Sinirli Turkce Programlama Dili Derleyicisi - SiNiR

CmpE 150.03 Introduction to Computing, Fall 2020
Homework 5 – Due: 28/12/2020, 23.59

_____

In this homework, you are going to implement the SiNiR compiler. The details are as follows:

Given a file named **calc.in**, your program should check whether there are any syntax errors.
- There will always be a file named **calc.in**
- There will be no tab character in the file.
- If there is a syntax error,
  - Create a new file, named **calc.out** and print "Dont Let Me Down" inside.
- If there are no syntax errors,
  - Create a new file, named **calc.out** and print "Here Comes the Sun" inside.
- Do not print out anything to the terminal in the submitted file.

calc.in is composed of three parts. If one of those parts are missing, there is a syntax error.

```
AnaDegiskenler
<init-statement>
<init-statement>
<init-statement>
….
YeniDegiskenler
<mid-statement>
<mid-statement>
<mid-statement>
…..
Sonuc
<final-statement>
```

The general structure of **calc.in** should be in the following form:
- Empty lines are allowed anywhere in the file.
- Spaces are allowed at any point unless specified. There should be no space in the shaded sequences. For example, there might be space before AnaDegiskenler or after; however there should be no space between <digit>.<digit>.
- The file should start with the AnaDegiskenler statement. If not, there is a syntax error.
- There are zero or more <init-statement>s after AnaDegiskenler statement. Each <init-statement> should be in a single line. If not, there is a syntax error.
- The file then continues with the YeniDegiskenler statement. If not, there is a syntax error.
- There are zero or more <mid-statement>s after the YeniDegiskenler statement. Each <mid-statement> should be in a single line. If not, there is a syntax error.
- The file continues with the Sonuc statement. If not, there is a syntax error.
- There is zero or one <final-statement> after the Sonuc statement. <final-statement> should be in a single line. If not, there is a syntax error.

Important:
- The language is case sensitive.
- Keywords cannot be used as variable names.

\<init-statement\>

- is constructed by
  - \<var-name\> degeri \<value\> olsun
- Any number of spaces are allowed in \<init-statement\>
- \<var-name\> is composed of at most ten alpha-numeric characters
- The same variable cannot be assigned more than once.
- \<value\> := \<a-term\> | \<l-term\>
  - the above line means that \<value\> can be \<a-term\> or \<l-term\>
- \<a-term\> := \<digit\> | \<digit\>.\<digit\> | \<t-digit\> | \<t-digit\> nokta \<t-digit\>
- \<digit\>   := 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
- \<t-digit\> := sifir | bir | iki | uc | dort | bes | alti | yedi | sekiz | dokuz
- \<l-term\>   := dogru | yanlis
- If above rules do not hold, it's syntax error.

\<mid-statement\>

- is constructed by
  - \<var-name\> degeri \<expression\> olsun
- Any number of spaces are allowed in \<mid-statement\>
- \<var-name\> is composed of at most ten alpha-numeric characters
- the same variable cannot be assigned more than once.
- \<expression\> := \<arith-expr\> | \<logic-expr\>
  - i.e. \<expression\> should be either \<arith-expr\> or \<logic-expr\>
- \<var-name\> will be of type float if it is \<arith-expr\>
- \<var-name\> will be of type boolean if it is \<logic-expr\>
-
- \<arith-exp\> := \<open-paran\> \<arith-exp\> \<close-paran\>
  - := \<arith-exp\> \<binaop\> \<arith-exp\>
  - := \<a-term\>
  - := \<a-var-name\>
- \<open-paran\> := (| ac-parantez

- \<clos-paran\> :=) | kapa-parantez

- \<binaop\> := + | - | * | arti | eksi | carpi

- \<a-var-name\> should represent a variable of type int or float
- \<a-var-name\> should be defined in the previous lines.

- \<logic-exp\> := \<open-paran\> \<logic-exp\> \<close-paran\>
  - := \<arith-exp\> \<binlop\> \<arith-exp\>
  - := \<l-term\>
  - := \<l-var-name\>
- \<binlop\> := ve | veya

- \<l-var-name\> should represent a variable of type boolean
- \<l-var-name\> should be defined in the previous lines.
- If above rules do not hold, it's syntax error.

\<final-statement\>

- is constructed by:
  - \<expression\>
- The same rules for \<expression\> apply (as above).
- Any number of spaces are allowed in \<final-statement\>
- If above rules do not hold, there is a syntax error.

Submission: You will submit a single python file over Moodle. Your .py file should be named with the underscore character (_) followed by your student number (e.g. _2019700030.py).

Your homework will be evaluated in different categories so that you can collect **partial points**:

A	Correctly parse the file without any \<init-statement\>, \<mid-statement\>, \<final-statement\>	+5	If the code fails in this test case, the absolute grade will be **zero**
B	Correctly parse \<init-statement\>s composed of only integers	+10	
C	Correctly parse \<init-statement\> composed of combination of integers, floats and booleans	+15	
D	Correctly parse \<mid-statement\>s composed of only integers	+10	No variable is used on the right side of the statement (after degeri)
E	Correctly parse \<mid-statement\>s composed of combination of integers, floats, booleans	+15	No variable is used on the right side of the statement (after degeri)
F	Correctly parse \<mid-statement\>s composed of only integers and integer variables	+15	This requires also clearing B
G	Correctly parse \<mid-statement\>s composed of any operand	+15	This requires also clearing C
X	Correctly parse any file	+15	This requires also clearing G

Late Submission: Allowed with penalty: -% 10*(number_of_days_late)^2. Example case: You are 2 days late, and you got 90 from evaluation. You will get 90 * (1 - 0.4) = 54 as your final grade. You will get 0 if you are more than 3 days late.

Example calc.in files (more will be provided)

```
AnaDegiskenler
x degeri 4 olsun
x1 degeri 2 olsun
a degeri    dort olsun
b degeri uc nokta dort olsun
y degeri 8 olsun
z degeri 3.2 olsun
b1 degeri dogru olsun
b2 degeri yanlis olsun
```

```
YeniDegiskenler
t1 degeri x + ( y ) - z olsun
t4 degeri x arti ac-parantez y kapa-parantez eksi z olsun
t2 degeri ( t4 +  ac-parantez ( ( y  ) ) - z ) kapa-parantez olsun
t3 degeri b1 ve b1 ve ( b2 veya b1 ) olsun

Sonuc
x arti ac-parantez y kapa-parantez eksi z
```

```
AnaDegiskenler
YeniDegiskenler
Sonuc
```