Assignment I (Answer to the Sample Program)

August 2023

1 Output Format

Consider a given example language at figure 1. You need to identify different tokens in the given input program and indicate their respective categories in the output file. The various categories of potential tokens are { `labels, ID, reserved words, punctuation, data types, operators, constants (integer literals, string literals), and special symbols` }.

- `labels` are the statement numbers in the input program. For example, `pp1` needs to be identified as a `label` (`pp` and `1` are **not** two different tokens).
- `ID`s consists of variable and function names. In the given example, `foo` and `c1` are examples of valid `ID`s.
- `reserved keyword`s are the keywords that have special meaning in the program and can not be used directly as a variable name. In figure 1, `gt, and, gteq, otherwise` are the examples of valid `reserved keyword`s.
- `punctuation` marks are used for various purposes, including syntax and code structure. The punctuation marks used include:
  - comma(,) separates function arguments and variable declarations.
  - Quotation Marks (" "): Used to define string literals.
  - Semicolon (;): Used to terminate statements.
  - Colon (:): Used in labeling a program statement.

  Note that parentheses and square brackets are **not** considered as punctuation.
- `special symbols` are considered to be those tokens that can not fall under any other category of tokens. For example, square brackets ('[',']') and parentheses ('(',')') are examples of `special symbols`.

We now provide an output format for the given example language. Though we have provided the output in three different columns, you can print the same in a single column with each class and the corresponding token in the new line. You should follow the format strictly, as your files may be checked automatically.

```
pp1: null foo()
pp2: [
pp3:    integer_2 x1, x2, 1x;
pp4:    character_1 c1, 1c, _c;
pp5:    string s = "Hello World\n";
pp6:    x1 = x2 / 1x + x _ 2;
pp7:    in case that x1 gt 2 and 1x gteq 3
pp8:        do 1x=2;
pp9:    otherwise
pp10:       jump to pp4;
pp11: ]
```

Figure 1: An example format of the input programming language

```
   label: pp1
punctuation: :
datatype: null
ID: foo
special symbol: (
special symbol: )
label: pp2
punctuation: :
special symbol: [
label: pp3
punctuation: :
datatype: integer_2
ID: x1
punctuation: ,
ID: x2
punctuation: ,
ID: 1x
punctuation: ;
label: pp4
punctuation: :
datatype: character_1
ID: c1
punctuation: ,
ID: 1c
punctuation: ,
ID: _c
punctuation: ;
label: pp5
punctuation: :
datatype: string
ID: s
operator: =
punctuation: "
constant: Hello World\n
punctuation: "
punctuation: ;
label: pp6
punctuation: :
ID: x1
operator: =
operator: x2
operator: /
ID: 1x
operator: +
ID: x
operator:_
constant: 2
punctuation: ;
label: pp7
punctuation: :
reserved keyword: in case
that
ID: x1
reserved keyword: gt
constant: 2
reserved keyword: and
ID: 1x
reserved keyword: gteq
constant: 3
label: pp8
punctuation: :
ID: 1x
operator: =
constant: 2
label: pp9
punctuation: :
reserved keyword:
otherwise
label: pp10
punctuation: :
reserved keyword: jump to
label: pp4
punctuation: ;
label: pp11
punctuation: :
special symbol: ]
```