1. AUTHORSHIP: Describe who did the work on this lab. If you worked as a pair, did you use good pair-programming practices? Explain.

* Marima Andrew (Navigator), and Kyaw Hpone Myint (Driver). Kyaw did the coding, and Marima came up with the algorithm for the functions.

1. INITIAL DESIGN PLAN: What is a pseudocode design plan which meets the computational requirements of this lab?

|  |
| --- |
| Class crawler():    def \_\_init\_\_(self, current position, the map as a 2D array):  #initialize instance variables  def read\_map():  Pre: the filename need to be provided by the user  Post: return the map as a 2D array to the main function  Def check\_location():  Pre: current position on the map  Post: Check all four directions, and will move according to the following priority:  ***west>east>north>south***    Def move\_crawler():  Pre: need a function call to check\_location to check what is around current position  Post: - will move the treasure hunter current position according to the data provided by the check\_location, and will keep track of all the junctions where there are multiple possible path  Def count\_treasure():  ‘’’this will keep track of the amount of treasure found during the crawling process.””” |

1. SUMMARY: A brief summary description of the design and implementation, including how much your initial design plan evolved, the final result you achieved and the amount of time you spent as a programmer in accomplishing these results. This should be no more than two paragraphs.

* We found that our initial design is too simple, and it did not account for how to back track the treasure hunter back to the latest junction. We overcame this problem by implementing the stack data structure which can keep track of what position that treasure hunter has traversed during a path. We spent about 2 hours each day during the last week (total of about 12 hours) to finish the assignment.

1. IMPLEMENTATION: A list in bullet form of specifically what was accomplished including any challenges overcome and innovations that were not specifically required by the assignment.

* Exception handler that raises IOerror in case of a nonexistent file
* We print out the map after every move, where the position already traversed by the treasure hunter was indicated by “\*” sign. This made it easier for us to keep track of our treasure hunter, and also debug the program
* We listed the total number of move made during the crawling process
* We returned the total number of treasure found in the cave

1. TESTING: A list in bulleted form of all input values used for testing. Here you should be careful to select representative input cases, including both representative typical cases as well as extreme cases.

* We tested with the input files we provided. Each of the following lines should be entered in separate runs of the program
* “map1.txt”
* “map2.txt”
* “map3.txt”

1. FILES: A list in bulleted form of the names of all files submitted (source code and input, etc.)
   1. main.py
   2. Stack.py
   3. crawler.py
   4. map1.txt
   5. map2.txt
   6. map3.txt
2. ERRORS: A list in bulleted form of all known errors and deficiencies.

* We used the exception handler to take care of all the possible errors; so, the problem will raise an error instead of crashing

1. COMMENTS: A paragraph or so of your own comments on and reactions to the Lab.

* This lab was a bit difficult to finish. We had a hard time trying to come up with a working algorithm which can crawl the cave in the fashion we want. But overall this is a very entertaining lab, and it teaches us practical application of the stack data structure.

1. BIG-O: What is the big-O analysis of the method (or methods) which do the backtracking?

* The big of the program will increase as a function of input map size. So I would say the big O will be O(n), given that we will have same number of junctions with an increase dimension of the map. For example, if we have four paths radiating from the center (much like a + sign), and we increase the dimension by 2n in each side, then the time required to run the program will increase by a factor of 2.