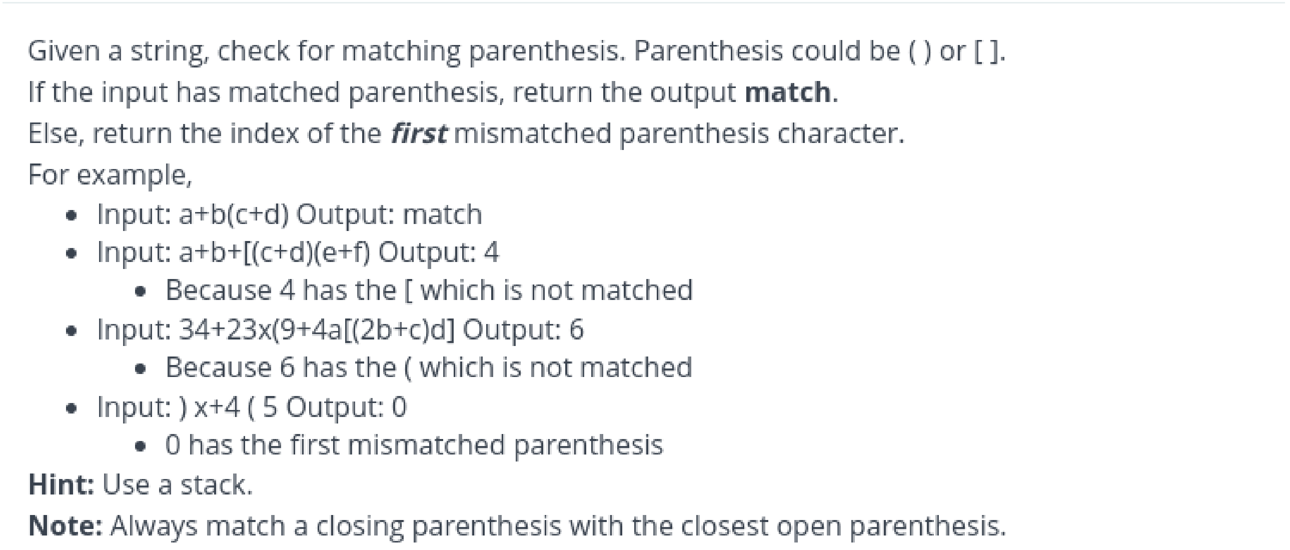
CS571 Lab 7

Sept 2021

1. Implement the question asked in quiz 1. Write a Python program which will match parenthesis. The question from HR is reproduced below:



Given a string, check for matching parenthesis. Parenthesis could be ( ) or [ ].
If the input has matched parenthesis, return the output **match**.
Else, return the index of the ***first*** mismatched parenthesis character.
For example,

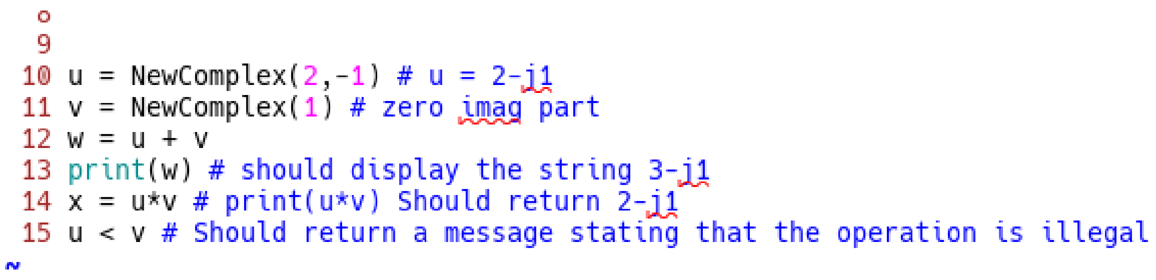
- Input: a+b(c+d) Output: match
- Input: a+b+[(c+d)(e+f) Output: 4
  - Because 4 has the [which is not matched
- Input: 34+23x(9+4a[(2b+c)d] Output: 6
  - Because 6 has the (which is not matched
- Input:) x+4 (5 Output: 0
  - 0 has the first mismatched parenthesis

Hint: Use a stack.
Note: Always match a closing parenthesis with the closest open parenthesis.

2. **Employees.**
   a) Create a class called Employee. The Employee class has the following methods:
      - changeSalary(x): adds x to the current salary
      - changeDept(y): changes the department id to y
      - In addition, the class has to have a constructor, and a method to print the details neatly. (`__init__ and __str__`)

   b) Using the data in `employees.csv`, populate a list of Employee objects.
   c) Give an increment of 60 units to all employees hired before 1 Jan 03.
   d) Write the updated employee details to a new file `employee_updated.csv`.
   e) Print the details of the employees who manage more than 3 employees.

3. Create a class called `NewComplex`, for handling complex numbers. Using the `NewComplex` class, you should be able to do things like:



```
u = NewComplex(2,-1) # u = 2-j1
v = NewComplex(1) # zero imag part
w = u + v
print(w) # should display the string 3-j1
x = u*v # print(u*v) Should return 2-j1
u < v # Should return a message stating that the operation is illegal
```

You need to define the methods `__add__`, `__mul__`, `__le__`, `__lt__`, `__ge__`, `__gt__`, `__str__` to get the above functionality.

4. **Polynomials.**
   a) Implement a class Polynomial, which will represent polynomials.
   b) You must be able to add, display and evaluate polynomials (`__add__`, `__str__`, `__call__`)
   c) Find the roots of the polynomial. You can use the numpy.roots() method for this. Note that this is **not** a dunder method in the Polynomial class.