TCP1101 Programming Fundamentals

Trimester 2, Session 2020/2021
Faculty of Computing & Informatics
Multimedia University

Assignment #2 (25%)

Let's Explore Mars!
Mission: Find Golds!

1. Introduction

The CyberScience Space Lab Inc. of which you work, had just detected gold on Mars. As the lead programmer of the lab, you are given the task to develop a simulation program that tests the control of the new Mars Rover in search of golds.

2. Deadline

This Assignment#2 must be submitted by **1st March 2020 (Monday of Week15), 10.00 a.m.** Submission to be done online.

You are given 2 hours of grace period; this should be sufficient for you to solve problems such as network congestion etc. Late submission penalty shall take effect starting at 12.00 p.m. of which 20% of the full marks shall be deducted; subsequence 20% of the full marks shall be deducted for each 24 hours (1 day) late.

Please note that submission of this Assignment#02 is **COMPULSORY** for this course.

Start your work as early as possible. Do not wait until last minute. If you start early, you can consult and get help from your lecturers or tutors when you face difficulties. You will rush if you do last minute work. **Your lecturers and tutors reserved their full right to reject your last-minute request for helps on problems related to earlier-taught lessons**.

3. Grouping

This assignment is to be done in a group of not more than 2 persons from the same toturial section. **STRICTLY NO COPYING** from other students in other groups or from any

other sources (Internet, books, etc.). **Plagiarism and cheating is an offence** in MMU, and you can be failed for this subject and be reported to the Faculty for action.

Please note that you may be called for an INTERVIEW to explain what you have done. If you cannot prove that you have done the Project on your own, you may be given a mark of ZERO.

Each member in a group may be given different marks in accordance to his/her contributions in the group.

4. Collaboration Policy

You are strongly encouraged to discuss, learn, and collaborate in a study group among your friends; the best way to learn is to teach each other and help each other understand the course materials.

However, you MUST NOT copy the work of each other, and risk being penalized for copying from each other.

One may ask, if you are doing the assignment together, how can you avoid having the same solutions and risk being penalized for copying from one another? The answer to this is to follow the rules below:

- You are highly encouraged to discuss with your friends about any problem that you may be facing.
- After your discussion, all forms of notes and solutions (be it written materials or things you entered the computers) MUST BE DESTROYED AND DELETED.
- Go do something which numbs your brain for an hour, such as playing with your pets, social media, or any other activities not related to your assignment.
- After an hour, you may proceed to do your assignment and solve your problems. If you cannot remember what you have discussed, it probably means you do not understand what you have discussed. What you submitted are expected to be your own original work rather than something which you copied during the discussion with your friends.
- Disclaimer: This rule does not imply that you are encouraged to play with social media, it is just an example. You should probably do something useful such as studying for a different subject etc.

5. Deliverables & Tips

The objective of this project is to test the skill of the student in problem solving using the C++ programming language.

The student is to create a small game that simulates the control of the Mars Rover.

Sample sessions of an actual such basic program is provided for your reference in the Figure in the next section of this document.

Take note that the program generates everything on the display. Those parts that required user's response (after the "=>" symbol) are to be typed by the user.

Take note that this is only the minimal basic requirement; you should add as much features as possible to your program and to make it as user-friendly as possible in order to get better marks.

Recommendation on how to develop the program:

a. Create a class named `Map` which would be used to create the following:
   - The actual Map of Mars, which is to be hidden from the user and will only be displayed to the user at the end of the game.
   - The Mars Rover Mapper, which keeps track of the cells in the map the Rover had already observed.

b. One possible minimal class declaration for `Map` is shown below:

```
class Map
{
   private:
      vector < vector<char> > map;
      int dimX, dimY;

   public:
      Map(int dimx, int dimy);
      void resize(int dimx, int dimy, char ch);
      void display();
      void setObject(int x, int y, char ch);
      char getObject(int x, int y);
      bool isEmpty(int x, int y);
      bool isHill(int x, int y);
      bool isTrap(int x, int y);
      bool isInsideMap(int x, int y);
      int getDimX();
      int getDimY();
}; //Map
```

c. Create a class named `Rover`, which would be used to keep track of the location and heading of the Rover. One possible minimal class declaration is shown below.

```
class Rover
{
   private:
      int x,y;
      Direction heading;
      char objectUnderRover;
      Map* p_mars; //a pointer to the map for Mars
      Map mapper;  //a map that keeps track of observed cells so far

   public:
      Rover();
      void land(Map& mars); //links a map of Mars to a Rover
      bool turnLeft();
      bool turnRight();
      bool move();
      void displayMapper();
      bool executeCommand(char command);
}; //Rover
```

d. The class declarations above are only the bare minimal, you may want to add member functions and data member such as for keeping track of golds collected etc.

e. The data type `Direction` can be defined using enumeration as shown below. Please refer to your textbook on how to use `enum`.

```
enum Direction
{
   north = 0,
   east = 1,
   south = 2,
   west = 3
};
```

f. As the Rover moves across the map, it would be able to read the three cells just in front of it.

g. There are hills on the map which prevent the Rover from moving to those cells.

h. There are traps on the map, if the Rover moves to any of those cells, the mission is failed.

i. You are free to design your scoring system and other type of cells, you may also introduce new challenges for the Rover.

The **GOAL** of the game is for the Rover to collect all the golds on Mars.

The Mars Rover assumed to be **ALWAYS** starts at the centre of the map which is **ALWAYS** an empty space, but the Rover's heading can start at any of the north, east, south or west direction.

You **MUST** design your map such that all the golds are reachable from the starting cell and that they can be collected by the Rover.

The **ALGORITHM TO GENERATE THE MAP** would constitute **AT LEAST ONE-THIRD OF THE TOTAL MARKS** of the assignment, all the cells must be randomly generated with good schemes. **FOCUS FIRST** on designing the algorithm that generates the map. Use your creativity!! Yes, programming is also a **CREATIVE PROCESS**.

For **MAP GENERATION**, you may want to make good use of the pseudo-random number generator from the C++ Standard Library. If we want to generate 100 characters with a 10% chance of getting 'X', 30% chance of getting 'Y' and 60% chance of getting 'Z', we can do that with a program such as the below:

```
#include <iostream>
#include <cstdlib>
#include <ctime>       //for time() in srand( time(NULL) );

using namespace std;

int main()
{
   char charlist[] = {'X',
                      'Y', 'Y', 'Y',
                      'Z', 'Z', 'Z', 'Z', 'Z', 'Z' };


   //seed for pseudo-random number generator
   unsigned int seed = time(NULL);

   srand(seed);

   int countX = 0, countY = 0, countZ = 0;
   for (int i=0; i<100; ++i)
   {
      int r = rand() % 10; //get a pseudo-random number in range 0 to 9

      char ch = charlist[r];

      cout << ch;
      if (ch=='X') ++countX;
      if (ch=='Y') ++countY;
      if (ch=='Z') ++countZ;
   }
   cout << endl;

   cout << "Number of X = " << countX << endl;
   cout << "Number of Y = " << countY << endl;
   cout << "Number of Z = " << countZ << endl;
}
```

The output can be shown in the below sample run:

```
C:\dev>demo_rand
ZZYYZZZYYZZZZYZZZYZZZZZXZYYZYZYZXZYZZZZZZZZZZZZZZ
YZXZXYZZYZZYZYYZZYZXZZZZYZYXXYZYZZZZZZZZXZYZZZZZZX
Number of X = 9
Number of Y = 23
Number of Z = 68

C:\dev>demo_rand
ZXZZXZZYZYYZYXZXZYYZZZYZYZXZZZXZZYYZZZYZYYZYZZZXYZ
XYYYYZYXZZZZZZZZZYZZZXYZZZXZYZZZZZZYXYYZYYYXXZYYZZ
Number of X = 14
Number of Y = 31
Number of Z = 55

C:\dev>demo_rand
ZZYZYYZYZZXYYXZZZZZXYYZZZZZXZZZXZZZZYZZZZZZXXYZYYZ
ZYZYZYXYZZZZZYZXZYZYZYYZYZZXZZYYZZZYXZYYYZXZYZYYZY
```

```
Number of X = 12
Number of Y = 32
Number of Z = 56

C:\dev>demo_rand
YZYZYZXZZXZXYZZZZYZYZZZZZZXZZXZZXZYZZZZYZZZZYYZYZZ
ZYZZZYZZZXZZZYYZYYYYZZZZZYZYYYYZXYYYZYYZYXYYZYYZZZ
Number of X = 9
Number of Y = 34
Number of Z = 57
```

A possible approach to generate the map is as follows:

Step 1: Populate the cells in the map with objects (i.e. represented using characters) using a desired probability.

Step 2: Change the cell at the centre of the map to be an empty space (since the Rover is always assumed to start at the centre).

Step 3: Do a partially random walk in the map starting at the centre of the map, with a random heading direction.

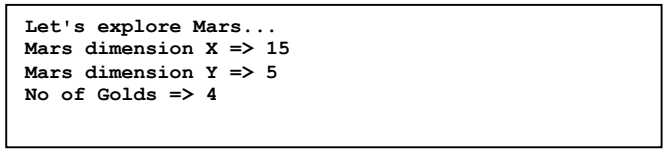
Step 4: Keep moving in the current direction and change the cells to be empty spaces as we move along that direction.

Step 5: Generate a pseudo-random number, and based on that number, decide whether to turn left, turn right, or continue in the same direction. For example, in order to have a 20% chance of turning left, 20% chance of turning right, and 60% chance of moving ahead, you can generate a pseudo-random number in the range from 0 to 9 and turn left if the number turns out to be 0 or 1, turn right if the number turns out to be 8 or 9, and stay in the same direction otherwise. If you reach the boundary of the map, just turn around.

Step 6: As you move along the map, generate a pseudo-random number, and decide to put a gold object or not. For example, generate a pseudo-random number in the range from 0 to 9, if the number is 0, you change the object at the cell to be a gold object, this means there is a 10% chance of dropping a gold when you move along the map.

Step 7: Repeat Step 4 onwards until you have dropped desired number of golds.

Sample Session of a Demo Program is shown below.



```
Let's explore Mars...
Mars dimension X => 15
Mars dimension Y => 5
No of Golds => 4
```

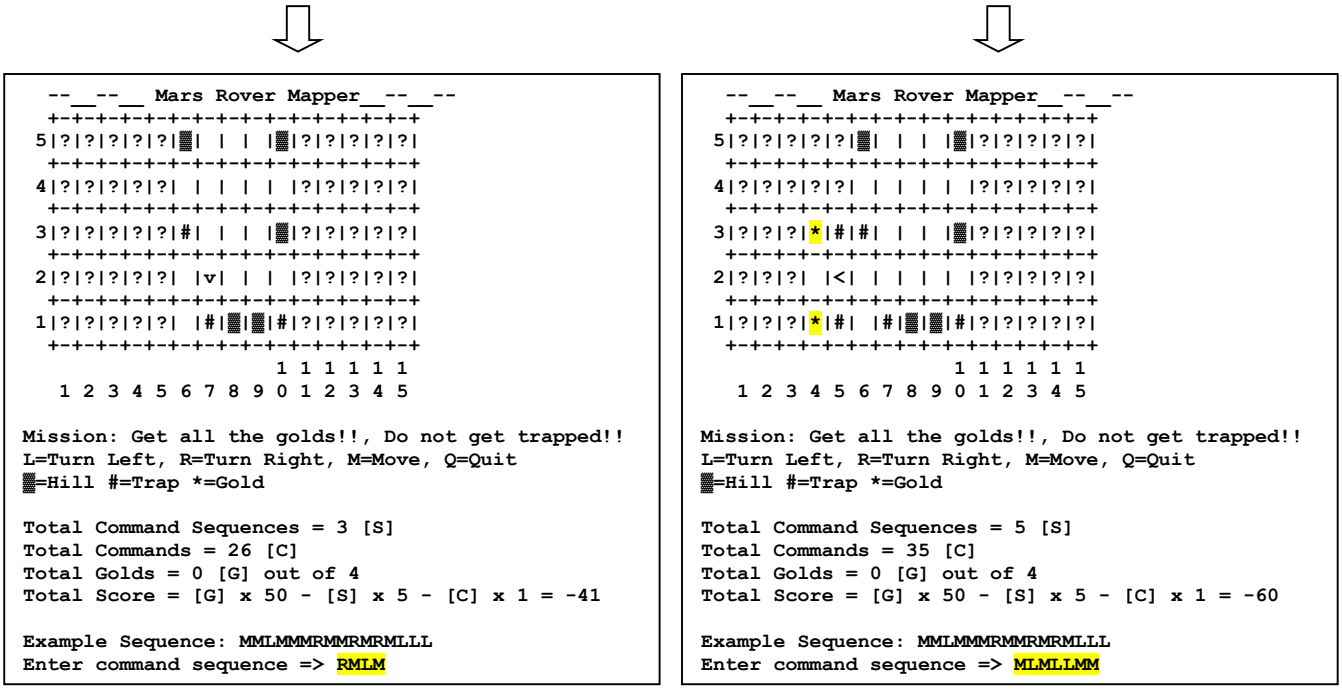
```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 5|?|?|?|?|?|?|?|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?|?| |?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 3|?|?|?|?|?|?|?|>| |?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 2|?|?|?|?|?|?|?|?| |?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 1|?|?|?|?|?|?|?|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                    1 1 1 1 1 1
   1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold

Total Command Sequences = 0 [S]
Total Commands = 0 [C]
Total Golds = 0 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = 0

Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => MLMLLMM
```

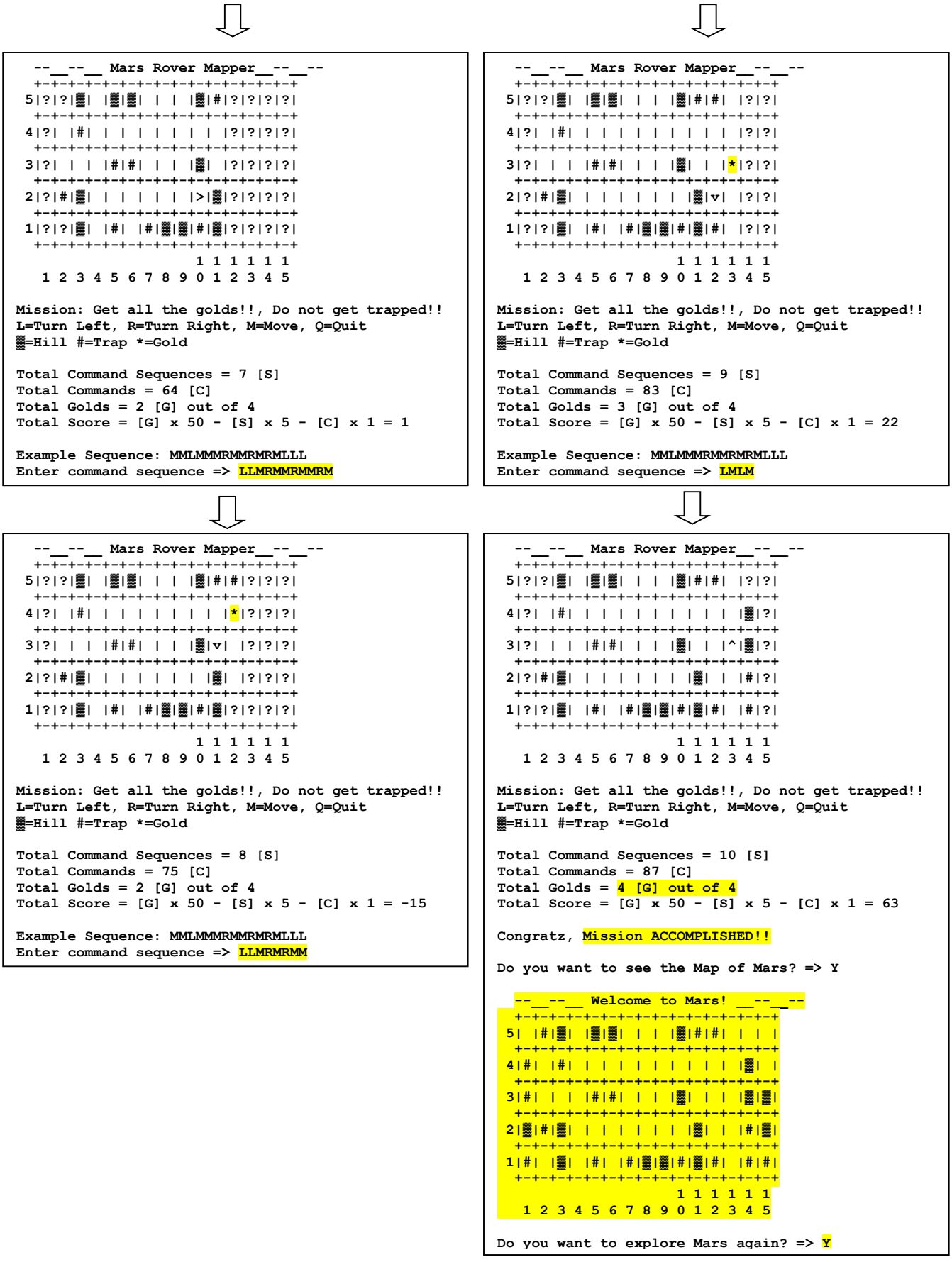
```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 5|?|?|?|?|?|?|?|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?|?| |?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 3|?|?|?|?|?|?|?|>| |?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 2|?|?|?|?|?|?|?|?| |?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 1|?|?|?|?|?|?|?|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                    1 1 1 1 1 1
   1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

Command = M ...
```

```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 5|?|?|?|?|?|?|?|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?|?| | |?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 3|?|?|?|?|?|?|?| |>|▒|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 2|?|?|?|?|?|?|?|?| | |?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 1|?|?|?|?|?|?|?|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                    1 1 1 1 1 1
   1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

Command = M [executed]
```

```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 5|?|?|?|?|?|?|?| | |▒|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?| | | |?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 3|?|?|?|?|?|?|?| | |▒|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 2|?|?|?|?|?|?|?| |v| |?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 1|?|?|?|?|?|?|?|▒|▒|#|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                    1 1 1 1 1 1
   1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold

Total Command Sequences = 1 [S]
Total Commands = 7 [C]
Total Golds = 0 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -12

Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => RMRMMMRM
```

```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 5|?|?|?|?|?|?|?| | |>|▒|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?| | | | |?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 3|?|?|?|?|?|?|?| | | |▒|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 2|?|?|?|?|?|?|?| | | | |?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 1|?|?|?|?|?|?|?|#|▒|▒|#|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                    1 1 1 1 1 1
   1 2 3 4 5 6 7 8 9 0 1 2 3 4 5

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold

Total Command Sequences = 2 [S]
Total Commands = 18 [C]
Total Golds = 0 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -28

Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => RRMMLMMM
```


--__--__ Mars Rover Mapper__--__--
1 1 1 1 1 1
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold
Total Command Sequences = 3 [S]
Total Commands = 26 [C]
Total Golds = 0 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -41
Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => RMLM
--__--__ Mars Rover Mapper__--__--
1 1 1 1 1 1
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold
Total Command Sequences = 5 [S]
Total Commands = 35 [C]
Total Golds = 0 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -60
Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => MLMLLMM
--__--__ Mars Rover Mapper__--__--
1 1 1 1 1 1
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold
Total Command Sequences = 4 [S]
Total Commands = 30 [C]
Total Golds = 0 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -50
Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => RRMLM
--__--__ Mars Rover Mapper__--__--
1 1 1 1 1 1
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold
Total Command Sequences = 6 [S]
Total Commands = 42 [C]
Total Golds = 2 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = 28
Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => LMLLMLMRMMMMMMRRMLMMLM


-- Mars Rover Mapper --
Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold
Total Command Sequences = 7 [S]
Total Commands = 64 [C]
Total Golds = 2 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = 1
Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => LLMRMMRMMRM
-- Mars Rover Mapper --
Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold
Total Command Sequences = 9 [S]
Total Commands = 83 [C]
Total Golds = 3 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = 22
Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => LMLM
-- Mars Rover Mapper --
Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold
Total Command Sequences = 8 [S]
Total Commands = 75 [C]
Total Golds = 2 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -15
Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => LLMRMRMM
-- Mars Rover Mapper --
Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold
Total Command Sequences = 10 [S]
Total Commands = 87 [C]
Total Golds = 4 [G] out of 4
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = 63
Congratz, Mission ACCOMPLISHED!!
Do you want to see the Map of Mars? => Y
-- Welcome to Mars! --
Do you want to explore Mars again? => Y

```
Let's explore Mars...
Mars dimension X => 8
Mars dimension Y => 4
No of Golds => 3
```

```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+
 3|?|?| |#|#|?|?|?|
  +-+-+-+-+-+-+-+-+
 2|?|?|?|^|?|?|?|?|
  +-+-+-+-+-+-+-+-+
 1|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+
   1 2 3 4 5 6 7 8

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold

Total Command Sequences = 0 [S]
Total Commands = 0 [C]
Total Golds = 0 [G] out of 3
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = 0

Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => M
```

```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+
 4|?|?| |#|#|?|?|?|
  +-+-+-+-+-+-+-+-+
 3|?|?| |@|#|?|?|?|
  +-+-+-+-+-+-+-+-+
 2|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+
 1|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+
   1 2 3 4 5 6 7 8

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold

Total Command Sequences = 1 [S]
Total Commands = 1 [C]
Total Golds = 0 [G] out of 3
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -6

TRAPPED!! Mission FAILED!!

Do you want to see the Map of Mars? => n

Do you want to explore Mars again? => y
```

```
Let's explore Mars...
Mars dimension X => 8
Mars dimension Y => 4
No of Golds => 3
```

```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+
 3|?|?|?|?|▒|?|?|?|
  +-+-+-+-+-+-+-+-+
 2|?|?|?|>|▒|?|?|?|
  +-+-+-+-+-+-+-+-+
 1|?|?|?|?|#|?|?|?|
  +-+-+-+-+-+-+-+-+
   1 2 3 4 5 6 7 8

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold

Total Command Sequences = 0 [S]
Total Commands = 0 [C]
Total Golds = 0 [G] out of 3
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = 0

Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => MMM
```

```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+
 3|?|?|?|?|▒|?|?|?|
  +-+-+-+-+-+-+-+-+
 2|?|?|?|>|▒|?|?|?|
  +-+-+-+-+-+-+-+-+
 1|?|?|?|?|#|?|?|?|
  +-+-+-+-+-+-+-+-+
   1 2 3 4 5 6 7 8

Command = M [failed to execute]
```

```
  --__--__ Mars Rover Mapper__--__--
  +-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?|?|
  +-+-+-+-+-+-+-+-+
 3|?|?|?|?|▒|?|?|?|
  +-+-+-+-+-+-+-+-+
 2|?|?|?|>|▒|?|?|?|
  +-+-+-+-+-+-+-+-+
 1|?|?|?|?|#|?|?|?|
  +-+-+-+-+-+-+-+-+
   1 2 3 4 5 6 7 8

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▒=Hill #=Trap *=Gold

Total Command Sequences = 1 [S]
Total Commands = 3 [C]
Total Golds = 0 [G] out of 3
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -8

Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => LQ
```

```
  --__--__ Mars Rover Mapper__--__--
   +-+-+-+-+-+-+-+-+
 4|?|?|?|?|?|?|?|?|
   +-+-+-+-+-+-+-+-+
 3|?|?|?|?|▓|?|?|?|
   +-+-+-+-+-+-+-+-+
 2|?|?|?|^|▓|?|?|?|
   +-+-+-+-+-+-+-+-+
 1|?|?|?|?|#|?|?|?|
   +-+-+-+-+-+-+-+-+
    1 2 3 4 5 6 7 8

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▓=Hill #=Trap *=Gold

Total Command Sequences = 2 [S]
Total Commands = 5 [C]
Total Golds = 0 [G] out of 3
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -15

QUITTED!! Mission FAILED!!

Do you want to see the Map of Mars? => n

Do you want to explore Mars again? => y
```

```
Let's explore Mars...
Mars dimension X => 8
Mars dimension Y => 4
No of Golds => 3
```

```
  --__--__ Mars Rover Mapper__--__--
   +-+-+-+-+-+
11|?|?|?|?|?|
   +-+-+-+-+-+
10|?|?|?|?|?|
   +-+-+-+-+-+
 9|?|?|?|?|?|
   +-+-+-+-+-+
 8|?|?|?|?|?|
   +-+-+-+-+-+
 7|▓|*| |?|?|
   +-+-+-+-+-+
 6|?|^|?|?|?|
   +-+-+-+-+-+
 5|?|?|?|?|?|
   +-+-+-+-+-+
 4|?|?|?|?|?|
   +-+-+-+-+-+
 3|?|?|?|?|?|
   +-+-+-+-+-+
 2|?|?|?|?|?|
   +-+-+-+-+-+
 1|?|?|?|?|?|
   +-+-+-+-+-+
    1 2 3 4 5

Mission: Get all the golds!!, Do not get trapped!!
L=Turn Left, R=Turn Right, M=Move, Q=Quit
▓=Hill #=Trap *=Gold

Total Command Sequences = 0 [S]
Total Commands = 0 [C]
Total Golds = 0 [G] out of 3
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = 0

Example Sequence: MMLMMMRMMRMRMLLL
Enter command sequence => q
```

```
Total Command Sequences = 1 [S]
Total Commands = 1 [C]
Total Golds = 0 [G] out of 3
Total Score = [G] x 50 - [S] x 5 - [C] x 1 = -6

QUITTED!! Mission FAILED!!

Do you want to see the Map of Mars? => n

Do you want to explore Mars again? => n

Good Bye from Mars!
```

6. Evaluation Criteria

In order for you to get better marks, the following are given for your consideration.

a. Efficiency

Your program should strive to be as computational efficient as possible.

b. Accuracy and Robustness (Error-free)

To make the program more usable, the program should handle special cases such as an accidental invalid input from the user. If the user enters the wrong things, you should allow the user to enter another input until it is a valid response.

c. Modularity of program (good and proper usage of functions and classes)

Each of your functions should NEVER be too long, always try to break down the problem into smaller functions.

A rule of thumb is that if you cannot draw the flowchart of a function in one A4 paper, then it is probably too big and should be broken into smaller ones.

Another rule of thumb is that if your function is more than two screens long, it is probably too long and should be broken into smaller ones.

d. Code Clarity

Codes should be written with clarity in mind and the purposes of codes should be self-explanatory. It is more preferrable to write codes that are clear and easy to understand without further code comments, than to flood the codes with too many reductant comments. Thus, code comments should be used in economical and appropriate manner, and not over-commenting the obvious.

e. *Other Evaluation Criteria*

Please refer to the Project Mark Sheet on the detailed marks allocation.

7. What to Submit?

7.1. Written Report

You are to prepare a written report that introduces and describes your program. You are free to design your report in any format you like as long as it is clear, neat, and interesting. Nonetheless, it should contain at least the following contents:

a) A Cover Page

It should at least contain your ID, name, Tutorial Section, and your tutor's name. A sample is included at the end of this document.

b) Introduction

Briefly introduces the purpose of your program and what your program can do.

c) User Manual / Instructions

Instructions on how to use your program, what to look for and what special feature your program has.

d) Screen Shots

Screen shots of the major screen output of your system. This is very important so that the instructor can remember the program that you have shown him/her earlier during the demo session.

e) Project Submission Declaration

Please sign the Project Submission Declaration Form at the end of this document, scan it and include it to the softcopy of your report.

7.2. Source Code

You also need to provide the source code of the program that you have written. It is your full responsibility to make sure that source code is error-free. Your code must be able to be compiled with **MinGW/g++ C++** compiler under Ms. **Windows** environment. Source codes that only work on other compilers will **NOT BE ACCEPTED**.

8. Submission Instruction

For all your source code files, insert the following information at the top of the file:

```
/*****************************************
Course : TCP1101 Programming Fundamentals
Session: Trimester 2, 2020/21
Assignment: 2
Lecture Section : TC1V
Tutorial Section: TTxy

Name of Student #1 : YOUR OFFICIAL CAMSYS NAME
ID of Student #1   : YOUR ID
Email of Student #1:
Phone of Student #1:
Name of Student #2 : YOUR OFFICIAL CAMSYS NAME
ID of Student #2   : YOUR ID
Email of Student #2:
Phone of Student #2:
*****************************************/
```

You are fully responsible to ensure your source code is bug-free and error-free. Your code must be able to be compiled and run without error on MinGW/g++ compiler under MS Windows environment.

You need to submit source code of your program and the report online. Instructions to be published later.

Assignment #2 Submission Declaration

TCP1101 Programming Fundamentals
Trimester 2, Session 2020/2021

To be Filled by Each Student

Name			
ID			
Lecture Section	TC1V	**Tutorial Section**	TT
Names and IDs of students which I have discussed regarding this assignment			
Number of hours I spent in doing this assignment			

I hereby declare that this assignment represents the work done by myself. I declare that no part of my work has been copied from other persons or by means of professional assistance.

I hereby declare, and I fully understood that, if I have copied any parts from any other persons OR if any other persons have copied from my work, I will get a mark of ZERO and I am fully responsible to ensure this does not happen.

I also hereby declare that every works or materials/codes that I may have borrowed, copied or modified from other sources are properly acknowledged in the report and the sources are listed in the ACKNOWLEGEMENT or REFERENCES section. I will take full responsibility and understood that I would get a mark of ZERO if I failed to make proper acknowledgement.

Signature: …….....…… Date: .....

Note: No work shall be accepted unless it is accompanied by this submission declaration with all sections completed and it has been signed and dated. Please scan the document and include this to the softcopy of your report.

FACULTY OF COMPUTING AND INFORMATICS

TCP1101 PROGRAMMING FUNDAMENTALS

TRIMESTER 2 2020/2021

Assignment #2
Report

Lecture Section : TC1V
Tutorial Section : TT__

INSTRUCTOR'S NAME:

Student ID	Name	Email Address	Phone No.

Assignment #2 Mark Sheet

TCP1101 Programming Fundamentals
Trimester 2, Session 2020/2021

To be filled by Examiner. This is for your reference only.

	Max	Actual Marks
1. Report (10%)		
a. Formatting and Style	2	
b. Language (Spelling, Grammar)	2	
c. Presentation Clarity	2	
b. Screenshots	2	
e. User Guide / Instructions to Use the Program	2	
2. Coding (50%)		
a. Error Checking Features	10	
b. Good Coding Style (indentation, spacing, consistent identifier naming convention etc.)	10	
c. Code Clarity (codes are written with clarity in mind and the purposes of codes are self-explanatory, code comments are used in economical and appropriate manner, and not over-commenting the obvious)	10	
d. Code Modularity (usage of functions)	10	
e. Code Efficiency and Strategy	10	
3. Program Execution (30%)		
(If the program cannot be run, even if it can be compiled, 0 mark will be given for this section)		
a. General Appearance of Program (display)	6	
b. Accuracy (program works and runs correctly)	6	
c. Usability (interaction with user, user-friendliness)	6	
d. Performs All Required Features	6	
e. Error-free During Runtime	6	
4. Additional Features (10%)		
a. Additional special features or any other significant contributions	10	
TOTAL	100	