Introduction

The goal of this assignment is to get acquainted with the imperative programming language Basic! While it lacks many modern control and data structures, Basic exemplifies the use of imperatives including its `goto` and `gosub`. This assignment is to implement the 1970's classic Animal Guessing Game (check out the links below). In the game you think of an animal and the program tries to guess it. Better yet the program is an AI as it learns as it plays!

The Game

The program starts by guessing that your animal is a lion. If it is not then the program asks for a question that separates your animal from a lion. For example, “Does it have a tail?” This question becomes the first question the program asks the next time you play. Here is a partial transcript (user input is shown in bold):

Think of an Animal! ... I’ll try to guess what it is.

Does it have a tail? Y
Is it a Lion? N
Rats! I want to learn! What was your animal? MOUSE
Thanks! What is a question whose answer is
yes for a MOUSE and
no for a Lion?

IS IT SMALL

Think of an Animal! ... I’ll try to guess what it is.

Does it have a tail? Y
IS IT SMALL? Y
Is it a MOUSE? Y
I rock at this! Let’s play again ;)

Thus the program builds up a decision tree where each internal node is a question and the leaves are the animals that it guesses. Because classic Basic lacks the pointers or reference variables required to create a linked data structure, we will represent the tree as a complete binary tree using an array, with the root being at 0, its left and right children at 1 and 2, respectively, etc.

The Assignment

Implement the Animal Guessing Game in Basic using http://www.calormen.com/jsbasic.

Your program’s first line must be “10 REM This is my code” and its second line your name.

Your solution must include the imperatives `goto`, `gosub`, and `on` `goto`.

Be sure to include good internal documentation using Basic’s `REM`ark “statement.”

What to hand in

(1) Upload the single file `animals.bas` to Moodle by 6:00am on the due date.
Pushes

(1) Some websites I found useful (or at least entertaining)
   (play the game online) http://www.animalgame.com/play/index.php
   (watch Alexa play) https://www.youtube.com/watch?v=tgHLtMcqMs
   (and win!) https://www.youtube.com/watch?v=AfUQPsWqVaM.

(2) The following code illustrates how to allocate an array of numbers (Line 20) and an array of strings (Line 30). (Basic’s use of a ‘$’ to denote a string variable portends later type indicating syntax such as found in the language perl.) The code also illustrates how to initialize these arrays using basic’s read and data statements.

```
10 magic = 100
20 dim QA(magic)
30 dim Q$(magic)
40 read count
50 for i=0 to count-1
  60 read N, QA(i), Q$(i)
  70 rem assert N == i
80 next i
2000 rem node, unused/Question/Answer (0/1/2), String
2005 data 3
2010 data 0, 1, "Does it have a tail"
2020 data 1, 2, "Lion"
2040 data 2, 2, "Monkey"
2080 data -1, -1, "<end marker>"
```

The data statements capture the program’s learned knowledge.

(3) When completed the following code will output replacements for the data statements. To make the animals that your program has learned permanent, cut and paste the output into your code.

```
1200 rem output current data
1210 print "2005 data " + str$(count)
```

Notice that line 1210 print the current count of used array entries. If a new question causes your program to use nodes beyond the current value of count, make sure to increase count, so that the new nodes are used by subsequent searchers and “saved” by the program.

(4) Finally, code to get a single character and to print two blank lines (two linefeeds).

```
10 get A$
20 print chr$(10) + chr$(10)
```