CSE 307 - Principles of Programming Languages
Homework Assignment 01
Python Basics and Parsing with Python

Assigned: 02/06/2020
Due: 02/14/2020, at 11:59 PM
Total Points: 20

Solve the following problems in Python. Each problem is worth five (5) points each. You may use regular expressions in your solutions to the problems.

Problem 1:
Write a program that reads a line via input() and determines if the input is a legal identifier. The program should print only one output: True or False, and then exit.

A Python identifier starts with a letter (A - Z, or a - z) or an underscore (_), followed by zero or more letters, underscores, or digits (0 - 9). Legal Python identifiers cannot be Python keywords. Python keywords are reserved and so cannot be used as constants, variables, function names, class names, or as any other identifier in Python.

Note: Starting an identifier with a single leading underscore indicates that the identifier is private. Starting an identifier with two leading underscores indicates a strongly private identifier. If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.

You are not permitted to use the following library methods for this problem:
    -keyword.iskeyword()
    -str.isidentifier()

The point of this problem is to work out how one would implement such a method by manually parsing the input.

You can easily view a list of all Python keywords by using the built-in "help" function.

```
Python 3.5.2 (default, Nov 12 2018, 13:43:14)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> help()

Welcome to Python 3.5's help utility!

If this is your first time using Python, you should definitely check out
the tutorial on the Internet at http://docs.python.org/3.5/tutorial/.

Enter the name of any module, keyword, or topic to get help on writing
Python programs and using Python modules.  To quit this help utility and
return to the interpreter, just type "quit".

To get a list of available modules, keywords, symbols, or topics, type
"modules", "keywords", "symbols", or "topics".  Each module also comes
with a one-line summary of what it does; to list the modules whose name
or summary contain a given string such as "spam", type "modules spam".

help> keywords

Here is a list of the Python keywords.  Enter any keyword to get more help.

False               def                 if                  raise
None                del                 import              return
True                elif                in                  try
and                 else                is                  while
as                  except              lambda              with
assert              finally             nonlocal            yield
break               for                 not
```

```
class               from                or
continue            global              pass

help>
```

Problem 2:
Write a program that reads a line via input() and determines if it is a legal number. The program should print only one output: int, float, or None and then exit.

Your program must manually parse the inputs. Do not merely pass the strings to the type conversion functions and catch the exceptions.

In Python 3, there is effectively no limit to how long an integer value can be. Of course, it is constrained by the amount of memory your system has, as are all things , but beyond that an interger can be as long as your need it to be. The following strings can be prepended to an integer value to indicate a base other than 10:

    0b (zero + lowercase letter 'b') Base 2 (binary)
    0B (zero + uppercase letter 'B') Base 2 (binary)

    0o (zero + lowercase letter 'o') Base 8 (octal)
    0O (zero + uppercase letter 'O') Base 8 (octal)

    0x (zero + lowercase letter 'x') Base 16 (hexadecimal)
    oX (zero + uppercase letter 'X') Base 16 (hexadecimal)

Values with these prefixes are legal numbers (integers) in Python.

The float type in Python designates a floating-point number. Float values are specified with a decimal point.

https://docs.python.org/3/tutorial/floatingpoint.html

Optionally, the character 'e' or 'E' followed by a positive or negative integer may be appended to specify scientific notation.

    Example: 4.2e-4 = 0.00042

Float values are 64-bit "double-precision" values, according to IEEE 754 standard (https://en.wikipedia.org/wiki/IEEE_754). In that case, the maximum value a floating-point number can have is approximately 1.8 x 10^308. Python will indicate a number greater than that by the string "inf".

The closest a nonzero number can be to zero is approximately 5.0 x 10^-324. Anything closer to zero than that is effectively zero.

Floating point numbers are represented internally as binary (base-2) fractions. Most decimal fractions cannot be represented exactly as binary fractions, so in most cases the internal representation of a floating-point number is an approximation of the actual value. In practice, the difference between the actual value and the represented value is very small and should not usually cause significant problems.

Problem 3:

3. Write a program that reads two lines (two sequential calls to input()) and then determines if the combined result is a legal string enclosed within single or double quotes. The program should print only one output: True or False, and then exit.

The following is a table of escape sequences which cause Python to suppress the usual special interpretation of a character in a string:

    \'   Literal single quote (') character

```
    \"          Literal double quote (") character

    \newline    Terminates input line. Newline is ignored

    \\          Literal backslash (\) character

    To break up a string over more than one line, include a backslash before
    each newline, and the newlines will be ignored:
        >>> 'a\
        ... bc'
        'abc'

    Equiavlently for the assignment:
        first input: 'ab\
        second input: c'
    The result is the valid string: 'abc', and your program should output True.
```

Problem 4:
Write a program that reads a date in the format: MM/DD/YYYY, from the command line and print it out as: Day, Month DD, Year. If the input is formatted incorrectly, or if the input does not correspond to a valid date, then your program should ouput: None.

Your program should first manually check that the input is formatted correctly.

You may only use methods from the datetime library to determine whether the input is a valid date, and then produce the output string.

For example we read "02/07/2019' and output "Thursday, Feburary 7, 2019".

Submission Instructions:
1. Please write each program in a separate file.

2. The file name should correspond the problem for which your program is written.
    Example: Your program for problem 1 should be called "hw1p1.py".

3. Please collect and submit your program files as a compressed zip file.

4. The title of the compressed file should be:
    cse307_hw01_LastNameFirstName

5. This is an individual assignment. Any collaboration on writing your programs will be treated as a violation of academic integrity.