CS2321 Lab 11

Lab Instructions:
Save the code you write for each exercise in this lab as a *library* -- that is, a textfile with a .py extension containing only executable python code (i.e. no angle-bracket prompts, etc). Name each file according to the exercise number (e.g. ex1.py, ex2.py, etc.) and save them to a directory containing the report file (in PDF), when completed, compress them together in a single zip file to be submitted on D2L.

Each function should have a docstring explaining what the function does.
Any Follow-up Questions and their Answers should be included in a **docstring** following the `main()` function.

e.g. the structure for a Python module should look like:

```
'''
   modulename.py
   Doc-string explaining what this module does
'''
# imports, such as math, random, etc., as needed

# Your code, includes definitions of classes, functions, etc.
def ...
def ...
.
.
.
def main():
   # Do what is needed.


if __name__ == "__main__":
      main()

'''
   Doc-string answering follow up questions
'''
```

Lab Deliverable: Once all your programs run correctly, collect their code and the results of their test-cases in a nicely-formatted **PDF** file exported from Word Processing document (e.g. MS Word or LibreOffice) to be included in the submission on D2L.

This **report** should consist of each lab exercise, clearly **labeled** in order, consisting of code, then copy/pasted text output, or, for GUI, screen-captured, of its four test-cases.
In this lab, take series of screen captures of your turtle graphics and insert them into the report.

Exercise

All recursive algorithms must obey three important laws:

1. A recursive algorithm must have a base case.
2. A recursive algorithm must change its state and move toward the base case.
3. A recursive algorithm must call itself, recursively.

Recursions with multiple recursive calls are normally very inefficient. However, by recording known results, they can be made pretty fast. This methodology is called memorization or caching. Please review page 178 of the second textbook.

Here is the project for this lab:

Professor Plums likes recursion, but his students typically find it confusing. During a recent faculty meeting his mind wandered, and he invented the following recursive mathematical function, H(n):

- H(n) = H(n+4) + H(n+2) for all values of n < -5
- H(n) = n * 2 for all values of -5 <= n < 2
- H(n) = H(n-8) – H(n-4) + H(n-3) for all values of n >= 2.

He wants you to write a program to compute values of the function H(n).

Once H(n) is implemented, please time it with the following n values, then plot a graph.
The n values to be tested are created from rangee(-1000, 1001, 10):

Hint: A direct implementation is not acceptable because of the multiple recursive calls from each recursion generate too many additional calls. It will take a long time for n values of 50 or larger.