Mark Scheme for Green Walker Problem

- **Evaluation**

(X/50 is where X denotes marks awarded, and 50 is the total marks available. X*/50 is where marks awarded can exceed its denominator)

- Code (60%)
  1. Completeness (2%, X/2)
     a) Run time error
     b) Syntax error

     Note

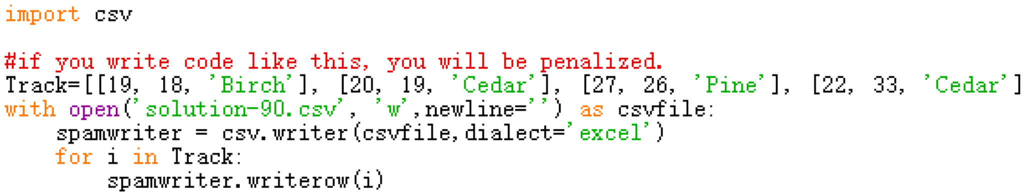
     We may help you to correct errors if we can easily identify them.
  2. Code Quality (8%, X/8)
     a) Simplicity (redundant program line) [3]
        i. If we delete a redundant program line, the program can still work without any difference (or with only a subtle reduction in execution time and space usage).

     b) Readability (comments, variable name) [5]
        i. Understandable variable names
        ii. Clear division and comments of the function of each part.
        iii. Explain your core algorithm in structured English or plain language that is easy to understand.

     Note

     Readability is important. Any suspicious part may be identified as 'tricks' that will influence your score in Algorithm Design.

     e.g. This part will be identified as a 'trick' because it is not for general purposes but written directly against a special case.



```
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)
```

  3. Algorithm Design (50%, X*/50)

     EITHER

     M1: Question1(X/50)

     a) Results [10]
        i. If your modified algorithm and your own algorithm generate an correct and optimal track for the Green Walker, you will be fully awarded, otherwise 0.
        ii. If your own algorithm exceeds time limit in large database but not in the small ones, you will receive a 7 here.

     b) Time complexity
        i. The 'Modified' Algorithm [10]

           You're given an answer code. If you improve the algorithm, you will receive a higher mark, since your timing is compared among all the participants.

ii. Your own Algorithm [25]

This algorithm has to use another method with another core code to solve M1, e.g. Brute force. Any code identified as the same as the given code will be penalized.

c) Space complexity

i. 4th night [1]

ii. 5th night [4]

OR

M2: Question2(X*/50):

a) Results [28]

i. Marks will be awarded range from 0 to 25

ii. A full mark (or the highest scored) solution may not be the optimal path, but close to.

iii. A ranking will be conducted for the results and you will be awarded marks based on your position in rankings. (Weight: 1st night 15%, 2nd night 15%, 3rd night 10%, 4th night 25%, 5th night 35%).

b) Time complexity [20]

i. If you are ranked first for the accuracy of results (after weighted), marks in this section will be fully awarded.

c) Space complexity

i. 4th night [1]

ii. 5th night [4]

Note

1.How to calculate the total score:

✧ IF both questions are solved, and each exceeds 40 marks, THEN total marks awarded in this part will be M1*65%+M2*65% (allowing for exceeding 50)

✧ IF both questions are solved, but at least one mark is under 40, THEN total marks awarded in this part will be the higher score between M1 and M2.

The way we evaluate complexity is designed for comparison between multiple programming languages. You may ignore the following explanation for complexities. However, **all you need to consider** is to improve your code so that it runs fast and takes up acceptable space.

2.How to evaluate time complexity

✧ Time Complexity will be automatically evaluated on lab computers (6th floor in the main building). Two codes in M1 and one code in M2 will be ranked separately.

✧ For each night case, run time limit is 1 hour. We will kill the program if it is still running.

✧ The time complexity is measured in 6 extra randomly-created databases (map and guidebook.) Your result will be penalized if the performance of your code in these extra databases is in large discrepancy with that in all the other given databases.

- ✧ The time complexity is evaluated on your code's ranking at absolute timing and comparative timing in the 6 extra databases.
- ✧ For M1, if any of the solution result is incorrect, you will be ranked the last. Plus, ranking in M1 will be curved according to a standard solution (given by the student committee). For M2, the solution result will not influence your scoring here.
- ✧ The absolute timing will be evaluated according to the ranking of the total timing of your program (how long it takes your program to finish processing all 6 extra databases.)
- ✧ The comparative timing will be evaluated according to the ranking of a special measurement: $r = t_2/t_1$, where t_1 denotes the physical timing of smaller-database night case (e.g. 100 trees and 40 visits), t_2 denotes the physical timing of a larger-database night case (e.g. 2000 trees and 700 visits), and r will be ranking from smallest to largest, as an indicator of complexity. This measurement will be repeated for several times and then averaged. Your code will be scored according to the grand rank. However, if the code uses tricks to intentionally enlarge the t_1, your result will be penalized.

3.How to evaluate space complexity

- ✧ Space Complexity will be automatically evaluated on lab computers (6th floor in the main building).
- ✧ The space complexity mark will be awarded for full marks if memory limit does not exceed; otherwise, you have to provide us a machine that can run the code, and your overall result will be penalized.

✧ Thesis Defense (40%, X/40)

1. Presentation Skills [X/25]
   a) Considering the judges, you will receive more credits if you are using plain language in addition to technical words. However, you should not forget this is still HACKATHON.
   b) Concise, precise, clear, understandable explanation of your algorithm(s). You may illustrate your algorithm(s) by printing out a page of flowchart or structured English to the audience and the judges. Prepare for questions from the audience. Judges may ask question at any time, and others only can ask question at the end of your presentation. You should notice that you are given a limited presentation time; however, any rushed and unclear explanation may lead to an adverse effect.
2. Idea [X*/15]
   a) Multiple algorithms
   i.e. You do not need to present any code, but you should explain its/their logic in detail.
   b) Further discussion
   i.e. You may take other factors into account, or alter some assumptions. You may raise questions and solve them. We expect a well-rounded answer.

c) Association with real life
i.e. This might be a simple Tree Visitor problem. However, there may be some similar problems happening around us. We are expecting a consideration of real life. Think of how this question inspire you, how to carry it in to real life, and how to make it more practical.

- **Extra Bonus**
- 71, 111, 111, 100, 32, 76, 117, 99, 107
- 72, 97, 118, 101, 32, 70, 117, 110

- Hint1: super easy
- Hint2: super super super easy
- Hint3: every IGCSE or A-Level CS student knows (or is supposed to know)
- Email us if you have cracked this puzzle, and you will get a ‘Helper’.
- Email Address: RDFZ_Hackathon@163.com