`program.py` issues and suggestions

Originally developed by Mike Jackson

EPCC, The University of Edinburgh

michaelj@epcc.ed.ac.uk

With acknowledgement to Greg Wilson (Software Carpentry), James Hetherington (University College London), Andrew Turner (University of Edinburgh) and EPCC MSc students of 2016.

7 December 2017

1 Readability and comprehension

- Python already enforces good indentation. This may not be the case for other languages
- Add blank lines between blocks and between functions to make them stand out.
- Use self-documenting function and variable names.
- Replace `sum = sum >>` 1 with the less cryptic `sum = sum / 2` or `sum /= 2`.
- Replace `if (sum % 2)` with the less cryptic `if (sum % 2 != 0)`.
- Don't reuse `x` – it is currently used to hold the current power when calculating powers, and is then reused to print the values in the loop.
- Don't name variables after Python's built-in functions e.g. `sum` as this causes the built-in function to cease to work.
- Conform to Python PEP 008 and other Python style guides.

2 Documentation and comments

- Remove misleading comments.
- Remove redundant comments which describe things that are clear from the code.
- Provide comments to explain why the code is doing what it is doing, if this is unclear.
- Add program, function and variable comments describing their role, inputs, outputs, return values etc.
- Add information on author, date, version, licence etc.
- Add comments in Doxygen / docstring form.

3 Structure and design

- Use the built-in Python `bin` function to convert the input number to binary, implicitly getting the powers of two. This would make doing the output more complex however.
- Replace `ps.insert(0, x)` with `ps.append(x)` if the order of powers is not important.
- Remove the hard-coded `2` from the function and pass this in as an argument to the function, and turn the function into a function that calculates the sum of powers of any value, not just 2.
- Make the code both Python 2 and 3 compliant by replacing `print x` with `print(x)`.`print x` is not compliant with Python 3.
- Make the function more cohesive, decouple output from computation, by pulling out the print loop into a new function, which is called from the main block.
- Make the code more cohesive by pulling the contents of the main block out into a new function called from the main block.
- Make the code more modular by putting the functions into a separate file and import these into the file with the main block.
- The "shebang", `#!/usr/bin/python`, has a hard-coded path which may not be consistent with certain systems. Using `#!/usr/bin/env python` is more portable.

4 Usability

- Improve output e.g. print powers as `30 = 16 + 8 + 4 + 2`.
- Add validation of inputs to the program or function. These currently raise errors if a sequence of letters is input, or a negative integer is input.
- Provide human-readable error messages, not stack traces when errors are detected.

5 Miscellaneous

- There is no need for automated build as it is Python.
- A complementary unit test suite would help convince users and developers of the correctness of the code.