COL100 Assignment 5

Due date: 16 February 2022

Our task is to build an interpreter for a simple Python program with a limited syntax, simulating the computer's execution. Let us start by interpreting an input text file consisting of a sequence of statements, each statement on a separate line. The informal syntax of a statement is indicated below.

Informal Grammar

A STATEMENT is of the form: VARIABLE = EXPRESSION

EXPRESSION is one of:

- TERM
- UNARY_OPERATOR TERM
- TERM BINARY_OPERATOR TERM

BINARY_OPERATOR is one of: $+, -, *, /, >, <, >=, <=, ==$, !=, and, or
UNARY_OPERATOR is one of: $-$, not
TERM is one of: VARIABLE, INTEGER_CONSTANT, True, False
VARIABLE is a sequence of one or more letters
INTEGER_CONSTANT is a sequence of one or more numeric characters ('0' to '9')

Interpreting the Input Program

Read the input file one statement at a time and INTERPRET ("execute") the statement by maintaining all the variables and values encountered so far in a list called DATA.

Example 1

Here is a possible list of actions taken by the interpreter for the input program:

```
x = 1 + 3
y = 4
x = 5
```

- On reading "`x = 1 + 3`":
  - insert elements 1 and 3 into the DATA list.
  - Add $1 + 3$ to get 4. Insert 4 into DATA. Let this be stored in list position i (that is, DATA[i] contains 4).
  - insert element (x, i) into DATA. This is a way to implement "reference" (x refers to object 4).
- On reading "`y = 4`":
  - search for 4 in the DATA list. Locate it in position i (DATA[i] already contains 4).
  - insert list element (y, i) into DATA.
- On reading "`x = 5`":
  - search for 5 in DATA. Since it is not present, insert new list element 5. Let this be in list position j (that is, DATA[j] contains 5).
  - search for x in DATA. Replace (x, i) by (x, j). x now refers to 5.

Example 2

Here is a possible list of actions taken by the interpreter for the input program:

```
q = 6
p = q + 5
r = p
```

- On reading "`q = 6`":
  - insert element 6 into DATA. Let this be stored in list position i (DATA[i] contains 6).
  - insert element (q, i) into DATA.
- On reading "`p = q + 5`":
  - search for 5 in DATA. Since it is not present, insert new list element 5. Let this be in list position j (that is, DATA[j] contains 5).
  - search for q in DATA. Since (q, i) is present, follow the reference i and read the value 6 from DATA[i].
  - Add 5 and 6 to obtain 11. Search for 11 in DATA. Since it is not present, insert new element 11. Let this be stored in list position k (that is, DATA[k] contains 11).

  - search for p in DATA. Since it not present, insert list element (p, k) into DATA.
- On reading "`r = p`":
  - search for p in DATA. You will find (p, k).
  - search for r in DATA. Since it is not present, insert (r, k) into DATA.

Output

When the program completes, print out:

- The name and current value of all the variables used in the input program.
- The list of GARBAGE integer objects used in the program but not referred to any more by any variable at the end of the program.

Notes

- The above language supports two types of variables: integer and boolean. All variables are scalar (no aggregate types such as list).
- If there is any error in the input, print the error and terminate the program.
- Use the Python **open** and **split** functions to help in reading the input file. Assume that spaces separate the different syntactic "tokens" in each line (VARIABLE, TERM, =, +, etc.)