Concepts

- **Numerical Expressions**
  - numerical types (`int`, `float`)
  - numerical values and notation (e.g. `7` vs. `7.0`)
  - numerical operators (`+`, `-`, `*`, `/`, `%`, `**`)
  - numerical comparisons (`<`, `>`, `<=`, `>=`, `==`, `!=`)

- **Booleans**
  - boolean type (`bool`)
  - boolean values and notation (`True`, `False`)
  - boolean operators (`and`, `or`, `not`)

- **Variables**
  - assignment (e.g. `x = 3 * 4`)
  - reassignment (e.g. `x = x + 2`)
  - augmented assignment (e.g. `x += 2`)

- **Functions**
  - type coercion functions (`int`, `float`, `bool`, `str`)
  - the `print` and `input` functions for interacting with the user.
  - function arguments and calling syntax
  - function parameters and defining syntax
  - semantic indentation
  - function signatures
  - variable tables
  - local vs. global variables
  - `return` statements
  - `returning` vs. `printing` a value

- **Flow Control**
  - `if` statements (`if`, `elif` and `else` clauses)
  - `while` loops
  - `for` loops
- **Strings**
  - string type (**str**)
  - string values and notation (e.g. ‘hello’ and “hi”)
  - string operators: concatenation (+), repetition (*) and substring (in)
  - string indexing (e.g. ‘hello’[2])
  - string slicing (e.g. ‘hello’[1:-1])
  - string methods (e.g. `lower`, `isalpha`)
  - string comparisons (<, >, etc.)

**Skills**

- Writing Python expressions and statements based on an English description.
- Writing short Python functions based on an English description.
- Reading a short excerpt of a Python program and determining the value and any effects resulting from running it.
- Understanding the role of parameter variables and return statements in a function definition.
- Distinguishing between a value returned by a function and information displayed to the screen by it using `print`.
- Distinguishing between an argument to a function and input obtained from the keyboard by it using `input`.
- Understanding how to use a boolean expression as a branch or loop guard.
- Writing syntactically correct conditionals and loops.
- Writing `while` and `for` loops over the characters in a string.
- Writing a boolean valued function (i.e. “predicate”) for validating user input.
- Understanding method syntax for calling functions (e.g. ‘Hi!’.`lower()`)

**Sources**

Material covered by the midterm exam will be drawn from the course lectures, lab activities, homework assignments and readings. In particular, you may want to consult:

- the lecture slides (http://emorehouse.web.wesleyan.edu/teaching/2017/spring/comp112/)
- the lab activity descriptions (see the Moodle page for your section)
- the homework exercises and solutions (see the Moodle page for your section)
- the textbook (http://greenteapress.com/wp/think-python-2e/ with specific readings listed on the slides for each lecture)