**Short Answer**

1. **a. Because there is a boolean stop condition: 5 points**

**Because there is no way to tell how many lines there are in a file beforehand: 5 points**

Correct answer but incorrect info about for loop: +4 points

For iterate more than needed but doesn't address while: 3 points

While loop has no restricted length: 4 points

Read every line, but bad explanation: 2 points

* 1. **Cell arrays can store heterogeneous types of data: 5 points**

**Cell arrays can store multiple strings without them being concatenated together: 5 points**

Award partial credit depending on the accuracy of the answer.

No data conversions: 2 points (i.e. casting doubles to chars)

Incomplete thought but towards correct answer: 2 points

Restricts array abilities (only can store doubles): 4 points

can get data easier (indexing): 3 points

**Short Tracing (20 points)**

Line 2: **fgets(fh1)** or **fgetl(fh1) : 2 points**

Incorrect input into fgets or fgetl: -1 point

Misspelling: -1 point

fget instead of fgets or fgetl: -1 point

Line 8: **(rest)** or **(rest, ‘ ‘) : 3 points**

Correct variable: 2 points

        Correct delimiter: 1 point

**rest(2:end): 3 points**

Line 10: **line = fgets(fh1)** or **line = fgetl(fh1) : 2 points**

Incorrect input into fgets or fgetl: -1 point (iff not wrong in line 2.  No double jeopardy--student earns 2 points if their answer on line 10 is identical to that on line 2, provided something is written in each blank)

Does not assign to 'line': -1 point

or assigns to different variable name: -1 point

    Line 12: **fclose(fh1) : 3 points**

**fclose(all) or fclose ‘all’ : 3 points (full credit)**

Incorrect input into fclose: -1 point (iff not wrong in line 2 & 10.  No double jeopardy--if they used filename instead of file handle, only deduct points once)

        Do not take points off for setting the line of code equal to a variable

6a) **When you run out of words, you get an empty string. But when you reach the end of the file, you get a -1, so it would not end the loop. (5 points)**

**Correctly identifies empty strings for words: +2 points**

**Correctly identifies reaching the end of the file does not return an empty value: +3 points**

    Answers that do not address both parts of the question do not earn full credit.

    Doesn't identify that in line 4 it ends in an infinite loop: +2 points

If fgetl() is used, "the loop might exit early upon an empty line" is a VALID answer for why isempty() can't be used on line 4, award full credit.

6b) **The while loop would never count the last word on a line, so the count would be wrong. (5 points)**

**Correctly identifies that count will be less than it should be: +2 points**

**Correctly identifies that the last word will not be counted on each line: +3 points**

    Correctly identifying that the count will be “off by one” in some fashion: 3 points minimum

    Suggests that the code will result in an infinite loop: +0 points

    Not count first word: -2 points

**Long Tracing**

1. **On line 4, isempty(board(a,b)) is always false because it isn't checking inside the cell, just the cell itself.**

**line 4: if isempty(board{a,b})** **(6 points)**

The only thing required for full credit is the correct line of code.

**Providing a line number (ANY line number) of code that is changed: +1 point**

**Changing line 4: +2 points**

**Correct fix: +3 points**

-Correctly identifying that the issue is with indexing using {}s vs. ()s, but changes the wrong punctuation/wrong line: +⅗ points minimum on last two criteria (earns one point for marking a line number)

-Providing a *completely* valid explanation for what’s causing the error, but not fixing the code correctly: +⅗ points minimum on last two criteria

-Changing line 4 to be ~isempty or a completely different logical condition: +⅖ points on last two criteria

1. **The X in the first row, third column is accessed first. This is because the inner for loop indexes through the columns of each row, before the outer for loop moves on to the next row.**

+2 points for correct answer

+3 points for an explanation that invokes the order in which the loops are nested (need not earn the other two points to earn this credit)

1. **[1; 2; 3; 6; 8]: +4 points**

**[1 2 3 6 8]: +4 points**

[1 4 6 7 8] or [1;4;6;7;8]: +3/4 points (indexing horizontally rather than vertically)

+1 for explanation, no numerical answer

Vector of doubles: minimum +2/4 points

Arguing for the code causes an error: maximum 2/4 points, depending on how convincing the explanation is.

Forgetting []s around a vector: -1 point automatically

1. **15 points:**

possMoves = 0;

for i = spaces

    winLog = gameCheck(board,i,move);

    if winLog

        possMoves = possMoves + 1;

    end

end

**+5 points for correctly iterating through the spaces vector**

    Iterating through board, rather than spaces, without compensating for mistake: deduct 3 points

**+5 points for initializing and successfully executing a counter**

Not initializing possMoves: deduct 2 points

**+5 points for calling the gameCheck function using the proper input and output arguments**

Using the word “function” in the function call: -2 points

    Incorrect inputs and outputs/incorrect order: -2 points for each mistake

Can be newBoard or board for full credit.

No iteration: +7 points maximum for the entire problem (can earn full credit for using gameCheck function correctly, though)

Syntax errors: deduct 1-3 points depending on severity, for each different syntax error, tabulated over entire problem

Issues with variable names (not using board, not setting output to possMoves, etc.): -1 point for each variable

Hardcoding for the particular value/length stored in spaces for this example: +10 points maximum for the entire problem

*Take into account the answer for part c) for how the student uses the spaces variable and do not deduct double jeopardy if the coding solution is valid, depending on what the student believes is stored in spaces.*

**Long Coding (40 points)**

1. **Function header (2 points)**

no partial credit

1. **Opens xls file: (5 points)**
   * 1. Using fopen, deduct 3 points
     2. Incorrect output order: deduct 2 points
     3. Using the test case as the filename (but only doing this once): -3 points

Hardcode filename: deduct 2 points

1. **Iterates through correct data in file: (8 points)**
   * 1. Takes into account number of rows in file (uses size function): Award 3 points minimum
     2. Iterate through first row of raw: (deduct 3 points)
     3. Infinite loop: (deduct 4 points)
     4. Does not use iteration: Deduct maximum points here, then award credit for the rest of the problem based on how close the solution could get to the required criteria neglecting iteration. (If no counter is used, then deduct points for d)iii as well.)
2. **Calculates the minimum value on each row and tabulates index (15 points)**
   * 1. **Calculates minimum on each row (4 points)**
        1. Uses raw and does not convert to doubles (deduct 2 points)
        2. Includes “angle” column when calculating minima of each row (deduct 3 points)
        3. If they use the min function, they get 2 points here automatically.
        4. Do not deduct points if iterates through wrong data (see criterion c.)
        5. If there is no iteration, award full credit here if they take the minimum of the entire numerical array.

Don't deduct points for absolute value for magnitude.

* + 1. **Makes use of/stores the column index of the minimum on each row (3 points)**
    2. **Implements a counter to keep track of “minimum” normal forces (8 points). For example, a vector of four zeros that will be updated with each column index.**
       1. Does not initialize counter: deduct 2 points
       2. Incorrect scheme for accounting for the counter: award partial credit as appropriate

1. **Finds column with “maximum minimum” normal forces. (5 points)**
   * 1. Uses max function on appropriate vector: +2 points
     2. Store the column/index: +1 point. Do not deduct twice for syntax errors if this was done incorrectly in d)ii.
2. **Outputs correct direction as a string: (5 points)**
   * 1. Outputs the title as a cell array: (deduct 2 points)
     2. Assumes that the order of the titles will always be the same as in the test case: (deduct 3 points)
     3. Account for the fact that the “first” column of num and raw is always offset from the index of the maximum (i.e., only index the top row starting at the second column to avoid indexing the word “Angle”): Deduct 2 points if they do not do this.
     4. Does not assign output of function: (deduct 2 points)

General syntax errors should incur 1-3 points for each different error, depending on the severity. Errors using {} vs () will always lose 2 points.

Any solution which directly hardcodes from the test case and does not account for an unknown number of rows can earn 20 points maximum.