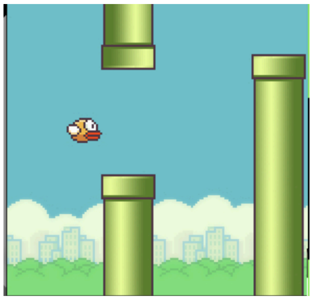
Midterm Assignment RL:

Introduction

In this assignment, you will apply reinforcement learning techniques to solve a Flappy Bird environment implementation here FlappyBirdMidTerm.ipynb. **Read the NoteBook carefully and follow its guidelines!**



Your objective is to develop an agent that can consistently navigate the bird through at least **ten pipes per episode**. The environment's difficulty is controlled by the `gap_size` parameter, which determines the gap between pipes.

Definition of "Solving": The agent consistently passes through at least **10 pipes per episode**.

Assignment Objectives

Your technical work is divided into three main parts: **Preprocessing**, **Reward Shaping**, and **Agent Implementation**. Provide reasoning for every major step in the final report.

Preprocessing

Prepare the environment's observations for training your agent. This may involve transforming, normalizing, or extracting features to enhance learning efficiency. Implement all preprocessing code yourself, using only built-in Python packages or those already imported in the provided Jupyter notebook. Justify each preprocessing step and explain its impact on the agent's performance. Analyze the observation space size with and without your preprocessing method.

Reward Shaping

Design an effective reward function to guide the agent's learning process. Modify or enhance the reward structure of the Flappy Bird environment to better align with your agent's objectives.

Provide detailed reasoning for your reward shaping choices and explain how they improve learning efficiency and overall performance.

Agent Implementation

Implement **at least two reinforcement learning methods** covered in class, such as Q-learning, SARSA, etc. Develop these algorithms from scratch **without using external libraries** for reinforcement learning algorithms. Use only built-in Python packages or those already present in the notebook. We suggest you use higher values of `gap_size` when debugging your code. For each method, explain your implementation choices, parameter settings, and how the method addresses the challenges posed by the Flappy Bird environment. Refer to concepts and implications we discussed in the class such as table size, exploration, discount factor and other related topics.Use only methods and approaches that were covered during the lessons.

For each method you choose to implement, display **the learning progress of one successful experiment**. Include one video for every 10% of the learning process. The videos should be stored in the notebook for convenience. You can use the `embed_mp4` function (available in the notebook. See *Running environment* section) or any other method to display videos in Jupyter (you can import external packages here if needed)

Report Requirements

Submit a **three-page report (not including graphs)** documenting your work. The report should include:

- **Introduction**: Provide a short introduction to your approach for this assignment.
- **Methodology**: Describe your approach in preprocessing, reward shaping, and agent implementation.
- **Results**: Present your experimental outcomes with relevant graphs and observations.
- **Discussion**: Interpret your results, compare the effectiveness of the methods, and reflect on your learning experience.
- **Conclusion**: Summarize your findings and suggest potential future improvements.

Include graphs and visualizations that illustrate the learning process, such as reward over time, score per episode, number of steps per episode, and any other interesting measurements. **All graphs, tables, and visualizations should be included at the end of the report**, starting after page 3. Each graph, table, or visualization must be clearly labeled and enumerated sequentially. For example: "Figure 1", "Figure 2", etc. **Ensure that every graph, table, or visualization is properly referred to in the main body of the report**. For instance: **"As shown in Figure 1**."

Submission Guidelines

Submit your work as a zip file named: `<student_id1>_<student_id2>.zip`

Your submission should include:

- **Notebook**: A Jupyter notebook link containing all the code necessary to reproduce your results, with all cells executed. The code must be well-organized and commented.
- **Evaluation Notebook**: An additional notebook that can load a policy and run it for 10 episodes, evaluate the policy, and display a video of one of the episodes. (Assume we will load your policy file to `/content` before we run using "Run All" in Colab.)
- **Report:** Submit the report in PDF format, adhering to the specified content and formatting requirements.
- **Policy**: Include the policy file or a link to the file. It can be in any format you choose in the evaluation notebook (e.g., NumPy array, pickle file).
- **Info**: An `info.txt` file that includes your names, student IDs, and links to notebooks or policy files if you prefer not to include them in the zip file.

Grading Criteria

- **Code Implementation (25%)**: Correctness and functionality of your code. Effectiveness in solving the environment with `gap_size` in default setting.

- **Advanced Challenge (10%)**: Successfully solving the environment with a `gap_size` equal to **65**.

- **Report Quality (60%)**: Depth of analysis, clarity of explanations, insights into your approach, and quality of graphs and visualizations.

- **Code Style and Documentation (5%)**: Readability of your code and adequacy of comments and documentation.

Support

For questions or clarifications, contact the course staff through the designated communication channels. assafcaf@gmail.com