Mastering Data Structures and Algorithms
Homework 7

Yiying Peng (Christine)

March 19, 2019

Problem

Create a program that creates and uses the valid word matrix described in class for the Valid-WordSequence recursive problem. i.e. Give a string and a dictionary create a two dimensional boolean matrix where m[i,j] = true if substring(i through j) is valid word and false if not. Then use the best algorithm to employ that matrix to solve the problem that was shown to be O(n!) in its recursive form.

Sol. My Python algorithm implementation is as follows.

```
class ValidWordSequence:
    def __init__(self, word, word_set):
        self.word = word
        self.len = len(self.word)
        self.word_set = word_set
        self.n = len(self.word_set)
        self.matrix = [[False] * self.len] * self.len

    def build_matrix(self):
        for i in range(self.len):
            for j in range(i, self.len):
                if self.word[i:j+1] in self.word_set:
                    print(i, j, self.word[i:j+1])
                    self.matrix[i][j] = True

    def testify(self):
        res = [False] * (self.len + 1)

        # It is always true to reach the starting of string
        res[0] = True

        for i in range(self.len):
            if res[i]:
                for j in range(i, self.len):
                    res[j+1] |= self.matrix[i][j]
        return res[self.len]


def main():
    word = 'yellowbeariscute'

    with open('../resources/word_set_small.txt') as f:
        word_set = set(map(lambda s: s.strip(), f.readlines()))
```

```
    valid_word_sequence = ValidWordSequence(word, word_set)
    valid_word_sequence.build_matrix()
    print(valid_word_sequence.testify())


if __name__ == '__main__':
    main()
```

□