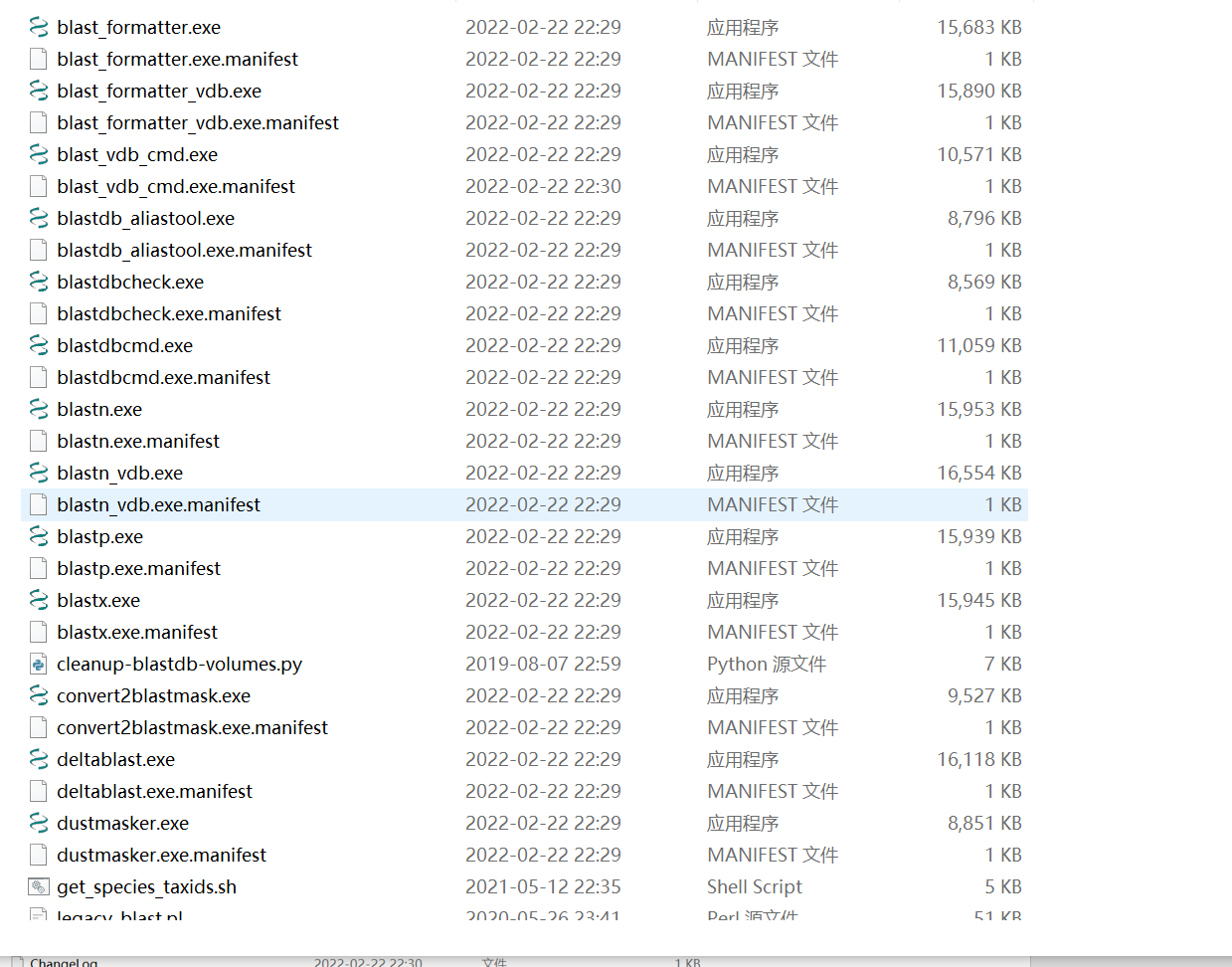
**第二讲作业：序列比对**

## 作业一：下载Blast软件，完成人和牛ACE2蛋白序列的本地Blast比对

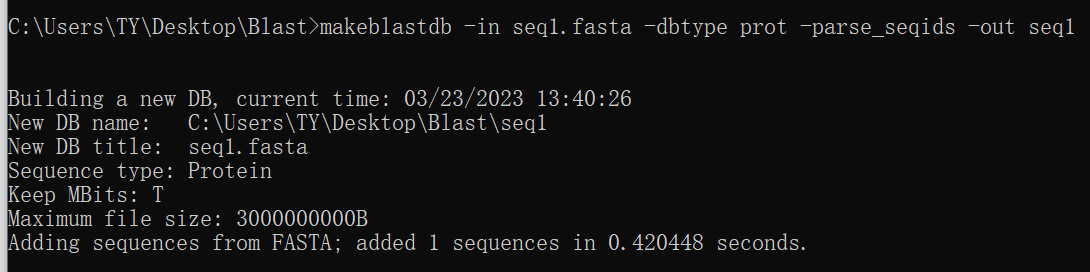
1.下载Blast软件。





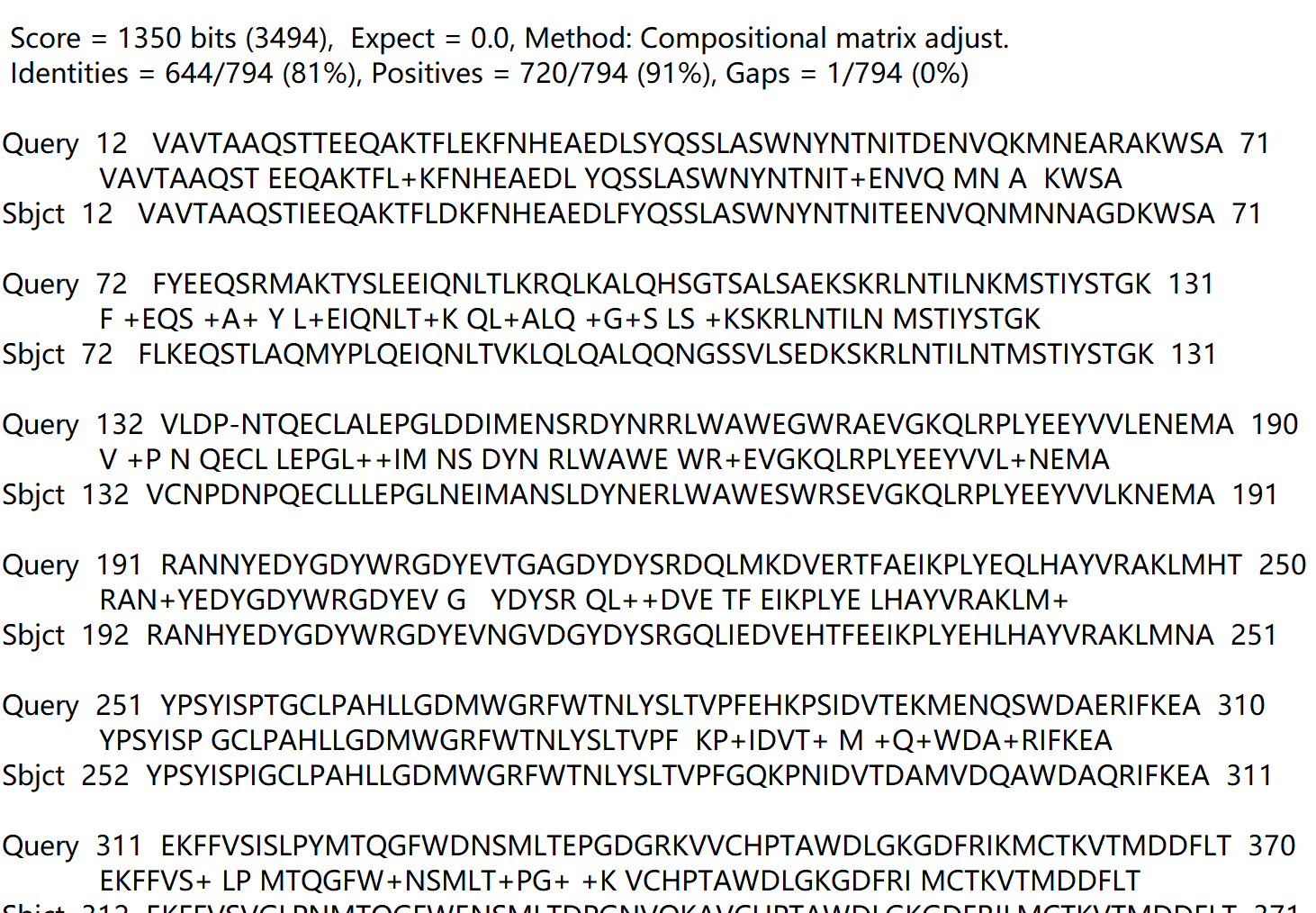
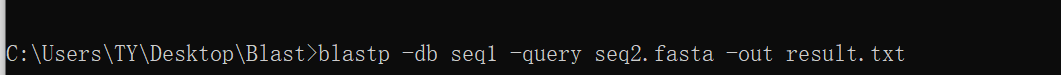
2.本地比对人和牛的ACE2蛋白序列。

建库：





比对：



## 作业二：利用python实现SW比对算法

1.代码

import numpy

A = "TGTTACGG"         # DNA序列

B = "GGTTGACTA"

n, m = len(A), len(B)  # 两个序列的长度

W = 2  # 空位罚分

# 判分

def score(a, b):

    if a == b:

        return 3

    else:

        return -3

# 字符串

def point(x, y):

    return '[' + str(x) + ',' + str(y) + ']'

# 回溯

def traceback(value, result):

    if value:

        result.append(value)

        value = path[value]

        x = int((value.split(',')[0]).strip('['))

        y = int((value.split(',')[1]).strip(']'))

    else:

        return

    if H[x, y] == 0:  # 终止条件

        xx = 0

        yy = 0

        s1 = ''

        s2 = ''

        md = ''

        for item in range(len(result) - 1, -1, -1):

            position = result[item] # 取出坐标

            x = int((position.split(',')[0]).strip('['))

            y = int((position.split(',')[1]).strip(']'))

            if x == xx: # 判断是否为左方元素

                s1 += '-'

                s2 += B[y - 1]

                md += ' '

            elif y == yy: # 判断是否为上方元素

                s1 += A[x - 1]

                s2 += '-'

                md += ' '

            else:   # 判断是否为左上元素

                s1 += A[x - 1]

                s2 += B[y - 1]

                md += '|'

            xx = x

            yy = y

        # 输出最佳匹配序列

        print('s1: %s' % s1)

        print('    ' + md)

        print('s2: %s' % s2)

    else: # 未到终点 继续回溯

        traceback(value, result)

H = numpy.zeros([n+1, m+1], int)

path = {}

for i in range(0, n+1):

    for j in range(0, m+1):

        if i == 0 or j == 0:

            path[point(i, j)] = None

        else:

            s = score(A[i-1], B[j-1])

            L = H[i-1, j-1] + s

            P = H[i-1, j] - W

            Q = H[i, j-1] - W

            H[i, j] = max(L, P, Q, 0)

            # 添加进路径

            path[point(i, j)] = None

            if L == H[i, j]:

                path[point(i, j)] = point(i-1, j-1)

            if P == H[i, j]:

                path[point(i, j)] = point(i-1, j)

            if Q == H[i, j]:

                path[point(i, j)] = point(i, j-1)

end = numpy.argwhere(H == numpy.max(H))

for pos in end:

    key = point(pos[0], pos[1])

    traceback(path[key], [key])

2.思路：

3.Terminal：

