Week 5 Lab. Session:
Files and Exceptions

1. Following last week's exercise you should have a file called `marklist.py`. Create a new `python` file to test your module and enter:

   ```
   import marklist
   help(marklist)
   help(marklist.add_module)
   ```

   This demonstrates the purpose of the documentation strings (i.e. the triple quoted strings at the top of the file and the top of every function). If the import statement causes any output on your screen then you need to make sure that all the executable statements are protected by the following `if` statement:

   ```
   if __name__=="__main__":
   ```

   (see last week's lab sheet).

2. Thoroughly work through section 2.5-2.6 of the notes, playing around with any of the code snippets that don't seem trivial to you.

3. Use a text editor to create a file containing the following text or something similar:

   ```
   COF180;10;40
   COF181;10;90
   PHF110;15;73
   PHF210;15;81
   ```

   This is supposed to be a list of modules together with the number of credits and the mark obtained, i.e. the same data that you were dealing with last week. Start by writing a program to read the first line and convert it to a list of strings. Use a simple print statement to print the list; the result should be:

   ```
   ['COF180', '10', '40']
   ```

   Now add some lines to extract the values in the list into three variables `module`, `credits` and `mark`, so that `module` contains 'COF180' (a string), `credits` contains 10 (an int) and `mark` contains 40 (an int).

   Insert the code you have written at the bottom of the `marklist.py` program that you wrote last week; below it, insert a call to `add_module` to add the module to `marklist`; check that the `marklist` has changed as required.

   Finally, modify `marklist.py` so that it processes every line in the file in the same way, i.e. reads it in, decodes it and adds the data to `marklist`; write it so that no matter how many lines the file contains, they are all treated in the same way.

4. Take the code you have written and use it to create the following function in `marklist.py`. Make sure it works as required.

```
def read_modules( filename ):
    """
    Opens a file in which line is formatted as:
       module_code;credits;mark
    e.g.
       COF180;10;73
    Each line is read and the data added to marklist.
    Finally the file is closed.
    Parameters:
       filename = the file to open (string)
    """
```

5. Add the following function to `marklist.py` and test it:

```
def write_modules( filename ):
    """
    Opens a file for writing and writes out all the
    data contained in marklist in the format:
       module_code;credits;mark
    e.g.
       COF180;10;73
       COF181;10;81
    Finally the file is closed.
    Parameters:
       filename = the file to open (string)
    """
```

   Check that a file that is written by `write_modules` can be read by `read_modules`

6. Add the following function to `marklist.py` and test it:

```
def reset_marklist( ):
    """
    Set marklist to be an empty list.
    """
```

7. Write a program called `mark_editor.py` which provides the user with a menu interface to the functions contained in `marklist.py`. The program should import marklist.py and the menu should include the following options:

   - Print out the current mark list
   - Add marks from file
   - Add a mark from the keyboard
   - Write marks to file
   - Empty the mark list

   The "guess my number game" program in section 1.8.2 shows how to display a menu.