GAME DESIGN

CHALLENGE 3

Furious Fowl

TIME: 2 hours

Just in time for the 2012 IT Olympics, we have created an interactive Angry Birds-style videogame for you to enjoy. The catch is, in order to play (and win), you have to learn a new programming language, the FuriousFowl Programming Language (FPL).

The gameplay is simple, yet addictive. You are in control of a configurable set of birds and blocks, in an Angry Birds competition against one other team. The FPL syntax allows you to select your blocks, place them in the scene, sling birds around, and shift your scoring target defensively as necessary. Both teams have a fixed slingshot location, and a nearby scoring target, divided into regions as follows:

- If a bird (from either team) hits the blue region of your target, the other team receives 1000 points.
- If a bird (from either team) hits the white region of your target, the other team receives 2000 points.
- If a bird (from either team) hits the red region of your target, the other team receives 5000 points.

You are given the FuriousFowl game executable, as well as a sample FPL file that you can edit. The game can either be run using a single FPL file (for testing your individual team), or with two FPL files (for the competition part). Let's explore the syntax of the sample FPL file:

```
select Black 2
```

allocates two of the Black birds (they explode, see below) to be sent towards the other team's target. Your FPL program should allocate as many birds as you like, given a total cost constraint is map-specific.

```
select Wood 1
```

This allocates a wood plank to be placed defensively in front of your own scoring target (you can also select boxes with `Wood.box`). Both the birds and the blocks contribute to the overall budget, so as part of your FPL program you will have to decide to what extent you want to allocate defensive versus offensive resources.

```
Place 10, 50, 90
```

is a command that places the next block in your team's allocation, specifically at coordinate [10, 50], rotated to a 90 degree angle. The valid coordinate range for user block placement it [0, 0] to [50, 100], with [0, 0] corresponding to the corner closest to your target. Invalid placements will likely lead to blocks that don't behave correctly (and won't be useful for your team's defensive purposes), so experiment with the parameters!

Also, although commands for bird/block allocation and usage can be intermixed, it will likely be easier to debug if you allocate all the birds and blocks in the very beginning of your program, and then use them later in the code.

```
Sling 10, 45, 1000
```

Is an example of the main bird sling command—in this example, we are specifying a sling power (strength) of 10, at an angle of 45 degrees, with a delayed action (if any) occurring 1000 microseconds after the bird begins its flight. Sling power is limited to the range [0-15], with angles from [0-90], and delays of anything from 0 to 10 seconds (10000 microseconds).

The rest of the commands are similar in nature:

```
Wait 1500
```

waits for 1.5 seconds before executing the next command.

```
skip
```

moves your current bird to the end of your list of birds. This command incurs a slight wait penalty as well, to discourage non-stop skipping operations.

```
Shift 50
```

moves your team's scoring target to approximately the vertical midpoint of the map—this can be a vital defensive maneuver and so also incurs a wait penalty. The valid range of shift operations is [0, 100].

Experiment with the FPL syntax on your own for the first half of this challenge. Then, your team will be entered into a round-robin style tournament against the other competitors. You can tweak your team's moveset after every round of the tournament, although once begun the games will go by very quickly!

You can select your blocks from any combination of the following three block classes:

1. **Ice** — these slippery blocks are cheapest of the three block types, it also breaks the easiest. Use these to deter your opponent's first few birds, but don't expect these blocks to survive to the end of the round!

2. **Wood** — these blocks are medium cost, medium strength. These will deter the weaker birds.
3. **Stone** — the most expensive and also the toughest of the blocks. A few well-placed stones will eat up your budget, but will provide some formidable defense!

The selectable bird classes are a subset of the familiar choices from the original Angry Birds game, each with their own benefits and costs to explore:

Name	Size/Weight	Cost	Delayed Action
Red	Small	Lowest	None
Blue	Tiny	High	Splits into three birds
Yellow	Medium	Medium	Increases speed
White	Large	Medium	Drops explosive egg
Black	Large	High	Explodes!

Your task is to build a team that can score the most points in a given round. The basic rules will not change, but some aspects like the structure of the map and the budget constraints may be modified as the tournament progresses.

Finally, we have added an optional feature that allows you to conditionally execute instructions, based on the state of the game. These conditional instructions are predicated with the '@'symbol, for example:

```
@score sling 4, 45, 1
```

Is a conventional sling command, with the added condition that the sling operation will only happen if your team's previous bird scored! There are several conditions that can be checked , including whether your team is `Winning`, `Losing`, `Tied`, or your bird supply is `Empty`, or your most recent bird `Hit` something, `Break` something, or achieved a `Score`. You can also query the color of the current bird (e.g. `White`, `Black`, `Blue`, `Red`, or `Yellow)`.

You can test your opponent's status with these predicate conditions, and can also check for the inverted condition. For example, the following command:

```
@!opp.empty wait 5000
```

Will wait for 5 seconds if your opponent is not currently out of birds.
Please test your .fpl files before submitting, to make sure that they compile correctly.

Most importantly, have fun!