Programming Assignment
Kalah Game Analyser

Daniel Brownridge
Bank of England

September 2, 2025

1 Summary

Kalah is a two-player board game in the Mancala family of games. The task for this assignment is to write a program in Python which can analyse a record of the moves made in a game of Kalah. The analyser should validate that a given input file conforms to the provided specification and that the recorded moves are legal according the rules of the game. For valid input the program should output which player won or if the result was a draw along with the final score for each player. The aim of the assignment is to assess general programming skill and problem solving ability.

2 Rules

2.1 Setup

The game of Kalah is played with counters, known as *seeds*, and a board made up of holes, called *houses* and *stores*, which contain the seeds. The houses are aranged into two rows, one belonging to each player, with the player's store located at the right-hand end of their row.

The game starts with the stores empty and a fixed number of seeds in each house. In the standard version of the game there are 6 houses per player and 4 seeds per house, but more generally a varient of Kalah may have m houses per player and n seeds per house.

Here is a representation of the board at the start of a standard game. The positions of the houses relative to the player are labelled starting from 1.

	6	*5*	*4*	*3*	*2*	*1*	
	4	4	4	4	4	4	
0							0
	4	4	4	4	4	4	
	1	*2*	*3*	*4*	*5*	*6*	

2.2 Moves

To make a move a player first chooses one of their houses with seeds and gathers them all into their hand. The seeds are then *sown* by being placed into holes on the board from the players hand. Starting with the hole immediately to the right of the chosen house one seed is sown into each consecutive adjacent hole. Sowing continues anti-clockwise, looping around the board as many times as necessary, including every hole on the board apart from the opponent's store, which is skipped over, until the player's hand is empty.

What happens at the end of a move is determined by where the last seed in the player's hand is sown. One of three things can happen:

1. When the last seed from the player's hand is sown into the player's store then the player may move again by selecting another one of their houses.

2. When the last seed is sown into one of the player's empty houses, and the corresponding opponent's house directly opposite has seeds, then they make a *capture*. In this case, both the last seed sown in their own house, as well as all the seeds from the opposite opponent's house, are collected into the player's store. After making the capture the player's turn ends.

3. In all other cases the player's turn ends after sowing the last seed in their hand.

2.3 Game end

Play continues until either row of houses becomes completely empty (note that either row may become empty on any given turn). At this point all seeds remaining in play are gathered from the houses and returned to their respective owner's store.

The winner is the player who has collected the most seeds to their store, or, in the event both players have collected an equal number of seeds the result is a draw.

3 File format

Kalah games can be recorded in the following minimal plain ASCII text format:

The file describes the varient of Kalah being played in terms of number of houses and seeds per house. The first line of the file consists of two positive integers m and n, separated by whitespace, where m is the number of houses belonging to each player, and n is the number of seeds initially allocated to each house.

The rest of the file records the moves made in the game. By numbering the houses belonging to a player from $[1..m]$ an individual move can be represented by a single integer corresponding to the house selected by the player. Each subsequent line of the file consists of a single integer describing the move. By convention player 1 is the player who makes the first move.

Apart from on the first move, a consequence of this format is that, it is not possible to determine which player's turn it is by looking at an arbitrary line on it's own.

3.1 Simplest example

As an example consider the simplest possible Kalah game, which would not be very interesting to play, consisting of 1 house per player and 1 initial seed per house. Listing 1 shows the file describing the game in full. The file only consists of two lines since only one move is possible.

In the game player one selects house 1 which is sown into their store. This means they would get another go but they can't take it since they don't have any seeds left. So the game is over and player two collects their one seed into their store and the game result is a draw.

Listing 1: 1-1-1-1.txt

```
1 1
1
```

4 Analyser specification

The analyser program should conform to the following specification.

4.1 Language version

The script should be written in Python 3. The script will be tested using the latest stable release of Python 3.12.

4.2 Libraries

Any script may import any part of The Python Standard Library. Using additional third-party packages and libraries is not permitted.

4.3 Source code

The analyser script should consist of a single Python source code file named `kalah.py`.

4.4 Invocation

The script should take a single command-line argument which is the name of Kalah game file to be analysed. The script will be invoked as follows:

```
$ python3.12 kalah.py input.txt
```

4.5 Standard output

When a valid input file for a complete game is provided the analyser should produce a single line of output on STDOUT. The output line should consist of three integer values separated by whitespace in the format:

$$result\ score\ score \tag{1}$$

The first value, *result*, should be 0 for a draw, 1 if the game was won by player one and 2 if the game was won by player two.

The remaining two *score* values indicate the final number of seeds gained by player one and player two respectively.

For example in the game described by listing 1 above, the output should be as follows, since the game was a draw and each player scored one seed each.

```
0 1 1
```

4.6 Error handling

The script should handle error conditions and display the appropriate message to STDERR. There should be no output to STDOUT when an error occurs. In the event a file contains multiple errors only the first identified error should be displayed. The nine specified error conditions are defined below in order of precedence.

4.6.1 No input file

The script expects the path to a game input file to be provided as a command-line argument. If no-command line arguments are supplied then the following error message should be displayed:

```
kalah: error: no input file
```

4.6.2 Too many arguments

The script only expects one command-line argument to be given. If more than one command-line argument is supplied then the following error message should be displayed:

```
kalah: error: too many arguments
```

4.6.3 Could not open file

If exactly one argument is supplied to the script but a file could not be opened for some reason, such as an invalid path or incorrect file permissions, then the following error message should be displayed:

```
kalah: error: could not open file
```

4.6.4 Expected two values in header line

The header (first) line of the file should contain exactly two values separated by whitespace characters. If too few or two many values are found then the following error message should be displayed:

```
kalah: error: expected two values in header line
```

4.6.5 Invalid value in header line

The values in the header line should be positive integers. If a non-positive integer value is found in the header line then the following error message should be displayed:

```
kalah: error: invalid value in header line
```

4.6.6 Expected one value in body line

Each line of the body (rest) of the file should contain exactly one value (ignoring whitespace). If no values or more than one value is found then the following error message should be displayed:

```
kalah: error: expected one value in body line
```

4.6.7 Invalid value in body line

Each value in the body of the game file should represent a single move given as an integer corresponding to position of the house in the row on the board. If the value found is not an integer or the value falls outside the range of possible house positions then the following error message should be displayed:

```
kalah: error: invalid value in body line
```

4.6.8 Invalid move house is empty

A move must involve seeds to be legal. If the selected house is empty then the following error message should be displayed:

```
kalah: error: invalid move house is empty
```

4.6.9 Insufficient moves

A game file should describe a game played to completion so a final score can be determined. If all the moves in the file have been interpreted but the end of the game has not been reached then the following error message should be displayed:

```
kalah: error: insufficient moves
```

5 Submission

To submit your solution create a ZIP archive file of your kalah.py and send it as an email attachment. Do not attach Python source files directly to your reply as it will likely be blocked. There is no need to include any other files with your submission. If you would like to provide any notes on your solution feel free to so by including docstring comments in your source file.

6 Assessment

Submissions will initially be auto-marked against a suite of pre-defined tests. The test cases will assess both logical correctness and performance of the code. Note that some of the test cases may be very large. Submissions that pass automated checks will additionally be qualatively assessed for use of good coding practices.