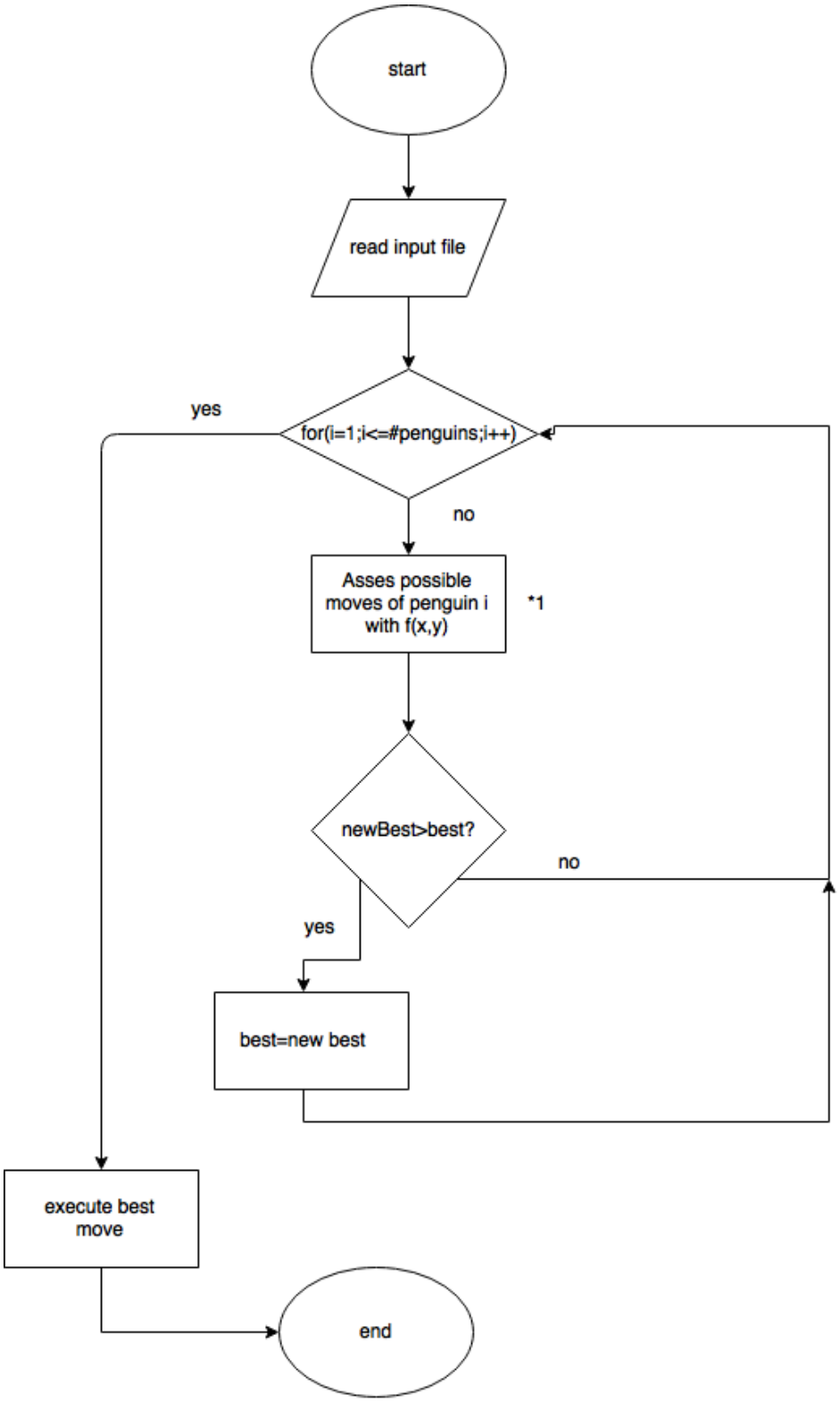
Programing fundamentals, meeting 2, progress summary

Flowchart of algorithm determining next move



best and newBest variables are pairs of cooridnates, determing current location of a penguin, and his location after the move.

*1 This block, will assess every move, that a penguin can make, using a mathematical formula that will take into account factors such as:

1. amount of fish gained directly from the move

2. amount of fields accessible after the move, by all friendly penguins (taking into account amount of fish on those fields, by increasing their “value”)
3. amount of fields accessible after the move, by all enemy penguins (taking into account amount of fish on those fields, by increasing their “value”)
4. current situation on the map (ex. ratio of penguins vs fields)
5. who is winning? (if we are winning, algorithm will focus on disrupting enemy paths to good fields, to keep the advantage. If loosing, it will try to gather as much fish as possible, ignoring the enemy, to catch up)

This formula will be the “heart” of our program. It will allow us to quantify how good a field is, by assigning a value to every possible move. In first stages, only basic factors will be implemented, to test overall concept and code already in place.
In it’s most general form, it can be written as:

$$MoveValue = \frac{1}{A} \cdot f(me) - A \cdot f(oponent)$$

Where $f(player)$ is a function calculating number of fields accessible by a $player$after a move, taking into account, the amount of fishes on them (fields with more fish will be considered much more valuable, while those with only one fish will be regarded as almost not existing), and A is the aggresivnes factor, value of which will be calculated based on situation on map.
After implementing this algorithm, that considers only the current move, we might modify it to look two, three or more moves ahead.

File format

NumbOfPlayers=#;NumbOfPenguins;
[player#]:score:penguin1x:penguin1y:...;
[player#]:score:penguin1x:penguin1y:...;
Map
field1x:field1y:NumberOfFish:Penguin
field2x:field2y:NumberOfFish:Penguin
field3x:field3y:NumberOfFish:Penguin
field4x:field4y:NumberOfFish:Penguin

Proposed file format, would consist of two sections:
First devoted to players, containing:

1. number of players
2. number of penguins per player
3. for every player:
    a. current score
    b. location of their penguins

Second devoted to map, started by header “Map” to make reading the file easier. This section will contain information on fields in form of a list. Each field will be described using folowing variables:

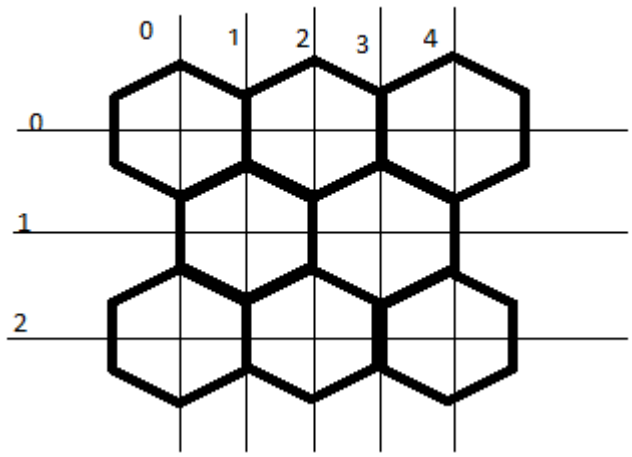
1. field#x - x coordinate of the field
2. field#y - y coordinate of the field
3. NumberOfFish - number of fish on the field

4. Penguin - indicates whose penguin is on the field, number 0 will indicate that there is no penguin there

Example file:

NumbOfPlayers2#;1;
1:0:0:0;
2:0:1:1.;
Map
0:0:1:1
1:1:1:2
2:0:3:0
0:2:3:0

Coordinate system:



This coordinate system sacrifices memory efficiency, for a consistent rules for navigating the map. It utilises only 50% of allocated memory, but rules for moving between different cells are very simple and consistent. Every possible move can be expressed using a multiple of one of six vectors:

A. [-1,-1]
B. [0,-2]
C. [+1-1]
D. [+1,+1]
E. [0,+2]
F. [-1,+1]