Teacher：张涵翠

Analysis and Design of Algorithm

(semester 2, 2021-2022)

Final Report

Group ID：2

Student IDs：2020329621239　2020329621161

2020329621236

Student Names：刘宓祎 张恺茗 陈佳怡

Class and grade：计算机科学与技术（全英文）20

College of Information Science, Zhejiang Sci-Tech University

Problem 11:Palindrome2

2020329621239 刘宓祎 Algorithm designing and final presentation
2020329621161 张恺茗 Algorithm optimization and PPT making
2020329621236 陈佳怡 Make project summary and report writing

Abstract:

This paper is based on a visual palindrome string segmentation app design by python. After the analysis of time complexity and space complexity, the code we finally designed is presented in the form of "backtracking + dynamic planning". In the code, we also use a large number of libraries to achieve palindromes and window visualization, including PyQt5, Matplotlib, and sys.

I. Introduction

When the forward reading and reverse reading of a string are the same, we call the string palin drome, such as I, DEED, RACECAR.
Any string can be split into a set of palindromes, such as “BUBBASEESABANANA” can be divide d into several palindromes as the following steps:

BUB + BASEESAB + ANANA
B + U + BB + A + SEES + ABA + NAN + A
B + U + BB + A + SEES + A + B + ANANA
B + U + B + B + A + S + E + E + S + A + B + A + N + A + N + A

Now will design a algorithm get the minimum number of palindromes that can be split for a given string and analyze the time and space complexity of the algorithm.

II. Problem analysis and Algorithm design

1. problem analysis

Palindromic substrings are a well-studied topic in stringology and combinatorics on words. **[4]**Since a single character is a palindrome, there are always between n and (n/2)+n=ɵ(n) non-empty palindromic substrings in a string of length n.**[1]**Some infinite strings (e.g., the regular paperfolding sequence) are highly asymmetric in that they contain only a finite number of distinct palindromic substrings.

2. Algorithm comparison

Since all the segmentation schemes of the string s are required, we consider using the search + backtracking method to enumerate all the possible segmentation methods and judge.**[5]**

Tracing algorithm is an organized systematic optimization search technology, which can be regarded as an improvement of brute force method exhaustive search.Backtrack method often avoids searching for all possible solutions, so it is applicable to solving problems with a large number of tissues.**[6]**

The process of backtracking is a process of traversal of a tree, which is divided into transverse traversal and longitudinal traversal. In the backtracking algorithm, the for loop corresponds to the transverse traversal, and the recursion corresponds to the longitudinal traversal.In this question, in traversal, we need to constantly trying to cut strings in different positions.**[7]**

Method 1: backtrack

Calculate the calculation string total length s_l = len (s).

Horizontal traversal: start_index is the first character index of the substring (s), horizontal traversal from left to right constantly attempts to cut the string into s1 and s2, if s1 is a palindrome string, perform a longitudinal search, recursively perform a horizontal traversal of s2, otherwise try a new traversal position.(If the cut s1 is not a palindromic string, it does not meet the topic requirements, so there is no need to cut s2, so directly look for a new cut position.）

Vertical traversal: perform a horizontal traversal of the substring s2 of s;

Returns logic: when start_index is equal to s_l.First, it shows that the multiple strings meeting the transient memory meet the palindrome string requirements.Second, you show that the string segmentation has been finished, so you can put this result in the result sequence.**[8]**

But compared to this design, the following two designs are more optimized, so the following approach is omitted.

Method 2: backtracking + dynamic planning preprocessing

This method adds dynamic programming to the backtracking and it is also an algorithm that we mainly designed.

Assuming that we currently search for the i th character of the string and all of the characters at the s[0..i-1] position have been split into several palindromic strings and the split results are placed into the answer array ans, then we need to enumerate the right boundary j of the next palindromic string so that s[i..j] is a palindromic string.

Therefore, we can start with i and enumerate j from small to large.For the current enumeration of the j value, we use a two-pointer method to determine whether s[i..j] is a palindromic string: if s[i..j] is a palindromic string, add it to the answer array ans and search the next layer with j + 1 as the new i, and remove s[i..j] from the ans during future backtracking.

If we have finished searching the last character of the string, then we find a segmentation method that meets the requirements.

When we judge whether s[i..j] is a palindromic string, the conventional method is to use two pointers to ii and jj, each time to judge whether two pointers point to the same character until the two pointers meet.However, this method produces repeated calculations, such as the following example:

When s=aaba, for the first 2 characters aa, we have 2 segmentation methods [aa] and [a, a], and when we search the i=2 character b of the string, we need to use a double pointer for each s[i..j] whether it is a palindrome string, which produces repeated computation.Therefore, we can preprocess whether each substring s[i..j] of the string s is a palindrome string, using dynamic programming.Let f (i, j) indicate whether s[i..j] is a palindromic string, then there is a state transfer equation:

$$f(i,j)=\begin{cases} \text{True}, & i \geq j \\ f(i+1,j-1) \wedge (s[i]=s[j]), & \text{otherwise} \end{cases}$$

Where the logic and operations, i. e. s[i..j] is a palindromic string, if and only if it is an empty string (i> j), its length is 1 (i=j), or the first-end character is the same and s[i+1..j-1] is a palindromic string.

After the preprocessing is completed, we only need O (1) time to determine whether any s[i..j] is a palindromic string.

Method 3: backtracking + memory search

The dynamic programming preprocessing in Method 1 calculates whether an arbitrary s[i..j]s[i..j] is a palindrome string, and we can also change this step to memorization search.

III. Algorithm analysis

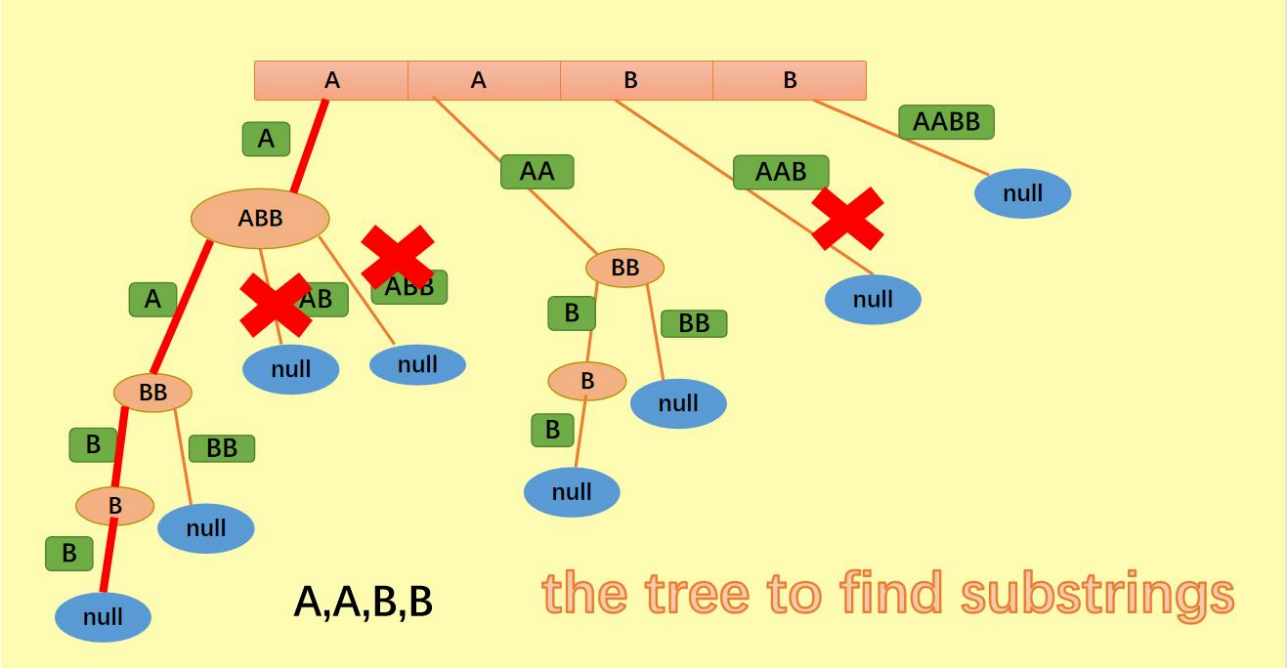
1) backtracking + dynamic planning preprocessing:

Time complexity: O (n *2 ^ n), where n is the length of the string s.In the worst case, s contains n identical characters, so either of its partitioning methods meets the requirements.The number of partition schemes of strings of length n is 2 ^ (n-1) =O (2 ^ n), and each partition method needs O (n) time to find the corresponding partition result and put it into the answer, so the total time complexity is O (n * 2 ^ n).Although O (n ^ 2), it is less than O (n * 2 ^ n in the progressive sense) and can be ignored.

Space complexity: O (n ^ 2), the space of the return answer is not calculated here.Array f required space is O (n ^ 2)

In the backtracking process, we need to use the stack space of O (n) and the space of O (n) used to store the current string segmentation method.Since O (n) is smaller than O (n ^ 2) in the step sense, the spatial complexity is O (n ^ 2).

We can draw the tree as:



2) backtracking + memory search:

Time complexity: O (n * 2 ^ n), where n is the length of the character s, the same as backtracking + dynamic planning preprocessing.

Spatial complexity: O (n ^ 2), the same as method 1.

IV. Experiment preparing

Language: python 3.10

IDE： VScode

Some external libraries introduction:

1) **PyQt5:**

PyQt5 is a python interface based on Digia's powerful graphics framework Qt5, which consists of a set of python modules.The Py Qt5 has multiple classes and functions.Can run on multiple platforms such as Unix, Windows, and Mac OS, There are three major modules in the common interface design: first, the Qt Core module, It covers the core of non-GUI functionality, This module is mainly used for directories, files, data types, text flow, time, mime, processes or thread objects involved in program processing; Second, the Qt Gui module, The module includes a variety of classes that handle basic graphics functions, Covering but not limited to: interface design, event processing, basic images, 2D graphics and text fonts; Third, the Qt Widgets module, The module contains a complete set of UI element components, For designing interfaces in multiple system styles, easy operation.**[2]** In this design, Windows, labels, and buttons are all designed from this library.So we need to download the library before running it.

2) **Matplotlib**:

Matplotlib work library is a simple programming software integrating various powerful functional advantages, no matter what kind of software visualization, Matplotlib can be more perfect beyond.Compared with other visualization software, Matplotlib is not only open source, but also more powerful applications, its advantages, feasibility, and development, etc.This library is required for the painting window in this design.**[3]**

3) **Sys**:

The sys module includes a very practical set of services, including many functional methods and variables to handle Python runtime configuration and resources, which can interact with the system environment outside the previous program, such as the python interpreter.**[9]**

V. Code with annotations

```
# -*- coding: utf-8 -*-


# Form implementation generated from reading ui file 'lmy.ui'
#
# Created by: PyQt5 UI code generator 5.15.6
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.
import numbers
from PyQt5 import QtCore, QtGui, QtWidgets
import sys
from PyQt5.QtWidgets import QApplication, QWidget, QTextEdit
```

```
from matplotlib import widgets
from typing import List
ret = list() # the overall substring
ans = list() # the single substring
class Solution:
    def partition(self, s: str) -> List[List[str]]:
        ret.clear()
        min_num=10000000
        n = len(s)
        f = [[True] * n for _ in range(n)]
        for i in range(n - 1, -1, -1):
            for j in range(i + 1, n):
                f[i][j] = (s[i] == s[j]) and f[i + 1][j - 1]
#depth first search
        def dfs(i: int):
            if i == n:
                ret.append(ans[:])
                return

            for j in range(i, n):
                if f[i][j]:
                    ans.append(s[i:j+1])
                    dfs(j + 1)
                    ans.pop()
        dfs(0)
        return ret
textEdit: QTextEdit = None  # intial string text
final_TexeEdit: QTextEdit = None  # final substring text
#The UI part of the window, including page styles written in CSS, etc
class Ui_MainWindow(object):
```

```
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(800, 600)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.label = QtWidgets.QLabel(self.centralwidget)
        self.label.setGeometry(QtCore.QRect(30, 10, 121, 31))
        self.label.setObjectName("label")
        self.label_2 = QtWidgets.QLabel(self.centralwidget)
        self.label_2.setGeometry(QtCore.QRect(30, 180, 111, 21))
        self.label_2.setObjectName("label_2")
        self.label_3 = QtWidgets.QLabel(self.centralwidget)
        self.label_3.setGeometry(QtCore.QRect(40, 350, 231, 51))
        self.label_3.setObjectName("label_3")
        self.pushButton = QtWidgets.QPushButton(self.centralwidget)
        self.pushButton.setGeometry(QtCore.QRect(600, 490, 93, 28))
        self.pushButton.setObjectName("pushButton")
        self.textEdit = QtWidgets.QTextEdit(self.centralwidget)
        self.textEdit.setGeometry(QtCore.QRect(40, 50, 441, 121))
        self.textEdit.setObjectName("textEdit")
        self.textEdit_2 = QtWidgets.QTextEdit(self.centralwidget)
        self.textEdit_2.setGeometry(QtCore.QRect(40, 230, 441, 121))
        self.textEdit_2.setObjectName("textEdit_2")
        self.textEdit_3 = QtWidgets.QTextEdit(self.centralwidget)
        self.textEdit_3.setGeometry(QtCore.QRect(40, 410, 441, 121))
        self.textEdit_3.setObjectName("textEdit_3")
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 800, 26))
        self.menubar.setObjectName("menubar")
```

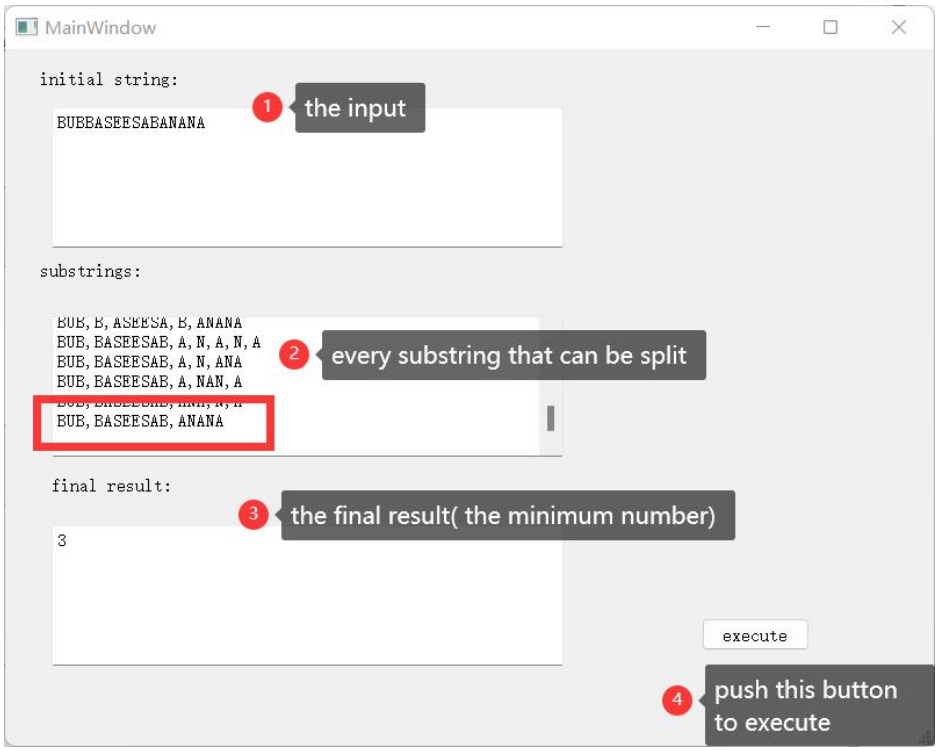
```
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)
        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)
    #The logical part of the UI part, which is used to handle various operations
of the window
    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "MainWindow"))
        self.label.setText(_translate("MainWindow", "initial string:"))
        self.label_2.setText(_translate("MainWindow", "substrings:"))
        self.label_3.setText(_translate("MainWindow", "final result:"))
        self.pushButton.setText(_translate("MainWindow", "execute"))
        self.textEdit.setHtml(_translate("MainWindow", "<!DOCTYPE HTML PUBLIC
\"-//W3C//DTD HTML 4.0//EN\" \"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"
"<html><head><meta name=\"qrichtext\" content=\"1\" /><style
type=\"text/css\">\n"
"p, li { white-space: pre-wrap; }\n"
"</style></head><body style=\" font-family:\'SimSun\'; font-size:9pt;
font-weight:400; font-style:normal;\">\n"
"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;
-qt-block-indent:0; text-indent:0px;\">ABACDDEE</p></body></html>"))
        self.textEdit_2.setHtml(_translate("MainWindow", "<!DOCTYPE HTML PUBLIC
\"-//W3C//DTD HTML 4.0//EN\" \"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"
"<html><head><meta name=\"qrichtext\" content=\"1\" /><style
type=\"text/css\">\n"
"p, li { white-space: pre-wrap; }\n"
```

```
"</style></head><body style=\" font-family:\'SimSun\'; font-size:9pt;
font-weight:400; font-style:normal;\">\n"
"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;
-qt-block-indent:0;
text-indent:0px;\">here are all possible paindrome substrings<br
/></p></body></html>"))
        self.textEdit_3.setHtml(_translate("MainWindow", "<!DOCTYPE HTML PUBLIC
\"-//W3C//DTD HTML 4.0//EN\" \"http://www.w3.org/TR/REC-html40/strict.dtd\">\n"
"<html><head><meta name=\"qrichtext\" content=\"1\" /><style
type=\"text/css\">\n"
"p, li { white-space: pre-wrap; }\n"
"</style></head><body style=\" font-family:\'SimSun\'; font-size:9pt;
font-weight:400; font-style:normal;\">\n"
"<p style=\" margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px;
-qt-block-indent:0;
text-indent:0px;\">here is the minimum number of palindronme substrings<br
/></p></body></html>"))
class NameController(QtWidgets.QMainWindow,Ui_MainWindow):
        def __init__(self):
            super().__init__()
            self.setupUi(self)
            self.controller() # add slots to the specific functions
        def controller(self):
                self.pushButton.clicked.connect(self.d)


        def d(self):
                global stri,stry
                s = Solution()
                #stri=initial_TextEdit.toPlainText()
                stri=self.textEdit.toPlainText()
```

```
                #stri=self.textEdit.toPlainText()
                global min_num
                min_num=10000000
                #stri="BUBBASEESABANANA"
                s.partition(stri)
                k=0
                for k in range(k, len(ret)):
                        if len(ret[k])<min_num:
                                min_num=len(ret[k])
                l=0
                stry=""
                for l in range(l, len(ret)):
                        stry+=','.join(ret[l])+"\n"
                self.textEdit_2.setText(stry)
                self.textEdit_3.setText(str(min_num))


if __name__ == '__main__':
    app = QtWidgets.QApplication(sys.argv)    # Instantiate an app
    window = NameController()       # Instantiate a window
    window.show()                  # Show window at default size
    sys.exit(app.exec_())           # Run the app and do some cleanup when the app
exits
```

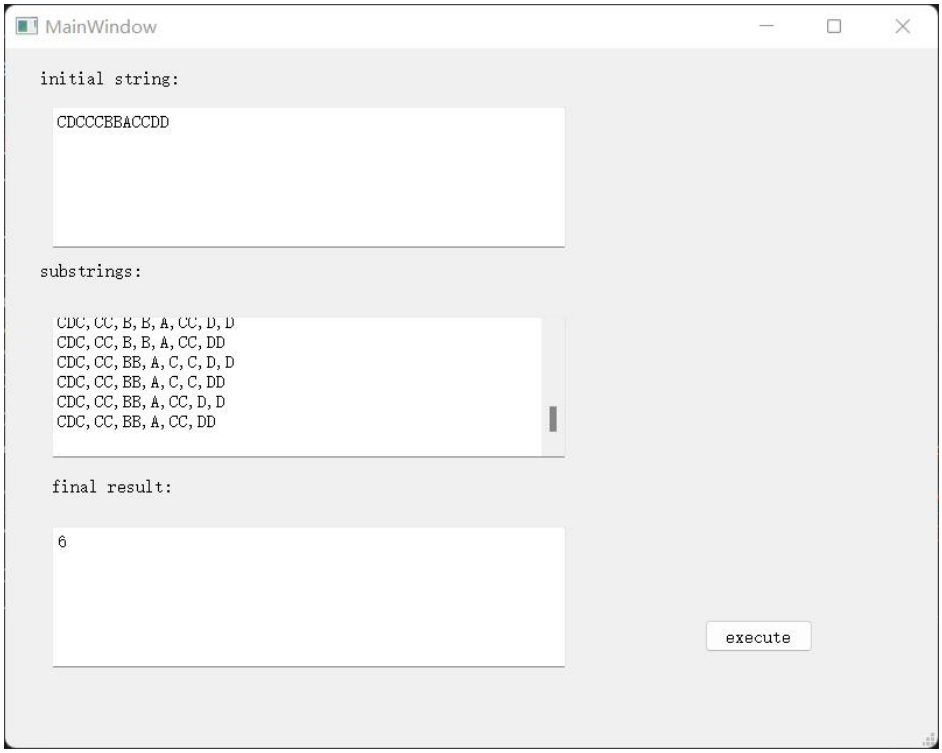
Sample input 1: **BUBBASEESABANANA**

Sample result 1:



Sample input 2: CDCCCBBACCDD

Sample result 2:



VI. Conclusions

1. about algorithm

Through the production of this project, the team has completed the task of finding the

minimum number of palindrome substrings from a given string, and found the relevant characteristics, advantages and disadvantages in the comparison of backtracking, dynamic planning and other algorithms, and finally realized the mastery of greed, dynamic planning and string related operations.

2. about cooperation in one team

Stephen Robbins first proposed the concept of a "team": a formal group of collaborative individuals in order to achieve a certain goal.[10] So teamwork is very important,and we have benefited a lot from the design of such a program.The design of the team is difficult, but it also improves our teamwork ability and makes us benefit forever.

VII. Arrangement and Schedule

Student	Task	Schedule	
刘宓祎	Algorithm designing and the final presentation	Search information	2022.5.10-2022.5.11
		Algorithm designing	2022.5.11-2022.5.15
		Visual page making	2022.5.15-2022.5.20
张恺茗	Algorithm optimization and PPT making	Search information	2022.5.20-2022.5.23
		Algorithm optimization	2022.5.23-2022.5.27
		PPT slide making	2022.5.27-2022.5.29
陈佳怡	Make project summary and report writing	Search information	2022.5.27-2022.5.29
		Project summarize	2022.5.29-2022.5.30
		Report writing	2022.5.30-2022.6.1

References

Article:

[1]Gabriele Fici; Travis Gagie; Juha Kärkkäinen; Dominik Kempa[J].Journal of Discrete

[2]AlgorithmsVolume 28, 2014. PP 41-48

[3]姜华林.基于 PyQt5 界面的词云制作软件设计[J].电脑知识与技术,2021,17(13):74-76+92.

[4]薛先贵,黎路.数据可视化教学中 Matplotlib 教学应用探讨[J].福建电脑,2020,36(12):215-216.

[5]毛梓铭.回文数，妙不可言[J].数学小灵通(5-6 年级版),2021(11):24-29.

Website:

[5] https://blog.csdn.net

[6] https://blog. csdn.net/weixin_40302130/article

[7] https://blog.csdn.net/qq_37763204/article

[8] https://blog.csdn.net/baidu_36399851/article

[9] https://blog.csdn.net/cheng_5230/article

[10] https://baike.baidu.com/item/团结