### Encryption / Decryption with Binary Search Trees (Due 16 Nov 2020 )

In this assignment you will create a simple encryption scheme using a binary search tree. To encode a sentence, insert each letter into a binary tree using the ASCII value as a comparative measure.

To encode the sentence *meet me*, start by inserting the letters "m", followed by "e" and followed by "t" into the binary tree. In the first insertion, the binary tree is empty, so "m" becomes the root node of the tree. The "e" is inserted next. Since "e" is less than "m", it becomes the left child of "m" node. The second "e" is not inserted as there is an "e" in the tree already. The "t" becomes the right child of the "m" node. The next character is the space character and is considered less than any letter and becomes the left most leaf.

To encode, use the following convention: assign the root node of the tree a "\*" character. Every other character in the tree, assign a character string based on how many "lefts" and how many "rights" are involved in the tree traversal. For "left" traversals, use a "<", for "right" traversals use a ">". In the above example, "e" will be represented as "<" and "t" will become ">". The space character will become "<<". To complete the code, every character must be separated by a marker called the delimiter. Use "!" (the exclamation mark) as a delimiter for the code.

Using these conventions, the string *"meet me"* when encrypted becomes *"\*!<!<!>!<<!\*!<"*.

For this assignment we are only going to encrypt lower case letters "a" through "z" and the space character. When the input string is given for encryption, convert it to lower case. Only encrypt the lower case letters and spaces. Ignore all digits, punctuation marks, and special characters. Basically, you will drop the the digits, punctuation marks and special characters.

For the encryption to work, a key has to be given. This key is used to create the binary search tree. The encryption key for this assignment is *"the quick brown fox jumps over the lazy dog"*.

You will have to modify the Binary Search Tree code given in class. Here is a suggested outline of that:

import sys

class Tree (object):

# the init() function creates the binary search tree with the

# encryption string. If the encryption string contains any

# character other than the characters 'a' through 'z' or the

# space character drop that character.

def \_\_init\_\_ (self, encrypt\_str):

# the insert() function adds a node containing a character in

# the binary search tree. If the character already exists, it

# does not add that character. There are no duplicate characters

# in the binary search tree.

def insert (self, ch):

# the search() function will search for a character in the binary

# search tree and return a string containing a series of lefts

# (<) and rights (>) needed to reach that character. It will

# return a blank string if the character does not exist in the tree.

# It will return \* if the character is the root of the tree.

def search (self, ch):

# the traverse() function will take string composed of a series of

# lefts (<) and rights (>) and return the corresponding

# character in the binary search tree. It will return an empty string

# if the input parameter does not lead to a valid character in the tree.

def traverse (self, st):

# the encrypt() function will take a string as input parameter, convert

# it to lower case, and return the encrypted string. It will ignore

# all digits, punctuation marks, and special characters.

def encrypt (self, st):

# the decrypt() function will take a string as input parameter, and

# return the decrypted string.

def decrypt (self, st):

def main():

# read encrypt string

line = sys.stdin.readline()

encrypt\_str = line.strip()

# create a Tree object

the\_tree = Tree (encrypt\_str)

# read string to be encrypted

line = sys.stdin.readline()

str\_to\_encode = line.strip()

# print the encryption

print (the\_tree.encrypt(str\_to\_encode))

# read the string to be decrypted

line = sys.stdin.readline()

str\_to\_decode = line.strip()

# print the decryption

print (the\_tree.decrypt(str\_to\_decode))

if \_\_name\_\_ == "\_\_main\_\_"

main()

To run this code on the command line on the Mac you will do

python3 BST\_Cipher.py < bst\_cipher.in

To run the code on the command line on a Windows machine you will do

python BST\_Cipher.py < bst\_cipher.in

You may add other helper functions as needed. There is no need to add

a delete function.

A pangram is a sentence that has all the letters of the alphabet. We

have chosen a classical pangram that has been used by typists for years

- *the quick brown fox jumps over the lazy dog*. Here is a sample

input file - [bst\_cipher.in](https://www.cs.utexas.edu/users/mitra/csFall2020/cs313/assgn/bst_cipher.in). For this

input file the output is given in

[bst\_cipher.out](https://www.cs.utexas.edu/users/mitra/csFall2020/cs313/assgn/bst_cipher.out).

The file that you will be turning in will be called **BST\_Cipher.py**.

The file will have a header of the following form:

# File: BST\_Cipher.py

# Description:

# Student Name:

# Student UT EID:

# Partner Name:

# Partner UT EID:

# Course Name: CS 313E

# Unique Number:

# Date Created:

# Date Last Modified:

For this assignment you may work with a partner. If you do, both of

you must read the paper on  [Pair](https://www.cs.utexas.edu/users/mitra/csFall2020/cs313/assgn/PairProg-CACM-1999.pdf)

[Programming](https://www.cs.utexas.edu/users/mitra/csFall2020/cs313/assgn/PairProg-CACM-1999.pdf)  and submit **only one** copy of the program from

either account but make sure that you have your partner's name and eid

in the header. If you are doing this assignment by yourself, then

remove the *Partner Name* and  *Partner UT EID* from the

header.

Use the  [Canvas](http://canvas.utexas.edu/)  system to

submit your **BST\_Cipher.py** file. We should receive your work by

11 PM on Monday, 16 Nov 2020. There will be substantial penalties if

you do not adhere to the guidelines. Remember Python is case sensitive.

The name of your file must match exactly what we have specified.

* Your Python program should have the proper header.
* Your code must run before submission.
* You should be submitting your file through the web based
* *Canvas* program. We will not accept files e-mailed to us.