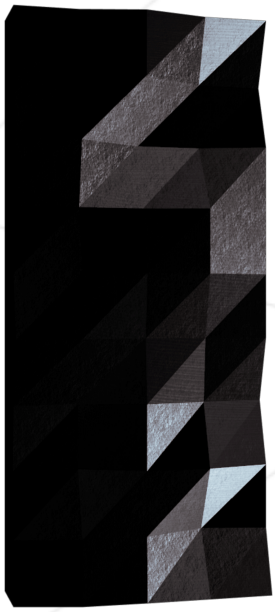

Introduction to Programming - 42

Day 07

Kai kai@42.us.org
Gaetan gaetan@42.us.org

Summary: This document is the subject of the day 06 of the introduction to programming piscine.

Contents

I	**Guidelines**	**2**
II	**Preamble**	**3**
III	**Goals**	**5**
IV	**General Instructions**	**6**
V	**Mandatory Part**	**7**
VI	**Bonus Part**	**8**

Chapter I

Guidelines

- Corrections will take place in the last hour of the day. Each person will correct another person accoring to the peer-corrections model.
- Questions? Ask the neighbor on your right. Next, ask the neighbor on your left.
- Read the examples carefully. The exercises might require things that are not specified in the subject...
- Your reference manual is called Google / "Read the Manual!" / the Internet / ...

Chapter II

Preamble

From the Wikipedia article on Maven (Scrabble):

> Maven is the current best known artificial intelligence Scrabble player, created by Brian Sheppard. It has been used in official licensed Hasbro Scrabble games, and the downloadable Funkitron Scrabble.
>
> Maven's game play is sub-divided into three phases: The "mid-game" phase, the "pre-endgame" phase and the "endgame" phase.
>
> The "mid-game" phase lasts from the beginning of the game up until there are nine or fewer tiles left in the bag. The program uses a rapid algorithm to find all possible plays from the given rack, and then part of the program called the "kibitzer" uses simple heuristics to sort them into rough order of quality. The most promising moves are then evaluated by "simming", in which the program simulates the random drawing of tiles, plays forward a set number of plays, and compares the points spread of the moves' outcomes. By simulating thousands of random drawings, the program can give a very accurate quantitative evaluation of the different plays. (While a Monte Carlo search, Maven does not use Monte Carlo tree search because it evaluates game trees only 2-ply deep, rather than playing out to the end of the game, and does not reallocate rollouts to more promising branches for deeper exploration; in reinforcement learning terminology, the Maven search strategy might be considered "truncated Monte Carlo simulation". A true MCTS strategy is unnecessary because the endgame can be solved. The shallow search is because the Maven author argues[1] that, due to the fast turnover of letters in one's bag, it is typically not useful to look more than 2-ply deep, because if one instead looked, e.g. 4-ply, the variance of rewards will larger and the simulations will take several times longer, while only helping in a few exotic situations: "We maintain that if it requires an extreme situation like CACIQUE to see the value of a four-ply simulation then they are not worth doing." As the board value can be evaluated with very high accuracy in Scrabble, unlike games such as Go, deeper simulations are unlikely to change the initial evaluation.)
>
> The "pre-endgame" phase works in almost the same way as the "mid-game" phase, except that it is designed to attempt to yield a good end-game situation.

```
The "endgame" phase takes over as soon as there are no tiles left in the
bag.  In two-player games, this means that the players can now deduce from
the initial letter distribution the exact tiles on each other's racks.  Maven
uses the B-star search algorithm to analyse the game tree during the endgame
phase.
```

We will not be building a champion level Scrabble AI yet today.

Chapter III

Goals

Text analysis is an important aspect of projects we can accomplish in code: We could do `sentiment analysis` of online reviews, build a `chatbot` to reply to helpdesk requests, or write a `search engine` which indexes pages based on the keywords they include.

We're going to start with something simple: Use your string-skills to challenge your friends' vocabulary through a game of Hangman!

Chapter IV

General Instructions

Ready for a word game?

Save Marvin!

Marvin, the Paranoid Android, is on trial for talking to another computer and making it so depressed that it ceased functioning, leading to the aspyxiation deaths of two Galactic Policeman.

Code a version of Hangman, `save_marvin.rb` that we can play in the terminal. Tomorrow, you'll have the opportunity to turn it into a game that can be played online through a browser window.

Chapter V

Mandatory Part

- Read in a word list from a file or from online.
- Pick a word at random from the vocabulary list.
- Display to the user a set of underscores ("_") indicating how many letters there are in the chosen word.
- Ask the user to guess a letter. If it is a correct guess, fill that in and display the partially completed word. It if is a failed guess, notify the user.
- Print a success message if the player guesses all letters in the word!

Chapter VI

Bonus Part

- Add a limit on the maximum guesses a player can make before losing.
- Vary the game with "easy", "medium", and "hard" modes: Ask the user which difficulty they want at the beginning of the game. Easy mode gives shorter words. Hard mode gives longer words.
- Add a cute Ascii art drawing of Marvin.
- Can you make the game animated by clearing the screen each time and printing a new image in place of the old one?
- Done with hangman? Code another word game: Scrabble Helper. Given a set of letters, suggest all the legal Scrabble words that could be formed by those letters.