4. *(50 points)*

In this problem, you will find an executable file `trog-win.exe` is an oracle that decides whether a sentence you type bodes Joy or Despair. This program was written by Prof. Andrew Moore. You can invoke the program as follows:

```
Trog-win 1 play

Speak to the oracle> hello there
******* Despair *******
Speak to the oracle> whats your problem
******* Joy *******
Speak to the oracle> fair enough
******* Despair *******
Speak to the oracle> i am perplexed
******* Despair *******
```

There are many oracles. Each has a different oracle number (that's the 1 on the command line above). If you choose a different oracle, one of many possible different rules may be used. Some oracles are easy for a human to work out, while others are very cryptic. Your goal in this assignment is to write a MATLAB program that can predict an oracle's pronouncement given a set of inputs.

What your program should do

To begin with, you have been provided a set of support codes written in MATLAB. And your task is to complete the missing functions, so that the overall program runs in the following way.

The file `main.m` will be the starting point of the whole program. When invoked by pressing F5 in MATLAB, this file first calls a `trog_DataManager` which in turn calls `trog-win.exe` to obtain a collection of training and testing samples from the oracle. The requested data are stored in `trog.dat` and `trog.tst`. To complete this homework assignment, it is not required to study or understand the details of `trog_DataManager.m`.

In `trog.dat`, you will find a set of sentences generated by some oracle. The format of the file is:

<*features1*><*sentence1*> → <*value1*>
<*features2*><*sentence2*>→<*value2*>
.
.
.
<*featuresN*><*sentenceN*> → <*valueN*>

where each `<sentence>` consists of one, two, or three space-separated strings. Each string is 7 characters or less. Each `<value>` is either joy or despair. Each `<features>` is a string of binary digits, giving you information about the attributes if the sentences that the oracle might use in making is Joy/Despair decision.

Here is an example `trog.dat` file:

```
111011000000010       gloves    werent      sent  -> despair
110110110000111      attract      glad     grime  -> joy
111111110000011      insight     names   stained  -> joy
011001000000010        doggy   scooped            -> joy
111111100000001       turned    fuller   vaguely  -> joy
111101100000000       wildly    planet   begging  -> joy
```

Typically, you can expect many more lines. Note that given the 15 binary features, an oracle can perfectly make its Joy/Despair decision. So the actual words in this file can be ignored from a learning perspective.

In `trog.tst` your will find something similar to the above, but without the Joy/Despair decisions:

```
101111101011001         wake     aside   greener
111101000000001        picks     gazes     threw
010000000000001       warmly
101101100000001         west    debris     floor
001011000000000         reek    gaping
011011000000000       dialed   caliber
110010000010011        budge      hums      says
```

Having obtained both training set and test set, `main.m` will instantiate a decision tree (`DecisionTree.m`) and initiate its training algorithm, so as to learn the decision rules from the training set from `trog.dat`, build the decision tree by adding nodes (`DecisionTreeNode.m`), and classify the test set in `trog.tst`. The classifications results will be stored in a new file called `trog.sub` which has same format as `trog.dat`. For example:

```
101111101011001     wake     aside   greener  -> Joy
111101000000001    picks     gazes     threw  -> Despair
010000000000001   warmly                      -> Despair
101101100000001     west    debris     floor  -> Joy
001011000000000     reek    gaping            -> Joy
011011000000000   dialed   caliber            -> Joy
110010000010011    budge      hums      says  -> Despair
```

After `trog.sub` is generated, `trog_DataManager` will submit it to `trog-win.exe`, so as to check the accuracy (or error rate) of your predictions. An output similar to the following will be printed:

```
The oracle says you have 97 out of 100 correct ( 97.0%)
Oracle      #Training Sample     #Test Sample    Error Rate (%)
   2                50               100              3.00
```

Finally, the `DecisionTree` class we provided would finally visualize the structure of the tree in a pop-up figure window. You could take snapshots of the tree structure to include in your report.