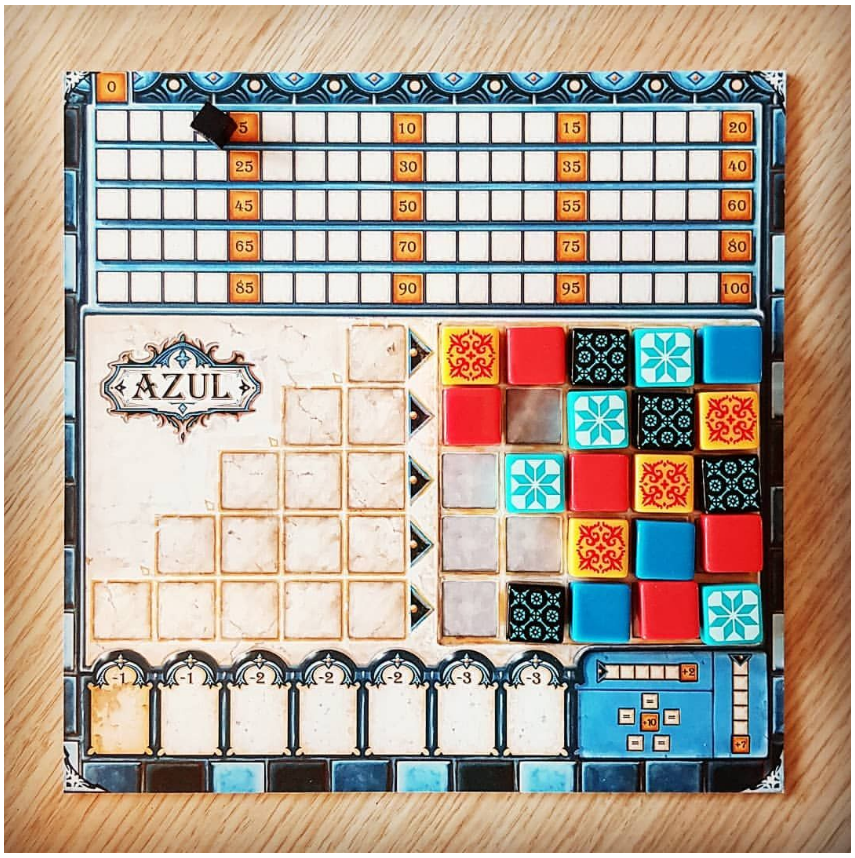
Azul Project Template

You must fully and carefully read the assignment specification and instructions detailed in this file. You are NOT to modify this file in any way.

- **Course:** COMP90054 AI Planning for Autonomy @ Semester 2, 2024.
- **Instructor:** Professor Adrian Pearce & Associate Professor Nir Lipovetzky
- **Deadline Team Registration:** Monday 16 September, 2024 @ 18:00 (start of Week 9)
- **Deadline Preliminary (Individual) Submission:** Friday 20 September, 2024 @ 18:00 (end of Week 9)
- **Deadline Final Submission:** Monday 14 October, 2024 @ 18:00 (start of Week 12)
- **Deadline Individual Self-Evaluation (Final):** Friday 18 October, 2024 @ 18:00 (Week 12)
- **Course Weight:** 35% total, comprising 10% (preliminary submission) + 25% (final submission)
- **Assignment type:** Groups of 3 (required) or 2 (special case)
- **CLOs covered:** 1-5



The purpose of this project is to implement an autonomous agent that can play the game Azul and compete in the UoM COMP90054-2023 Azul competition:

Please read carefully the rules of the Azul game. Azul can be understood as a deterministic, two-player game. Understanding the results and different strategies is important for designing a good agent for this project. Additional technical information on the contest project and how to get started can be found in file azul.md.

Table of contents

1. Your task

This is a **group project** of 3 members. Now that you have a repo, the next thing to do is to register your team in the Project Contest Team Registration Form provided in the ED A3 release announcement (assuming you followed ED instruction and got a Canvas Teams Name) and tell the other students to join the team in GitHub Classroom.

Your task is to develop an autonomous Azul agent team to play the **Azul Contest** by suitably modifying file `agents/<t_XXX>/myTeam.py` (please replace "XXX" in "<t_XXX>" with your own Canvas Teams ID and there maybe some other auxiliary files you may implement). For example, if your team's Canvas Teams Name is "Canvas Teams 1", then you should change your folder from `agents/t_XXX/` to `agents/t_001/`. The code submitted should aim to be commented at high standards, be error-free, and *never crash*.

In your solution, you have to use at **least 3 AI-related techniques** (**2 techniques at least for groups of 2**, but it is highly not recommended.) that have been discussed in the subject or explored by you independently, and you can combine them in any form. **We won't accept a final submission with less than 3 (or 2 depends on your final group size) techniques**. Some candidate techniques that you may consider are:

1. Blind or Heuristic Search Algorithms (using general or Azul specific heuristic functions).
2. Classical Planning (PDDL and calling a classical planner).
3. Policy iteration or Value Iteration (Model-Based MDP).
4. Monte Carlo Tree Search or UCT (Model-Free MDP).
5. Reinforcement Learning – classical, approximate or deep Q-learning (Model-Free MDP).
6. Goal Recognition techniques (to infer intentions of opponents).
7. Game Theoretic Methods.

We recommend you to start by using search algorithms, given that you already implemented their code in the first project. You can always use hand coded decision trees to express behaviour specific to Azul, but they **won't count** as a required technique. You are allowed to express domain knowledge, but remember that we are interested in "autonomy", and hence using techniques that generalise well. The 7 techniques mentioned above can cope with different games much easier than any decision tree (if-else rules). If you decide to compute a policy, you can save it into a file and load it at the beginning of the game, as you have 15 seconds before every game to perform any pre-computation.

Important basic rules

When submitting a solution, please make absolutely sure you adhere to the following rules:

- Your code **must run *error-free* on Python 3.8+**. Staff will not debug/fix any code. If your code crashes in any execution, it will not have any valid results from the online server. To ensure your code is error-free, you can test with docker locally (following this link)
- You can install Python 3.8 from the official site, or set up a Conda environment or an environment with PIP+virtualenv.
- Your code **must not contain any personal information**, like your student number or your name. That info should go into the homepage of your wiki. If you use an IDE that inserts your name, student number, or username, you should disable that.
- You are **not to change or affect (e.g., redirect) the standard output or error channels** (`sys.stdout` and `sys.stderr`) beyond just printing on standard output, including `logging`. If your file mentions any of them it will be **breaking the "fair play" of the course (see below)**. These are used to report each game output and errors, and they should not be altered as you will be interfering negatively with the contest and with the other team's printouts.
- Being a group assignment, you must **use your project Github** repository and GitHub team to collaborate among the members. The group will have write access to the same repository, and also be members of a GitHub team, where members can, and are expected to, make commits from their own github account (both code and wiki), engage in discussions and collaboration. Refer to the marking criteria below.

2. Deliverables and submission

Pencil the following info in your calendar.

Team Registration: (18:00 Monday Week 9)

Please follow the instruction in this ED post: https://edstem.org/au/courses/17752/discussion/2188778

Every team that **does not contain 3 members** (In Canvas Team page) after the deadline will be randomly regroup to teams with 3 members by the staff member.

Preliminary (Individual) submission (18:00 Friday week 9) (10 Marks)

This is an individual submission, **each member** should finish the following steps:

1. The `agents/<t_XXX>/myTeam.py` implementing your AI-based Azul agent team as per instructions above by tagging the relevant commit as "`<your_student_id>`". It is recommand to create your own branch to develop (please try not to use your student id as the branch name to avoid confusion in pushing tags.)
2. Submit your individual self evaluation to Canvas.
3. Fill the Project Certification & Contribution Form (PRELIMINARY).

> ⚠️ Each member of the team should fill a separate certification form. Members who do not certify will not be marked and will be awarded zero marks.

Wiki and Final Submission (18:00 Monday week 12) (25 Marks)

In the **final submission** you are to submit your final submission as a group, which includes:

1. The `agents/<t_XXX>/myTeam.py` implementing your AI-based Azul agent team as per instructions above by tagging the relevant commit as "`submission`".
2. A **wiki** in your GitHub team repository, documenting and critically analysing your Azul agent system. Take a look at the **wiki-template** provided as a guideline of the structure that you should follow. <!-- * At the very minimum the wiki should describe the approaches implemented, a small table comparing the different agents/techniques you tried showing their performances in several scenarios (briefly the table), and an analysis of the strengths and weaknesses of your solution. For example, you may want to show how the addition of a given technique or improvement affected your system at some important point in the development.
    - However, you can also discuss other aspects, such as other techniques you experimented with (and why they were not used in the final system), future extensions or improvements on your system, etc. -->

And **each member** should submit:

3. A filled Project Certification & Contribution Form (FINAL).

> ⚠️ Each member of the team should fill a separate certification form. Members who do not certify will not be marked and will be awarded zero marks.

Submit your project substantially before the deadline, preferably one day before. Submitting close to the deadline could be risky and you may fail to submit on time, for example due to loss of Internet connection or server delays. There will be **no extensions** based on these unforeseen problems.

Individual Reflection (18:00 Friday week 12) (affecting 25 Marks)

As part of your final assignment submission, each team member is required to submit a short (one page maximum) PDF individual reflection to Canvas (under Assignments). This reflection will serve as your contribution report, it should include the following information:

10 marks for the code submission (final team code submission, not individual preliminary submission) and 15 marks for the wiki, you should consider the following and provide specific information:

Teamwork: whether you contributed meaningfully to your team, whether you did what was asked, whether you followed good teamwork and software engineering principles.

Technical contribution: **which technique** did you implement that was useful to your team; e.g. it was part of the tournament, it was useful for running comparisons, etc?

Written contribution: what you have **specifically** written to the wiki in terms of clear and concise content

Learning: What did I learn about working in a team? What did I learn about artificial intelligence?

3. Online Server

The online server is available for testing your code performance. Here is the link: http://115.146.95.253/www_a3/index.html

You should be able to see your team's tournament results from the A3 homepage, and detail of the individual tag and team tag results in your team's page.

The server will pick up the following tags for different purposes:

- `<student_id>`: these tags will be picked up and ran individual jobs for prelimianry submission. The last record will be used to mark your code performance.
- `test-submission` and `submission`: these tags will be picked up and ran group jobs. The last record of `submission` tag will be used to mark your team's code performance. You can use `test-submission` to test code without affect your marks.
- `tournament-submission`: this tag will be used to run tournament. That is, your code compete against all your classmates and staff teams.

The status of an individual submission could be:

- `unverified`: It means the server has picked up your updated tag, but has not submitted a verification job on it yet
- `verifying`: it means your commit is under verification by the server
- `verified (I)`: it means your commit passed verification and is ready for the individual jobs

- `individual_running`: it means the server is running individual jobs for your commit, there should be 10 games in total
- `verification_Failed`: it means your commit is failed to run against itself.
- `finished`: it means all job for this commit is done. If it is a group commit, then it will be put into the tournament.

The status of an group submission could be:

- `unverified`: It means the server has picked up your updated tag, but has not submitted a verification job on it yet
- `verifying`: it means your commit is under verification by the server
- `verified (G)`: it means your commit passed verification and is ready for the group jobs
- `group_running`: it means the server is running individual jobs for your commit, there should be 10 games in total
- `verification_Failed`: it means your commit is failed to run against itself.
- `finished`: it means all job for this commit is done. If it is a group commit, then it will be put into the tournament.

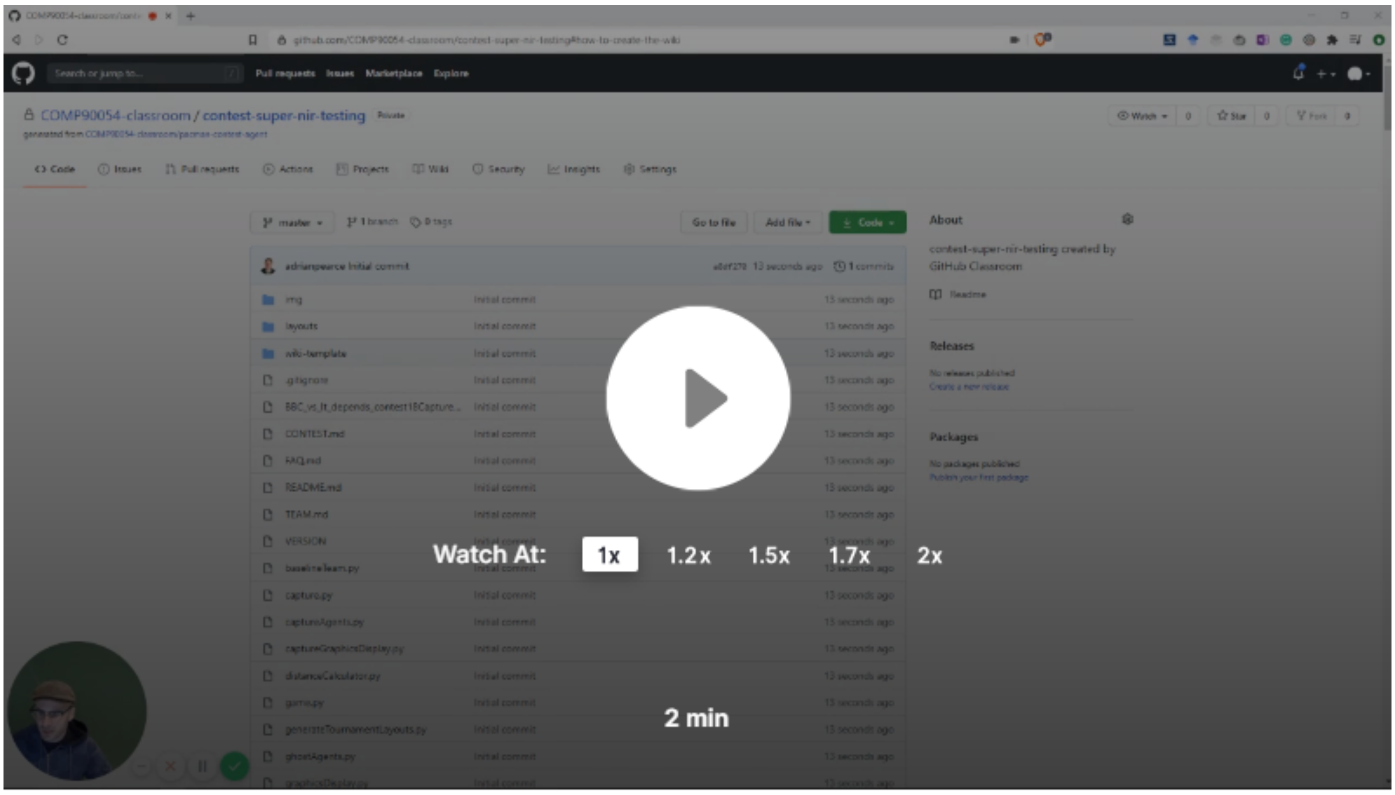
4. Marking Criteria

To be released soon.

5. Important information

How to create the Wiki

You can use the template given in wiki-template (will be released soon) folder in order to create your wiki. Watch the video below.



Corrections

From time to time, students or staff find errors (e.g., typos, unclear instructions, etc.) in the assignment specification. In that case, a corrected version of this file will be produced, announced, and distributed for you to commit and push into your repository. Because of that, you are NOT to modify this file in any way to avoid conflicts.

Late submissions & extensions

Late submissions are truly inconvenient for this large assessment, as it involves other team members working for several weeks. Late submissions may not enter into the "official" contest and the team may then not receive feedback on time. The project is available for 6 weeks, as a team, each member should plan and start early in order to minimize any unexpected circumstances near the end. Extensions will only be permitted in *exceptional* circumstances; refer to this question in the course FAQs. Note that workload and/or heavy load of assignments will not be accepted as exceptional circumstances for an extension (we are not allowed to give any extension beyond Week 12 either).

About this repo

You must ALWAYS keep your fork **private** and **never share it** with anybody in or outside the course, except your teammates, *even after the course is completed*. You are not allowed to make another repository copy outside the provided GitHub Classroom without the written permission of the teaching staff.

> ***Please do not distribute or post solutions to any of the projects.***

Academic Dishonesty

Academic Dishonesty: This is an advanced course, so we expect full professionalism and ethical conduct. Plagiarism is a serious issue. Please **don't let us down and risk our trust**. The staff take academic misconduct very seriously. Sophisticated *plagiarism detection* software (e.g., Codequiry, Turinitin, etc.) will be used to check your code against other submissions in the class as well as resources available on the web for logical redundancy. These systems are really smart, so just do not risk it and keep professional. We trust you all to submit your own work only; please don't let us down. If you do, we will pursue the strongest consequences available to us according to the **University Academic Integrity policy**.

We are here to help!: We are here to help you! But we don't know you need help unless you tell us. We expect reasonable effort from you side, but if you get stuck or have doubts, please seek help. We will ran labs to support these projects, so use them! While you have to be careful to not post spoilers in the forum, you can always ask general questions about the techniques that are required to solve the projects. If in doubt whether a questions is appropriate, post a Private post to the instructors.

Silence Policy: A silence policy will take effect **48 hours** before this assignment is due. This means that no question about this assignment will be answered, whether it is asked on the newsgroup, by email, or in person. Use the last 48 hours to wrap up and finish your project quietly as well as possible if you have not done so already. Remember it is not mandatory to do all perfect, try to cover as much as possible. By having some silence we reduce anxiety, last minute mistakes, and unreasonable expectations on others.

6. COMP90054 Code of Honour & Fair Play

We expect every UoM student taking this course to adhere to the **Code of Honour** under which every learner-student should:

- Submit their own original work.
- Do not share answers with others.
- Report suspected violations.
- Not engage in any other activities that will dishonestly improve their results or dishonestly improve or damage the results of others.

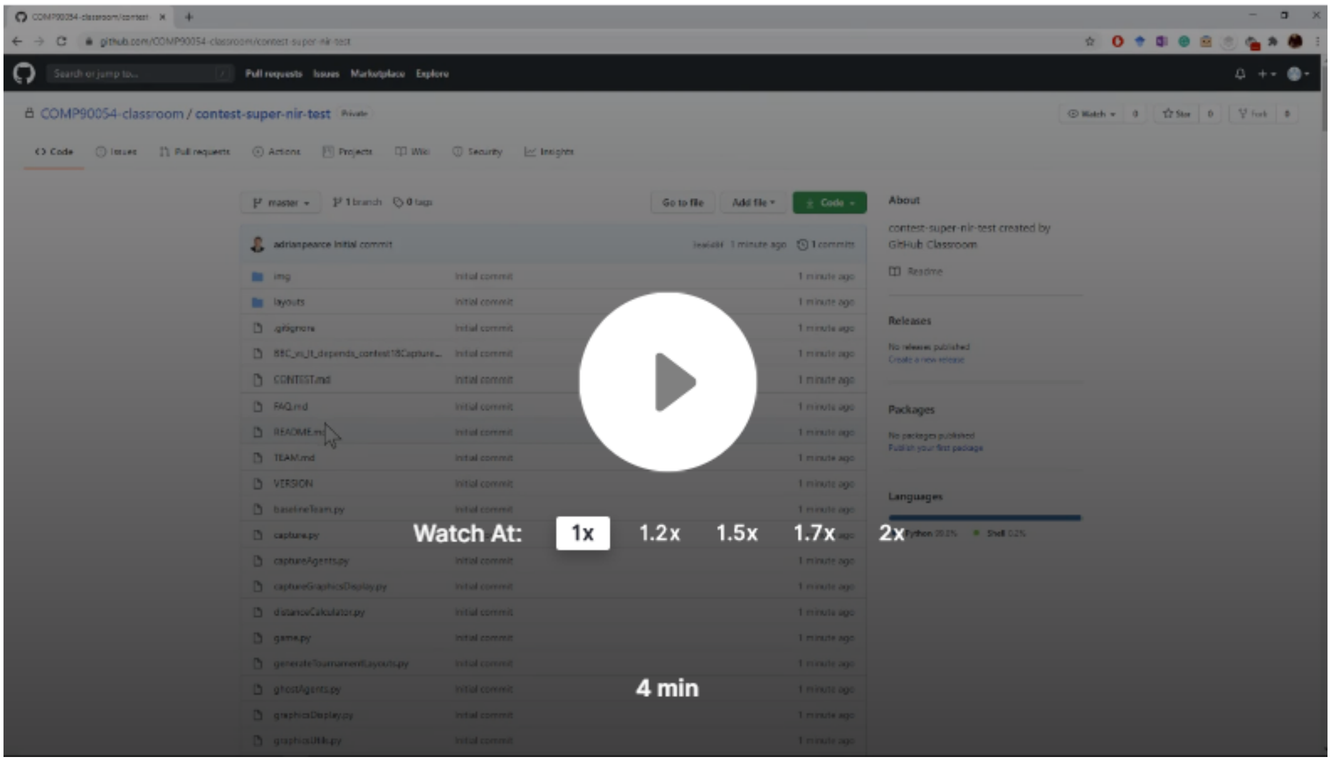
Being a contest, we expect **fair play** of all teams in this project. If you are in doubt of whether something would break the good spirit of the project, you must check with us early, not wait to be discovered. Any behaviour or code providing an unfair advantage or causing harm will be treated very seriously. We trust you, do not let us down and be a fair player.

Unethical behaviour is extremely serious and consequences are painful for everyone. We expect enrolled students/learners to take full **ownership** of your work and **respect** the work of teachers and other students.

7. Teamwork and Software Engineering professional practice

Besides the correctness and performance of your solutions, you must **follow good and professional SE practices**, including good use of git and professional communication during your development such as:

- *Commit early, commit often:* single or few commits with all the solution or big chunks of it, is not good practice.
- *Use meaningful commit messages:* as a comment in your code, the message should clearly summarize what the commit is about. Messages like "fix", "work", "commit", "changes" are poor and do not help us understand what was done.
- *Use atomic commits:* avoid commits doing many things; let alone one commit solving many questions of the project. Each commit should be about one (little but interesting) thing.
- *Use the Issue Tracker:* use issues to keep track of tasks, enhancements, and bugs for your projects. They are also a great way to collaborate in a team, by assigning issues and discussing on them directly. Check GitHub Mastering Issues Guide.
- *Follow good workflow:* use the standard branch-based development workflow, it will make your team much more productive and robust! Check GitHub Workflow Guide.
- *Communicate in the GitHub Team:* members of the group are expected to communicate, in an adequate and professional way, in the GitHub team created along the repo. For example, you could use GitHub team discussions, use issues and pull requests to track development status, or create project plans. Video and voice chats outside of GitHub are permissible (and encouraged), but text communication should be through the GitHub team where possible.



- *Pair program* if possible. You can use VScode and this extension to liveshare your local code with your team members as guests. In pair programming, take turns at who's hosting the session (driving the coding) and who's observing. Alternatively, use any online platform to share your screen and program with your team members. Pair Programming is a widely used practice in industry. it is known to reduce errors, improve code quality, improve learning of all members, and reinforce the quality of the team's communication. See this entry about pair programming

We will also inspect the **commit history** and **GitHub team** to check for high-quality SE practices and meaningful contributions of members. The results of this check can affect the overall mark of the project and point deductions may be applied when poor SE practices have been used along with uneven team members contributions (reported in the submission form). For example, few commits with a lot of code changes, or no or poor communication in the corresponding GitHub team may result in deductions, even if the performance is perfect. We need to make sure that you work as a team where everyone is contributing. This is a key skill in industry. “Effective teamwork begins and ends with communication.” — Mike Krzyzewski.

8. Conclusion

This is the end of the project specification.

> 📢 Remember to also read the azul.md file containing technical information that will come very useful.

If you still have doubts about the project and/or this specification do not hesitate asking in the ED Discussion Forum and we will try to address it as quickly as we can!

GOOD LUCK and HAPPY AZUL!