Term Project – Algorithm Design

Task: 1226. esreveR redrO
Difficulty: 116

Submitted by Kasperi Reinikainen
5818014
This task is a string reversal problem. The idea is to reverse only character-sequences consisting of latin letters, all other characters are left in place.

1. Plain text input, can be multiple lines
2. Reversed sequences only those of latin chars
3. **Reversing twice gives the original input**

Input: ciphered text < 1000 lines. < 255 char / line
Output: deciphered text from the input
Concepts / techniques

Idea was to break the problem into two subproblems:

1. Detecting character-sequences to reverse
2. Reversing strings recursively

I tried to focus on the efficiency of the algorithm as I foresaw that in python this problem might grow to polynomial times and therefore be unnecessarily slow on large inputs.
Concepts / techniques continued

As I did split the problem into 2 parts with emphasizes on running time I was able to find relatively fast algorithm that has time-complexity of about $O(2^n) \Rightarrow O(n)$

- String recursion: Linear, $\Rightarrow O(n)$
- Input line scan: Linear, no nested loops $\Rightarrow O(n)$
Progress

I ended up writing 5 versions of the algorithm.

- Major versions of how it evolved

```python
def reverse(i):
    if len(i) == 1:
        return i
    else:
        return i[:len(i)-1]:reverse(i[:len(i)-1])

def getReverseline(line):
    l = line.split()
    for i in range(len(l[0])):
        firstChar = tempStr[0]
        lastChar = tempStr[len(tempStr)-1]
        if ord(firstChar) > 122 or ord(lastChar) < 65 or (ord(lastChar) > 90 and ord(lastChar) < 97):
            tempStr = reverse(tempStr[::len(tempStr)-1])
        for char in tempStr:
            l[i] = firstChar + tempStr + lastChar
        else:
            tempStr = reverse(tempStr[::len(tempStr)-1])
    for char in tempStr:
        l[i] = reverse(tempStr)
    return ''.join(l)

lines = []
line = raw_input()
while line:
    if line:
        lines.append(getReverseline(line))
    else:
        break
try:
    line = raw_input()
except:
    break
text = '
'.join(lines)
print text
```
def reverse(i):
    if len(i) == 1:
        return i
    else:
        return i[len(i)-1] + reverse(i[:len(i)-1])

def getReverseline(s):
    start = 0
    for i in range(len(s)):
        if (ord(s[i]) < 123 and ord(s[i]) > 64) and (ord(s[i]) < 91 or ord(s[i]) > 96):
            if start < i:
                s = s.replace(s[start:i], reverse(s[start:i]))
            else:
                s = s.replace(s[start:i], reverse(s[start:i]))
                start = i+1
        return s

lines = []
line = raw_input()
while line:
    templine = getReverseline(line)
    print ord(templine[-1])
    lines.append(templine)
    try:
        line = raw_input()
        print ord(line[0])
    except:
        print "\n" in line
        break
text = "\n".join(lines)
print text
```python
def reverse(s):
    if len(s) == 1:
        return 1
    else:
        return s[:len(s)-1]+reverse(s[len(s)-1])

def getReverseLine(s):
    start = 0
    for i in range(len(s)):
        if (ord(s[i]) < 123 and ord(s[i]) > 64) and (ord(s[i]) < 91 or ord(s[i]) > 96):
            if start < i:
                s = s.replace(s[start:i+1], reverse(s[start:i+1]))
            else:
                if start < i:
                    s = s.replace(s[start:i], reverse(s[start:i]))
                start = i+1
        else:
            start += 1
    return s

lines = []
import sys
for t in sys.stdin:
    line = getReverseLine(t)
    if ord(line[-1]) == 10:
        line = line[:-1]
    lines.append(line)
for i in range(len(lines)):
    print lines[i]
```
Progress

4th

```python
@def reverse(i):
    if len(i) == 1:
        return i
    else:
        return i[len(i)-1]+reverse(i[:len(i)-1])

@def getReverseLine(s):
    start = 0
    for i in range(len(s)):
        if (ord(s[i]) < 64 and ord(s[i]) > 96) or (ord(s[i]) < 91 or ord(s[i]) > 96):
            if i == len(s)-1:
                if start < i:
                    s = s.replace(s[start:i+1], reverse(s[start:i+1]))
            else:
                if start < i:
                    temp = s[start:i]
                    r = reverse(s[start:i])
                    s = s.replace(temp, r)
                    start = i+1
                else:
                    start += 1
    return s

import sys
for t in sys.stdin:
    print getReverseLine(t),
```
Conclusion

- 1st version: Run until test case 4
- 2 – 4 versions: Run until test case 9
- Although significant improvement in the way program handles line breaks and a shown improvement in *my own test-cases*, couldn’t pass timus case 9
- Last versions of the program works so well in my own opinion that it’s hard to find the mistake anymore. If timus provided their test cases it would have eased my work to find mistakes.
Used test cases

TEST 1:
This is an example of a simple test. If you did not understand the ciphering algorithm yet, then write the letters of each word in the reverse order. By the way, "reversing" the text twice restores the original text.

TEST 2:
asdad asd
        sf dv

TEST 4:
You must inverse word and all.
But you could forget about the spaces and line breaks

TEST 3:
1
2
3
$

asd#!@#sad%$#%2 sdSA d!23`123?<M@! ^&(&*^&as asf kljdka jd1892u31ASDASD11 end.