CMPUT 174 Midterm Review

1. Identify the following tokens in the given python code.
   a. Constant
   b. Function
   c. Method
   d. Variable
   e. Parameter
   f. Argument

```
def helloWorld():
    print("Hello world!")

def helloPerson(name):
    print("Hello "+name+"!")

def returnPerson(name):
    return name.capitalize()

max_num = 4

for i in range(max_num):
    if i == 0:
        helloWorld()
    elif i%2 == 1:
        helloPerson("WORLD")
    else:
        print(returnPerson("true"))
    print i
print(helloWorld())
print(helloPerson(returnPerson("world"))
```

2. What is output by the program above?

3. True or False. If False, change to be true. If ambiguous, clarify.
   a. A formal language requires context to be correctly interpreted.
   b. All statements in any language that are syntactically correct have semantic meaning.
   c. Python is an interpreted language.
   d. Python can be compiled.
   e. The argument to a function is always known at runtime.
   f. The parameter of a function is known at runtime.
   g. An iterative function calls itself.
   h. An expression always evaluates to true or false.

i. A statement returns a value.
j. A statement performs a task.
k. A statement is a sequence of interpretable commands.
l. A semantic error causes an exception.
m. A syntax error results from bad input.
n. A runtime error can occur in a program that the interpreter has completed parsing.
o. Once the interpreter reports no syntax errors and a program has been successfully run, it is correct.

4. Computing terminology
   a. ___ is the process of writing instructions for the computer.
   b. The ___ is responsible for controlling access to memory.
   c. The ___ stores instructions and data.
   d. A(n) ___ is a sequence of well-defined steps for solving a problem.
   e. A(n) ___ is a sequence of well-defined steps written in a formal language.

5. Programming languages
   a. A(n) ___ language manipulates memory directly and is specific to a processor type.
   b. A(n) ___ language must be processed before it can be run.
   c. A(n) ___ language can have statements with more than one meaning.
   d. A(n) ___ language can be executed immediately by the computer
   e. A(n) ___ language can interweave reading and execution of the program code.
   f. A(n) ___ language must be processed entirely into machine code before being run.
   g. The ___ of a language describes the rules that determine what makes a valid statement.
   h. The ___ of a language describes what a statement means.

6. Programming terminology
   a. The ____ frees up memory that is no longer needed by the program.
   b. The ____ converts programmer-readable code into machine-readable code.
   c. The ____ is a program that reads code written in a particular language and executes the statements.

7. Which of the are statements? Expressions? Which evaluate to true?
   a. 4 = 4
   b. 4 is 4.0
   c. 4 == 4.0
   d. 4 < 4.0
   e. 4 <= 4.0

f. 4 == 12/3
g. 4 is 12/3
h. four = 4
i. four = 5-1

8. What type results from each expression?

z = 12
y=7
x=3

a. z%y = ?
b. z/y = ?
c. z//y = ?
d. z%x = ?
e. z/x = ?
f. z//x = ?

9. What structures can you use in python to repeat an identical block of code without having to retype it?

10. I have written a function called "tutorFunction" that takes one integer argument and returns a string. Have I given you enough information to use this function? Why or why not?

CMPUT 174 Midterm Review - Answer Key

1. Program anatomy
   a. `max_num`
   b. User-defined functions: `helloWorld()`, `helloPerson()`, `returnPerson()`. Built-in functions: `print()`, `range()`
   c. `capitalize()` is a method of the String object `name`
   d. `i`
   e. `name` is the parameter of `helloPerson` and `helloWorld`
   f. `max_num` is the argument passed to `range`. The string literals "WORLD" and "true" are arguments passed to the user-defined functions.

2. Output:
   Hello world!
   0
   Hello WORLD!
   1
   True
   2
   Hello WORLD!
   3
   Hello world!
   None
   Hello World!
   None

3. True/False questions
   a. False, a formal language is by definition unambiguous. A natural language would require context.
   b. False, all statements in a formal language that are syntactically correct have semantic meaning. Natural languages can have syntactically correct nonsense.
   c. T
   d. T, although it is interpreted by default it can be compiled into byte code
   e. F, an argument is often not known at runtime, although the code from the first question has an example with a constant argument.
   f. T, the parameter is part of the function definition.
   g. F, a recursive function calls itself (although a recursive function may iterate)
   h. F, an expression returns a value. A Boolean expression returns True or False

i. F, a statement does not return a value, although it may have side effects (eg. `print("Hello")`)
j. T
k. F, a statement is the smallest executable unit of code. A sequence would be a program, function, block, etc...
l. F, a runtime error causes an exception. A semantic error is when the program executes correctly as far as the computer is concerned (i.e. can be parsed correctly and has valid input) but "does the wrong thing".
m. F, a syntax error is a problem in the code structure. A runtime error can be caused by bad input.
n. T, a runtime error is not caught until the program is executed. The interpreter can only catch syntax errors.
o. F, the program may still have runtime errors with inappropriate or unexpected input and may have errors in its logic or operation (hence the need for testing).

4. Computing terminology
   a. Programming
   b. CPU (Central Processing Unit)
   c. memory (could also say registers)
   d. algorithm
   e. program (or function, or machine code)

5. Programming languages
   a. low-level (also assembly)
   b. high-level (also compiled or interpreted)
   c. natural
   d. low-level or assembly
   e. interpreted
   f. compiled
   g. syntax
   h. semantics

6. Programming terminology
   a. garbage collector
   b. compiler
   c. interpreter

7. Statements and expressions
   a. statement (and syntax error, literals can not be used as variable names)
   b. expression, False ("is" looks at type as well as value)

c. expression, True (the equality operator compares values)
d. expression, False
e. expression, True
f. expression, True
g. expression, False
h. statement
i. statement and expression

8. Types
a. z%y --> <integer>
b. z/y --> <float>
c. z//y --> <integer>
d. z%x --> <integer>
e. z/x --> <float>
f. z//x --> <integer>

9. Repeating code can be avoided by using a function (or module), a for loop, or a while loop

10. Syntactically, yes. Semantically, no. Since I have told you the function name, argument, and return value you know how it can be used in a program. However, since you do not know what it does (especially if it has any side effects or requirements from the global namespace) you can probably not use it correctly.