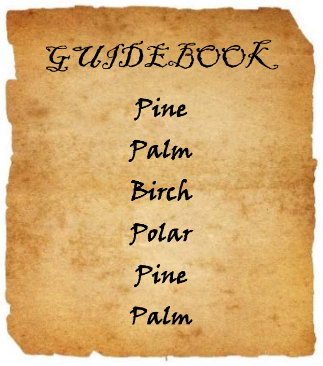
Green Walker Problem

➢ Problem Overview

Once upon a time, there was a Green Walker who lived in RDFZ. As far as the students knew, it hid in the daytime and crept out at night. Getting enough visits and rests under trees without any human was the only way it could avoid hurting people. For these trees, the Green Walker had a special ‘guidebook’ every night: It must follow the **order** of tree species given in the guidebook to rest under. (i.e. In this sample guidebook, it would first select a pine in the pine candidate set to visit, and then select a palm in the palm candidate set to visit, etc.) Plus, the trees (both their species and location) in RDFZ might **vary** from nights to nights. Usually, it could slowly finish its visits within a whole night. However, occasionally there were HACKATHON participants leaving late at night and CS students coming for exams in the morning. Therefore, it had to speed up to keep away from the crowds. The problem was that the Green Walker wanted to **minimize** its time on the way while keeping its rules of visiting trees. You were the one(s) helped it out, please show how you managed to do that.



A SAMPLE GUIDEBOOK

✧ Besides this instruction document are other two folders, ‘nights’ and ‘samples.’
✧ In the folder ‘nights’ there are five folders, each including a map and a guidebook file in .csv (Comma-Separated Values). Each folder is called a ‘night.’ You are expected to solve as many nights as possible. All input and output file should be in .csv.
✧ Your code should output and present your answer in solution-t.csv., where ‘t’ should be the total time taken in the nearest integer second (solution-20 means that the walk takes 20±0.5s). In this file, there should be a list of coordinates with the corresponding tree names, which as a whole is expected to be an optimal track for the Green Walker to go through while following the rule.

Use any programming language (or even any methods), choose a condition (M1 or M2) to solve the problem for each night (5 nights in total):

M1: The Green Walker can go to each tree for **more than one time**.

Note:

1. For this question, an answer code is given. In addition to the algorithm in this code, please come up with a new algorithm. In all, you should submit two codes. However, you are expected to make some changes to the given algorithm.

2. The answer code is in python. If you cannot understand python, please contact the student committee.

M2: The Green Walker can go to each tree for **only one time**.

➢ **Assumptions:**

✧ Green Walker's walk speed was constant (1 m/s).

✧ Green Walker was walking from one tree to another in a straight line, no block.

✧ Green Walker started from the coordinate (0,0)

✧ Ignore the resting time under every tree.

✧ There were **only** cedars, plane trees, palms, pines, maidenhair trees, birch, and poplar in the school.

✧ For each visit, the next one to it would not be the same type of tree. If there was a cedar on the list, the next one on the list could be anything but cedar.

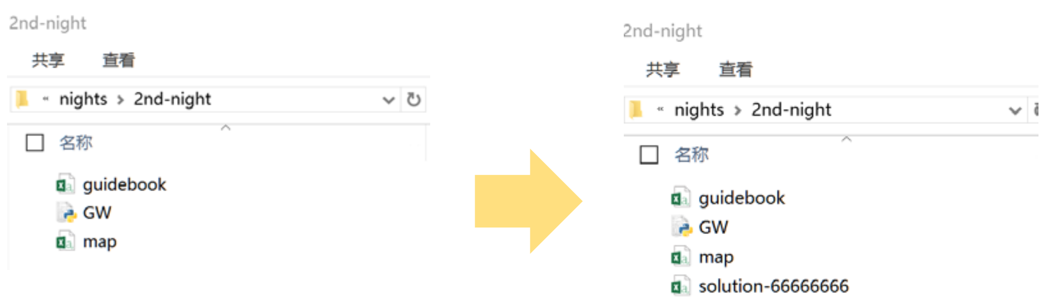
✧ All coordinates are non-negative integer.

Note:

✧ You have to follow assumptions listed above in order to get a better score in Code score. After codes are judged, you will be informed whether you are qualified for the Thesis Defense stage. You may create your own rules and clarify the reason why in the Thesis Defense. We expect a well-rounded answer.

✧ When we put your code in a 'night' file, which contains a guidebook.csv and a map.csv, it should automatically output a solution-t.csv.

e.g.



Do not write a code that process all the 5 nights.

✧ Your code will be judged on readability, simplicity, accuracy of the solution, time complexity, and space complexity.

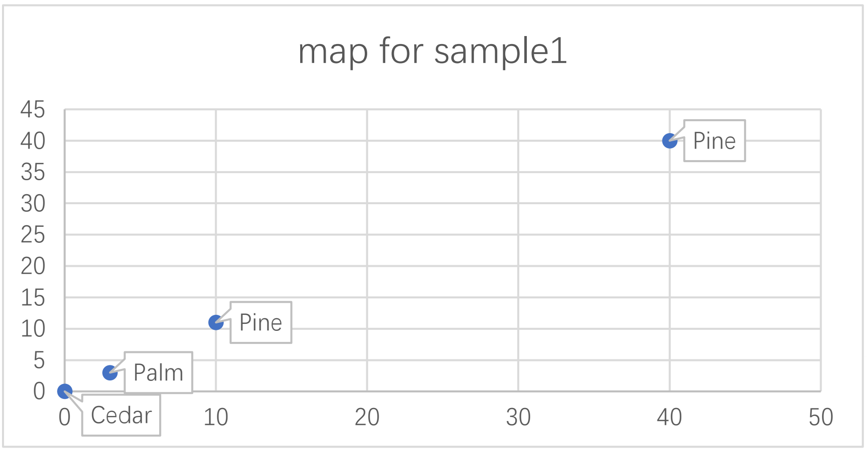
✧ Besides the given databases, we will judge your code using extra databases, so you should try to solve the problem for a general purpose. Do not write the solutions to the question directly in your code.

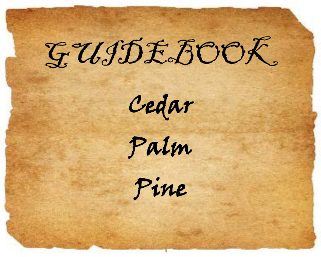
To help you understand…

➢ **Samples:**

There are 3 samples in the 'samples' folder. In sample1 and sample2, each consists of map.csv, guidebook.csv, 1-solution-t.csv (solution file for M1), and 2-soluton-t.csv (solution file for M2). Sample3(only for M1) contains a code whose output solution will be fully marked, but the code used tricks, so this cannot pass the further extra database.

✧ Sample1:



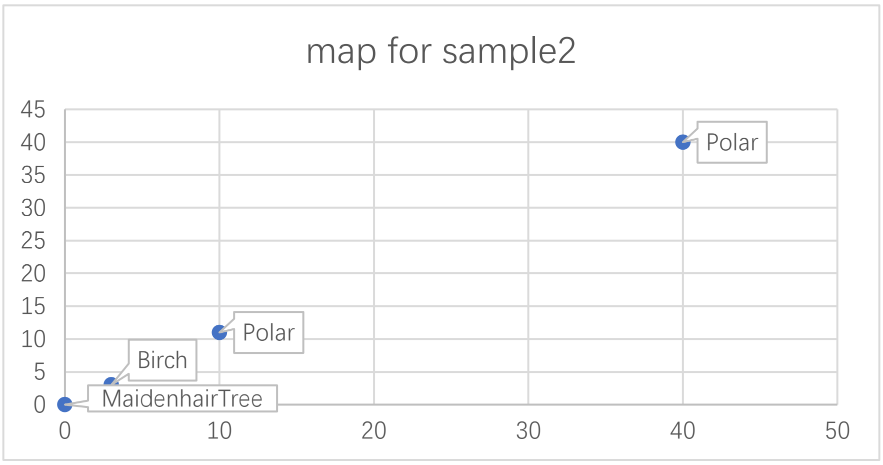


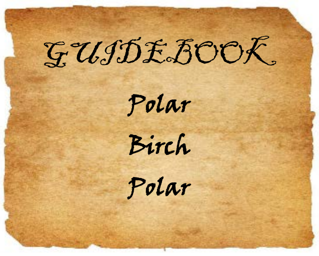
Here is the tree map in coordinate system. According to the guidebook.csv, the Green Walker has to first go to Cedar, second Palm, and third Pine.

For M1(allowed to go to each tree for more than one time): There are two possible routes in total. The optimal track for it is (0,0), (3,3), (10,11).

For M2(not allowed to go to each tree for more than one time, going through each tree without repetition): There are two possible routes in total. The optimal track for it is (0,0), (3,3), (10,11).

✧ Sample2:





Here is the tree map in coordinate system. According to the guidebook.csv, the Green Walker has to first go to Polar, second Birch, and third Polar.

For M1(allowed to go to each tree for more than one time): There are four possible routes in total. The optimal track for it is (10,11), (3,3), (10,11).

For M2(not allowed to go to each tree for more than one time, going through each tree without repetition): There are two possible routes in total. The optimal track for it is (10,11), (3,3), (40,40).

✧ Sample3:

This contains a map (50 trees), a guidebook (10 visits) and a sample code(python). Once processed, this sample code will automatically output a solution file. The solution file will be judged for full marks. However, this code is written with the answer directly in it. If you write code like this, you will be penalized.

Note:

This sample code and sample solution is ONLY for M1.

```
import csv

#if you write code like this, you will be penalized.
Track=[[19, 18, 'Birch'], [20, 19, 'Cedar'], [27, 26, 'Pine'], [22, 33, 'Cedar']
with open('solution-90.csv', 'w',newline='') as csvfile:
    spamwriter = csv.writer(csvfile,dialect='excel')
    for i in Track:
        spamwriter.writerow(i)

print('Good Luck')
print('Have Fun')
print('')
input('press any to quit')
```

➢ **Input and Output Format Illustrations**

✧ Input examples of guidebook (which is in guidebook.csv in each nth-night file):

[Cedar,Palm,Pine,MaidenhairTree,Cedar,Palm,Birch]

[Palm,Polar,Pine,Birch,Pine,MaidenhairTree,PlaneTree]

Where each element in the set is the tree name. In the first set, the Green Walker would have to go to firstly cedar, then palm, and finally birch. In the second set, the Green Walker would have to go to firstly palm, then polar, pine, birch, pine again, and finally plane tree. There might be many cedars or pines. Although all of the trees were different, but all of them were qualified for a visit if they were on the guidebook.

- ✧ Input examples of the map (which is in map.csv in each nth-night file):
  [[1,1,Cedar],[3,1,Palm]]
  [[4,1,Pine],[50,1,PlaneTree],[5,100,Pine]]
  Where in each element[x,y,n], x is the x-coordinate, y is the y-coordinate, n is the species of the tree. [1,2,Pine] means there is a pine tree 1 meter eastwards from the original point and 2 meters northwards.

- ✧ Output examples of the solution(which should be in solution-t.csv in each nth-night file):
  [[1,1,Cedar],[3,1,Palm]]
  [[4,1,Pine],[50,1,PlaneTree],[5,100,Pine]]
  Where in each element[x,y,n], x is the x-coordinate, y is the y-coordinate, n is the species of the tree. [[0,0,Start],[1,1,Cedar],[3,1,Palm]] means that the Green Walker would start at (0,0), and would first go to the cedar at(1,1), then the palm at (3,1).

Note:

1. **Your code is required to automatically output a csv file.**
2. **You should strictly follow the output format, or the computer will not be able to judge your code.**
3. **The submission should be your code ALONE.**
4. **Once you have submitted your code, we have your permission to release or publish your code and your ideas in your Thesis Defense.**