The slide numbers refer to the 2024 version of `reinforcement_learning_annotated.pdf`. Some of the required knowledge is only pictured in the annotations.

Similar to the guides by T. Brázdil, the list presents the *minimal* requirements for passing the exam (i.e., at getting grade at least E). Requirements for better grades are not specified, though I will definitely not require the complete knowledge of the slides - use your own judgement (e.g. I will definitely not ask about the various psychologists from the intro lecture).

Unless explicitly stated otherwise, graphs and tables showing experimental evaluation of the algorithms (taken from a third-party literature) serve an illustrative and motivational purpose and their knowledge is not required.

There will be no "random drawing" of questions at the exam, I will ask questions freely with the main intention of testing your **deep** understanding of the topic. Unless stated otherwise, if I write "you must know," I mean "you must know and be able to explain." You will always have time to prepare and write down notes.

Given that this is the first iteration of the course, I will only count the best mark from all your attempts (though I will always enter the latest grade into IS so as to have a record of your exam attendance). However, I do not plan to open more exam terms than currently published.

- Slides 1-15: This is just motivational intro, no knowledge required here, though having some understanding of the history can help you consolidate the overall picture of the field.
- Sl. 16-29: These are crucial definitions, you need to know everything in detail and be able to explain the meaning of the objects.
- Sl. 30-60:
    - Sl. 31: You need to know that optimal values/policies in an MDP are solvable in PTIME.
    - Sl. 32-35: You must know the Bellman evaluation and optimality equations + the corresponding operator and be able to intuitively explain why they hold.
    - Sl. 36-38: You must know that the operator has the property stated in Theorem 20. More in-depth knowledge is required for better grades than E.
    - Sl 39.; Essential, you must know and be able to explain the algorithm.
    - Sl. 40-44: You don't have to know the formulae in the theorems but you should be able to explain the "message" of at least one of the theorems. However, you **must** know the concept of greedy policy, it is absolutely essential.
    - Sl. 45-51: You must know the exact statement of the policy improvement theorem and be able to formulate the PI algorithm. For E, I do not require to know the proof, but we can discuss the proof when aiming for a better grade.
    - Sl. 52-54: Again, you must know the theorem statement but not the proof. You should be able to explain why the complexity is at most exponential.
    - Sl. 55-59: I will discuss this only with those aiming for better grade than E. However:
    - Sl. 60: Is essential, since it describes the backbone of most RL algorithms. The previous slides help to see the idea behind GPI.
- Sl. 62-65: Everything, it provides the necessary context. Be prepared to discuss whether some given algorithm is off-policy/on-policy.

- Sl. 66-92:
  - Sl. 66-70: You must know FV and EV Monte Carlo by heart and be able to argue why FV MC converges.
  - Sl. 71-78: You must know all the presented MC control algorithms, including the concepts IE, ES and eps-greediness (all essential).
  - Sl. 79-80: Only for higher grades. However, you should be able to sketch *some* correct pseudocode of an off-policy MC algorithm.
  - Sl. 81-85: Absolutely essential, you need to know everything deeply, including the proof of Theorem 39.
  - Sl. 86-87: You should know what weighted importance sampling is, but I will only go into details with those aiming for (at least slightly) higher grades.
  - Sl. 88-91: This is mostly summary, so nothing concrete required here for E, but again: you should be able to sketch *some* correct pseudocode of an off-policy MC algorithm.
- Sl. 93-113:
  - Sl. 93-96: You absolutely need to know how TD(0) works and what is bootstrapping.
  - Sl. 97-98: Nothing concrete, this mostly provides additional context.
  - Sl. 99-107: Everything in-depth, this is absolute basics.
  - Sl. 108: This also appeared in HW01, so you should be able to discuss what happens in the example.
  - Sl. 110-113: You should know that there is something like maximization bias. You must know what double Q-learning is, including the update formulae, and intuitively argue how it solves the bias (though I will be rather lenient about the quality of the argumentation).
- Sl. 114-136:
  - Sl. 114-121: You need to be able to define n-step updates and know their relationship to TD(0) and MC. You need to be able to formulate RL algorithms with n-step updates.
  - Sl. 122-123: Only for higher grades.
  - Sl. 124-126: You need to know at least one of the formulations of lambda-return on slide 125.
  - Sl. 127: Everything.
  - Sl. 129-132: You need to be able to intuitively explain the difference between forward and backward view and why we consider the backward view at all. Ideally, you should be able to give the definition of eligibility trace from slide 130, though I will tolerate errors in the formulae: I can help you with reconstructing the formula and the pseudocode on line 132, but you should be able to explain what it means.
  - Sl. 133-134: You should be able to show the equivalence for lambda = 0 and comment on how it is with the equivalence for general lambda. The computation on slide 134 is only for higher grades (and I will be willing to help here).
  - Sl. 135-136: Mostly context, nothing concrete here.
- Sl. 137-154: Here you must know almost everything.
  - Sl. 137-142: This is mostly context. I will likely not directly ask about these things but their understanding is implicit in like everything in the remainder of the course. 😉
  - Sl. 143-147: Essential things, you must know everything in detail.
  - Sl. 148-153: Also everything, including why TD methods are semigradient.

    - Sl. 154: You should also know this, but this is a simple generalization to what you know by far.

- Sl. 155-189:

    - Sl. 155-158: Just context, nothing concrete.
    - Sl. 159: Essential, must know.
    - Sl. 160: Just context.
    - Sl. 161-165: Everything essential. You of course do not need to memorize the pseudocode, but in principle you should be able to reconstruct the pseudocode from the previous knowledge.
    - Sl. 167-169: Again essential.
    - Sl. 171-175: You should understand the partial observability of Atari and that DQNs store a bounded history of frames (ideally illustrate on example analogous to slide 172). The rest is more like context. Do not memorize the hyperpamater values.
    - Sl. 176-189: You do not need to recall all six heuristics. However, you must know the definition of the advantage function (slide 178) and be able to explain the workings and the rationale behind the dueling architecture, at least intuitively. You must also know Double DQN. The rest is for higher grades.

- Sl. 190-210: A lot of essential stuff, be prepared to know almost everything.

    - Sl. 190-192: Essential knowledge.
    - Sl. 193-194: I will help you with the derivation, but you must be able to complete it. The focus here is testing whether you can perform gradient arithmetics: I can give hints of the form "Now use property XXX here" and I expect you to perform a correct rearrangement of the equation. The number of necessary hints will be reflected in the final grade. When I say "now use the log-probability/likelihood ratio trick," I expect that you will understand what I mean.
    - Sl. 197-199: Same as above.
    - Sl. 200-202: Everything. With my help you should be able to proof the baseline theorem (I left it as an exercise in the lecture, but the derivation is really at the level of Calculus 101, so with my hints this should pose no problem.)
    - Sl. 204: You should understand the usage of the term ``advantage estimation'' in PG methods.
    - Sl. 206-208: You must know everything.
    - Sl. 209-210: This is mostly context, but you should at least understand entropy regularization (it is also used elsewhere).

- Sl. 211-228: I understand that PPO is complex. Still, I expect that *with my help*, we should be able to get to the formulation of the algorithm. Things that you have to internalize are:

    - The high-level structure of the algorithm, particularly slide 214 (the previous slides serve as a useful context for this).
    - You must memorize Theorem 55 (proof only for higher grades, I will help here).
    - Why we cannot use the loss from Theorem 55.
    - The surrogate loss (slide 216) and that it relates to importance sampling.
    - The concept of trust region and how it relates to the original and surrogate loss (explained on slides 217-218, you do not need to memorize Theorem 56).
    - That we cannot optimize the loss directly and instead optimize the sample loss (slide 219, I can help with the formula).

  - You should then understand that there are various ways of ensuring that we stay in the trust region, though I dnot require you to know TRPO exactly.
  - The above points should give you a general understanding of trust region methods. We will then turn our attention to details of PPO. You should be able to explain its key distinguishing components: GAE and loss clipping.
  - I ask you to memorize the clipped loss (slides 223 and 224), though if you make slight error in the formula, it is not a big issue. What is more important is that once we arrive at the correct formula, you will be able to discuss its meaning (see, e.g., slide 224).
  - On the other hand, you should perfectly know what GAE is.
  - Sl. 226-228: You should know that PPO is an actor-critic method (i.e., we have the value loss) and that it uses entropy regularization, though I will not push you to reconstruct the total loss or the algorithm pseudocode. Still, you will need to somehow assure me that you understand how the individual components of PPO click together (maybe via diagram or just text description).

- Sl. 229-247:

  - Sl. 229-230: This is just introductory context.
  - Sl. 231-236: Everything in detail, including the lower bound on the regret of greedy policy.
  - Sl. 237-242: You do not need to prove the whole lower bound. However, you should be able to roughly formulate the Hoeffding bound (you can get the constants wrong, understanding the exponential decay is more important). You must memorize Theorem 61.
  - Sl. 243-246: You must know the UCB formula. Its derivation will onluy be necessary for a higher grade.

- Sl. 248-266:

  - This is similar case as PPO, basically I want you to demonstrate a knowledge of what MCTS does and how.
  - Sl. 248-252 provide the necessary context. In particular, you should understand the terms *online algorithm* and *unfolding* (so that you understand why MCTS can be applied to arbitrary MDPs).
  - Sl. 253-256: This again relates to the high-level operation of MCTS. I do not require you to produce exactly the same pseudocode as in the slides, but you must be able to legibly explain the workings of the algorithm, what a *search tree* is and where is it used.
  - Sl. 257-265: You must be able to name the four phases and provide at least their high-level description. If the description is not clear enough, I will push you (and help you) to produce the pseudocodes. You must know where and how is the UCT formula used, how does the tree grow, what a rollout is, and how are the statistics updated.
  - Sl. 266: You must know how actual action selection proceeds.

- Sl. 267-294:

  - For AlphaZero, I will leave it up to you on whether you want to describe it as a game-solving or as an MDP-solving algorithm. However, for better grade then E you must know how are the minimizer vertices handled throughout the algorithm.
  - Essential is to describe the novelty compared to plain MCTS: where and how is the function approximator used, what quantities does it predict, where and how are these quantities used.

    - I will not require explaining Dirichlet noise for E.
    - You have to understand how training works, at least on a high level (slide 282).
    - Sl. 283-285 are for higher grades only.
    - MuZero is for higher grades only, and even then I will not ask you to explain the training.