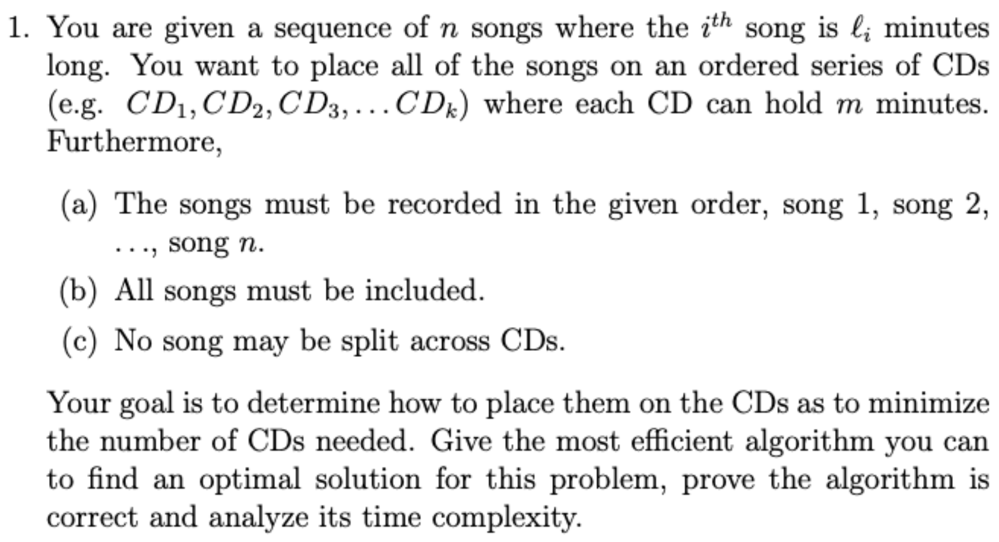
9101 Assignment 2

Haojin Guo

z5216214

Q1.



1. You are given a sequence of n songs where the i^{th} song is ℓ_i minutes long. You want to place all of the songs on an ordered series of CDs (e.g. $CD_1, CD_2, CD_3, \ldots CD_k$) where each CD can hold m minutes. Furthermore,

   (a) The songs must be recorded in the given order, song 1, song 2, . . ., song n.

   (b) All songs must be included.

   (c) No song may be split across CDs.

   Your goal is to determine how to place them on the CDs as to minimize the number of CDs needed. Give the most efficient algorithm you can to find an optimal solution for this problem, prove the algorithm is correct and analyze its time complexity.

Solution,

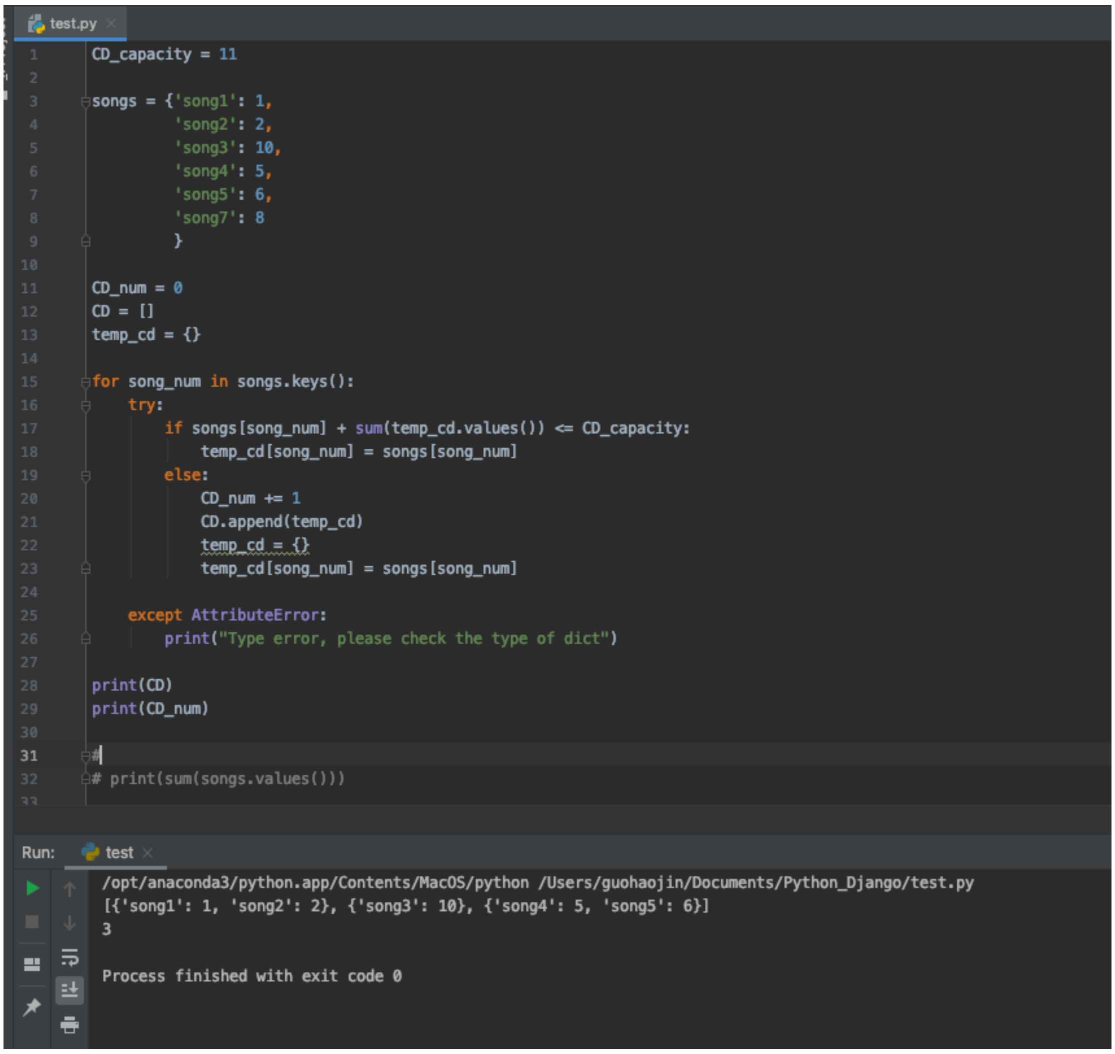
In order to satisfy the condition of (a),

1) it is need to traverse each song if the n song. (the complexity cost is O(n))
2) Then, putting as many songs as we can in CD1 within the capacity of m minutes. When CD1 is full, do the same as before for CD2, then CD3,
3) Until all the songs are stored in the CDs in the given order.

Proof,

1) Suppose there is another optimal solution called S (the number of CDs in S is equal to the solution we called G above) but it violates the rule of greedy algorithm.
2) Then suppose that the number of songs stored in the m-th CD in solution G is i, and the number of songs stored in the m-th CD in solution S is j. Because the greedy algorithm requires that the number stored in the m-th CD be as large as possible. Hence, $i \geq j$.
3) Then we transfer the first j-i songs from the m+1th CD in S to the m-th CD, here we can ensure that the previous i songs will not exceed the capacity of the CD, so there will be no additional CD quantity, and at the same time Songs removed from m+1 CDs will not increase the number of CDs.
4) Repeat 3), solution S will turn into the solution G (greedy solution.)

Therefore, the solution of greedy solution in this question is optimal.



```python
CD_capacity = 11

songs = {'song1': 1,
         'song2': 2,
         'song3': 10,
         'song4': 5,
         'song5': 6,
         'song7': 8
         }

CD_num = 0
CD = []
temp_cd = {}

for song_num in songs.keys():
    try:
        if songs[song_num] + sum(temp_cd.values()) <= CD_capacity:
            temp_cd[song_num] = songs[song_num]
        else:
            CD_num += 1
            CD.append(temp_cd)
            temp_cd = {}
            temp_cd[song_num] = songs[song_num]

    except AttributeError:
        print("Type error, please check the type of dict")

print(CD)
print(CD_num)

#
# print(sum(songs.values()))
```

```
/opt/anaconda3/python.app/Contents/MacOS/python /Users/guohaojin/Documents/Python_Django/test.py
[{'song1': 1, 'song2': 2}, {'song3': 10}, {'song4': 5, 'song5': 6}]
3

Process finished with exit code 0
```