

Homework 2

Ryan Reynolds

ryreynol

Due: 9/25/18

**Question 3:**

Code Description and Debugging Report:

The attached Code implements a top down recursive parser that has the following rules:

<s>--><X>z<Y>|<Y>z<X>

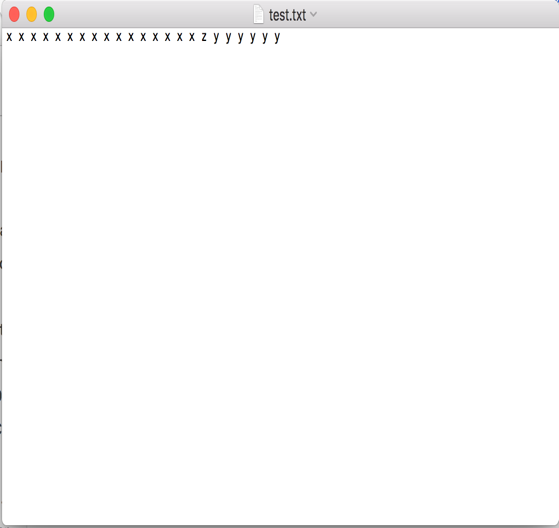
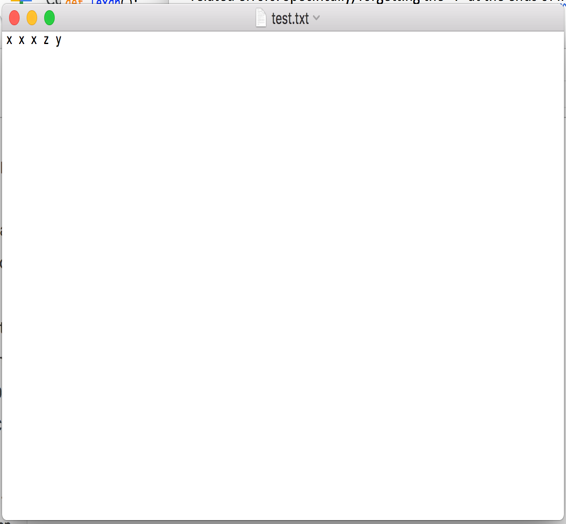
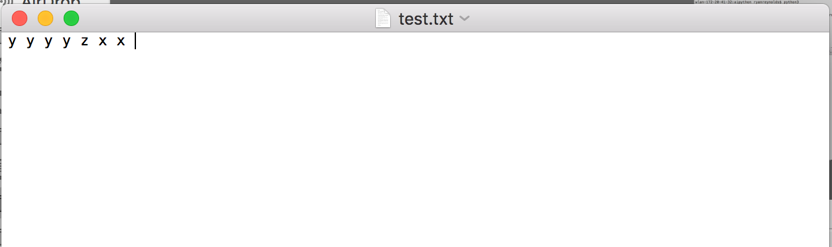
<X>-->x<X>|x

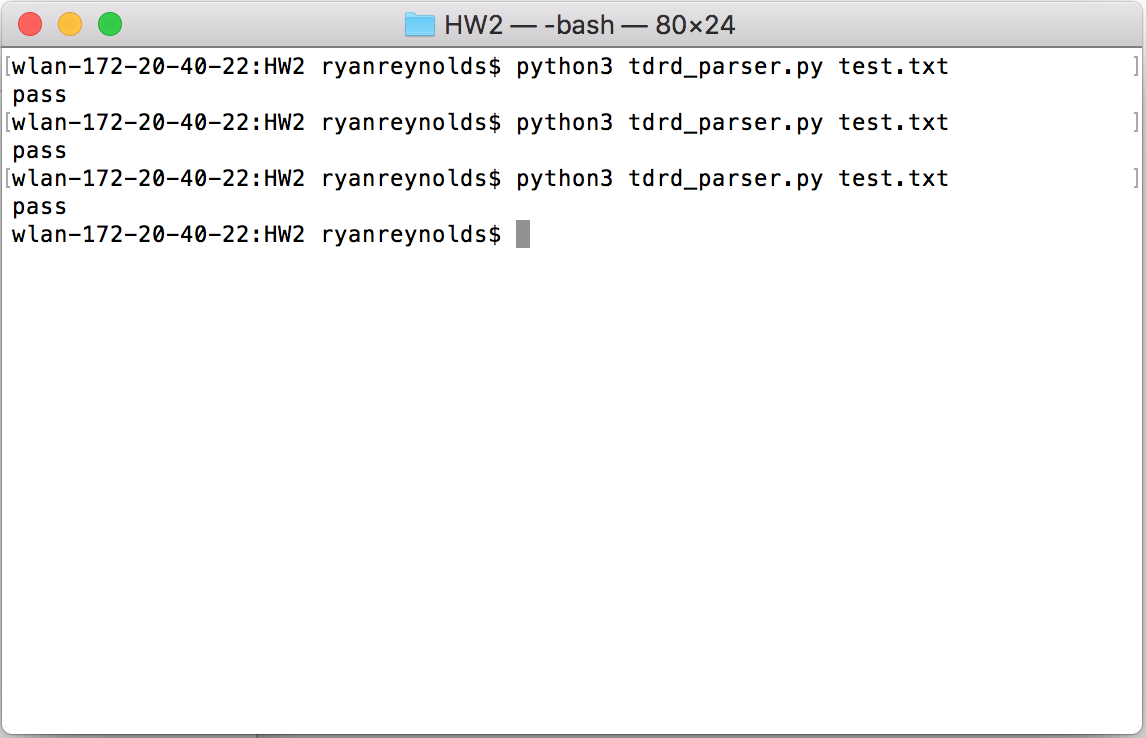
<Y>--y<Y>|y

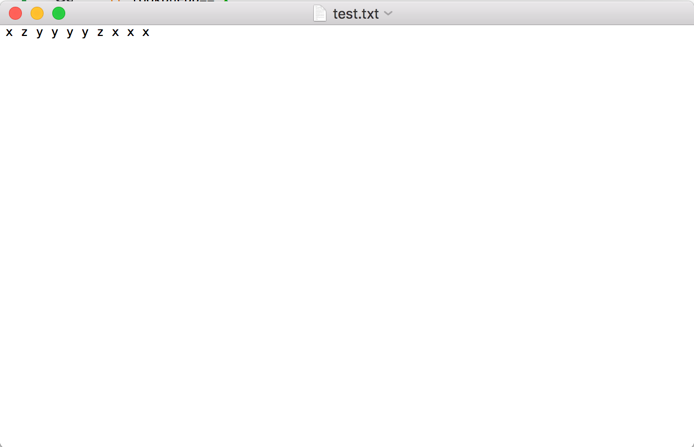
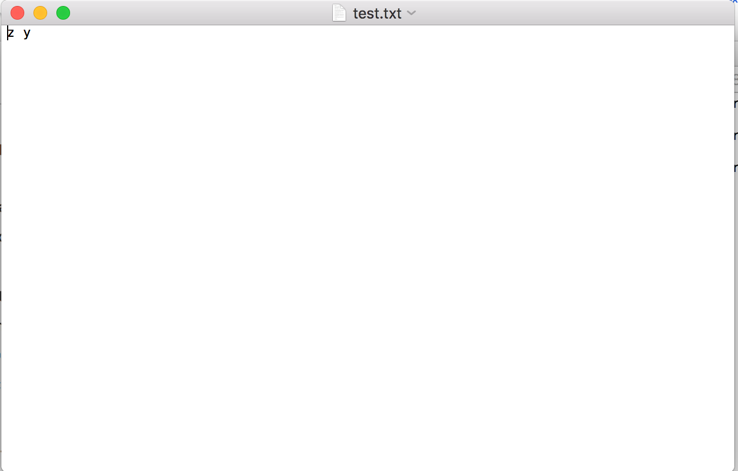
The valid sentences are any combination of consecutive y’s and consecutive x’s separated by a single z.

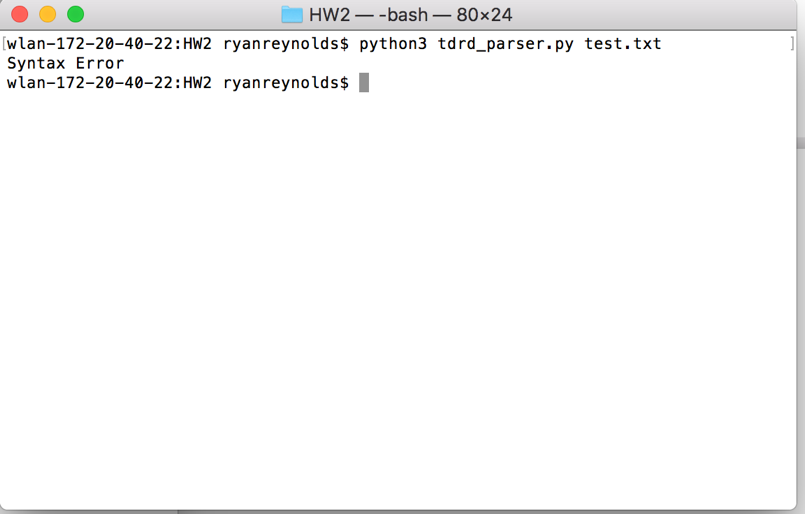
Examples: Valid: y y y y y z x x x x Valid: x x x x x z y y y Invalid: y y z x x z y y Invalid: y x z y y

Implementing a lexical analyzer caused the largest source of errors during programing. This was the first time I used an iterator within python and was not familiar with the declaration. I also came across a logic error within the S() function. Match was set to a capital ‘Z’ instead of lowercase ‘z’ in the elif statement. This caused valid sentences that started with y to be invalid.

Valid Inputs:

Valid Output:

Invalid Inputs:

Invalid Outputs:

Source Code:

# RULES FOR THIS PARSER

# <s>--><X>z<Y>|<Y>z<X>

# <X>-->x<X>|x

# <Y>--y<Y>|y

import sys

#lexical analyzer function to iterate through the input

def lexan():

global mitr

try:

return(next(mitr))

except StopIteration:

return('')

#match function is a utility function to check the token against your symbol table

def match(ch):

global lookahead

if ch==lookahead:

#gets the next token in the input

lookahead=lexan()

else:

print("Syntax Error")

exit()

#WRITE THE S FUNCTION

# <S>--><X>z<Y>|<Y>z<X>

#The S function uses the match and lookahead functions to check the syntax of the input by calling X() and Y() functions via Top down

#recursive descent

def S():

global lookahead

if lookahead=='x':

X()

match('z')

Y()

elif lookahead=='y':

Y()

match('z')

X()

else:

print("Syntax Error")

exit()

#WRITE THE X FUNCTION

# RULE:

# <X>-->x<X>|x

#this function checks for lone x characters and consecutive x characters

def X():

global lookahead

match('x')

if lookahead=='x':

X()

else:

return

#WRITE THE Y FUNCTION

# RULE:

# <Y>--y<Y>|y

#checks to for lone y characters and consecutive y characters

def Y():

global lookahead

match('y')

if lookahead=='y':

Y()

else:

return

#use the open function to create a buffer to the pointed file

file=open(sys.argv[1],"r")

#read the file from the buffer and store into wlist

wlist=file.read().split()

#get the lexemes one by one

#mitr.next will get the next word

#convert the list into an interatable list mitr

mitr=iter(wlist)

#Lookahead is used to always points to the token one index ahead of the current token

lookahead=lexan()

#Call the S() function to start top down recrusive descent

S()

#print whether the string was valid or invalid

if lookahead=='':

print("pass")

else:

print("Syntax Error")