justifications required for Part I and III
- Part III: Make sure midterm A&B; spring 2023 works (no internet) – see the video online and try yourself.

Autograding, etc.

- The tests/grade script check the same things as the print-statements in the code
- I don't try to be tricky with the other tests
- The `.token`-format tells me how many points you got on your computer
- Be careful with global variables (the tests can address this to some degree)
- *'What if the .token file does not work'*: Hand in anyway and write a note!

Observations from spring 2023

- Approximately the same average score in the three sections
  - But some 'skipped' the programming section
  - **Can you improve your score by solving 1-2 of the most simple programming problems?**
- Some misunderstood the evaluation: I.e. writing text-solutions to the multiple-choice section, or non-code solutions to a programming problem ("I think that the problem could be solved with the bandit algorithm from week 3...")
- A partial solution is often better than no solution
- Some had no justifications in part 2; some had **very long** justifications.
  - Mindset: **always** give **a** justification. *Enough so a reasonable person can see what you are thinking*.

Observations from spring 2023

- Maple can mess you up!
  - If you apply a strategy where they solve 'simple' tasks with maple, check that the result 'looks reasonable'
  - Example: mis-typed a symbol in maple so the 'simplify' command gives a long answer
- See if you can check your answer
  - Example: You need to find x and using an equation from the book you get $3x + 1 = 2x - 9$ therefore $x = 2$(!).
    - Insert $x = 2$ in the equation and see if it works – it is a much better check than going through intermediary calculations!.
    - This works for any simplification (differential equations, linear algebra, etc.).

Studying for the exam 80/20

- Old exam sets (midterm A&B, spring 2023)
- Quizzes
- Weekly exercises (many taken from exams)
  - Some of the exercises are not suitable for an exam (c.f. week 5)
- Projects
  - Note how some topics re-occur between spring 2023 exam, exercises, and the projects (Write a DP Model and apply DP, implement a control model, implement a MDP...)
  - But obviously not all project questions are suitable for the exam

Exam subjects

- Most subjects are not 'too hard', but 'too difficult to ask about'
  - Too much setup/text required
  - Too much code
  - Too much "Can you tell what I am thinking about"
- Often, these subjects are better explored using Multiple-Choice
- So you should look for subjects that are:
  - Simple/self-contained
  - Have very clear definitions
  - Are easy to ask new questions about (i.e., change parts of the problem)
  - Can be considered a core part of the material
  - (ideally) was part of a central problem/example (exercise/project/lectures)

Excellent subjects if you are making an exam

Finite-horizon Dynamical programming:

- Given a problem description, can you solve it with DP?
  - Given a problem description, can you 'translate' that into how f_k, g_k etc. are defined?
  - Inventory control, LQR, etc.
  - Can you manually apply the DP update rule in a given problem? (see lecture notes)
- Given a problem description, can you implement a `DPModel` class and solve it with DP?
  - Task example: Determine the expected cost
  - Task example: Compute optimal action in a given state

Excellent subjects if you are making an exam

Control theory:

- Control: Given a differential equation, can you
  - Simulate it; understand how control relate to differential equations
  - Discretize it (Euler, EI)
  - Linearize it
  - apply LQR to it
- Cost functions: Understand what a quadratic cost function does. What is the role of Q and R. Understand an optimal policy is one that minimize the cost.
- Programming: Translate the differential equation into a `ControlModel`; simulate/discretize/Linearize it.
- PID: Can you apply it to a problem? Do you know what the parameters do?

Excellent subjects if you are making an exam

Reinforcement Learning:

- MDPs:
  - Can you write down a Bellman equations for e.g. v_π (see spring 2023 exam)
- Programming:
  - Given a problem description, can you translate it into a MDP class and use it to perform tasks such as determining v_π or the optimal policy? (policy evaluation, value iteration)
  - See the Sarlac problem in project 3.
  - Implement 'simplified' versions of existing algorithms (see Midterm B)
- Tabular RL: Here is a gridworld example. Imagine we take a step. What does TD, Q-learning, MC-learning or Sarsa do? (as in the lectures)
- Bandits: Here are some actions/rewards. What does simple bandit/UCB do?
- Understand key quantities such as v_π, v_*, q_π, q_*, G_t.