TASK 4.1.1: "MYSTERIOUS CONTINENTS"

A MAP is a 48 by 16 rectangle of COORDINATES. Two coordinates are CONNECTED if they are neighbours either in south-north or in east-west direction. Initially each coordinate is only known to be either WATER (W) or GROUND (G).

```
There are four GROUND TYPES (GT):  G, M, P, and C.
And there are four WATER TYPES  (WT):  W, O, B, and L.
It is assumed that outside the map there is OCEAN (O).
```

There are certain geographic rules for changing the type of a coordinate (RELABELING). It may become a:

```
- MOUNTAIN  (M): If a GT is connected to 4 other GT.
- PENINSULA (P): If a GT is connected to 3 WT,
                                  or to 2 WT and at least 1 P,
                                  or to 1 WT and at least 2 P.
- COASTLINE (C): If a GT is not M and not P.
- OCEAN     (O): If a WT is connected to at least one O.
- BAY       (B): If an O is connected
                                 to at least 2 B  and at most  one O,
                              or to         1 B  and at least  2 GT,
                              or to at least 2 GT and at least one O.
- LAKE      (L): If a W remains unchanged till no other relabeling is
                 possible any more.
```

It may happen, that after a certain coordinate has been relabeled, it can be relabeled once again later, because the types of some neighbours have changed in the meantime.
A map is EXPLORED if no relabeling is possible any more.

PROBLEM STATEMENT

Implement a program which does the following in that order:

1. Read a map of an unknown continent from an ASCII input file and display it on the screen, together with the initial coordinate type statistics, as shown in Example-1.
2. Explore the map and relabel the coordinates correctly with M, P, C, O, B, or L according to the geographic rules.
3. Display the explored map on the screen, with the final coordinate type statistics, as shown in Example-2.
4. Write a screen copy showing the explored map and the final coordinate type statistics to an ASCII output file.

TECHNICAL CONSTRAINTS

```
Constraint-1: Put your solution program into an ASCII text file named
              "C:\IOI\DAY-1\411-PROG.xxx". Extension .xxx is:
              - .BAS for BASIC programs, .C   for C      programs,
              - .LCN for LOGO  programs, .PAS for PASCAL programs.
Constraint-2: The name of the ASCII input file for reading an unknown
              map from must be "C:\IOI\DAY-1\411-MAP.IN".
```

```
Constraint-3: The name of the ASCII output file for writing explored
              map and statistics to must be "C:\IOI\DAY-1\411-MAP.OU".

EXAMPLE(S)
==========
Example-1: The screen display, including initial statistics, of the
       unknown map in file "C:\IOI\DAY-1\411-MAP.IN" should look like:
WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWGGGGGGWWWWWWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWGGWWGGWWWWWWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWGGGWGGWWWWWWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWGGWWGGWWWGGGWGWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWGGGGGGGGGGGGGWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWGGGWWWGGWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWGGGWWWGGWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWGGGGWWGGWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWWGGWWWGGWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWWWGWWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWWWGWWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWWGGGWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
WWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWWW
MYSTERIOUS: G=61 W=707 ALL=768

Example-2: The screen display of the explored map, including final
   statistics and the file "C:\IOI\DAY-1\411-MAP.OU" should look like:
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOCCCCCCOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOCCLLCCOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOCMPLCCOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOCCLLCCBBBCCCBPOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOBCCCCCCCCMCCCCBOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOBCMCBBBCCOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOCMCBOOCCOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOCMMPOOCCOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOBCCBOOCCOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOBPBOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOBPBOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOPPPOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
EXPLORED: P=8 C=47 M=6 O=685 B=17 L=5 ALL=768


SAMPLE FILES
============
We provided these correct example files for your convenience:
"C:\IOI\DAY-1\411-MAP.IN" and "C:\IOI\DAY-1\411-MAP.OU".

WARNING: Successful execution of your program with Example-1 above
does not necessarily guarantee that your program is correct !!!


CREDITS
```

```
=======
Read from a file and display unknown map correctly .........  5 points
All Mountains correctly relabeled with M .................. 10 points
All Peninsulas correctly relabeled with P ................. 20 points
All Coastlines correctly relabeled with C .................  5 points
All Ocean correctly relabeled with O ...................... 10 points
All Bays correctly relabeled with B ....................... 20 points
All Lakes correctly relabeled with M ......................  5 points
Initial Statistics correct ................................  5 points
Final Statistics correct .................................. 10 points
Structure of output file correct ..........................  5 points
Technical constraints completely obeyed ...................  5 points
----------------------------------------------------------------------
                                                 maximal 100 points
```

TASK 4.1.2: "A MAZING WORKSHOP"

A MAZE completely covers an AREA of N times M squares. It consists of many WALL squares and of many SPACE squares, the latter of which include one ENTRY square and one TREASURE square.

A PATH is a sequence of adjacent space squares (bounded by walls) from the entry to a dead end, we refer to as an ENDPOINT. The LENGTH of a path is the number of squares it covers, including entry and endpoint.

The maze must be such that paths may fork but do not join, so for example no two paths can have the same endpoint. The entry is located somewhere at the top of the maze. The treasure is positioned at the endpoint of a path with maximal length.

The N times M area should be covered with paths as much as possible. It is nice to watch a maze growing over an area while it is computed. Because the algorithm is too fast for the eye, a DELAY TIME after each drawn square is necessary.

PROBLEM STATEMENT

Implement the following set of TOOLS dealing with mazes. The tools should be executable in any order and repetition through a main menue:

```
Tool-1: Set the main maze parameters N and M interactively.
Tool-2: Set a DELAY TIME interactively.
Tool-3: Compute a new correct maze basically using a random
        generator and display the maze while it is growing.
Tool-4: Write a generated maze and its size parameters to an
        ASCII text file, exactly as it is shown in Example-2.
Tool-5: Read an unknown maze from an ASCII text file
        and highlight the path from entry to treasure.
```

TECHNICAL CONSTRAINTS

```
Constraint-1: Represent each square by a two-character string:
              - walls by two times ASCII character #219 ...... "[["
              - paths and entry by two blanks ............... "  "
              - treasure by T and blank ..................... "T "
              - highlighted paths by full-stop and blank ..... ". "
Constraint-2: N and M must be greater than 2 and not larger than 20.
Constraint-3: Put your solution program into an ASCII text file named
              "C:\IOI\DAY-1\412-PROG.xxx". Extension .xxx is:
              - .BAS for BASIC programs, .C   for C      programs,
              - .LCN for LOGO  programs, .PAS for PASCAL programs.
Constraint-4: The name of the ASCII text file for reading and writing
              mazes must be "C:\IOI\DAY-1\412-MAZE.IO".
```

EXAMPLE(S)

```
Example-1: A screen display of sample file "C:\IOI\DAY-1\412-MAZ1.IO"
           by Tool-5 should look like:
```

```
N = 10, M = 8, DELAY TIME = 100
[[[[[[[[[[[[. [[[[[[
[[[[[[    . . [[  [[
[[[[    [[. [[    [[
[[    [[  . . [[  [[
[[  [[    [[. .   [[
[[[[    [[  [[. [[[[
[[    [[T . . .   [[
[[[[[[[[[[[[[[[[[[[[
LENGTH = 13
```

Example-2: The same maze's file output by Tool-4 should look like:

```
10   8
[[[[[[[[[[[[  [[[[[[
[[[[[[        [[  [[
[[[[    [[  [[    [[
[[    [[      [[  [[
[[  [[    [[      [[
[[[[    [[  [[  [[[[
[[    [[T         [[
[[[[[[[[[[[[[[[[[[[[
```

SAMPLE FILES
============
We provided these correct example files for your convenience:
"C:\IOI\DAY-1\412-MAZ1.IO" and "C:\IOI\DAY-1\412-MAZ2.IO".

WARNING: Successful execution of your program with these examples
does not necessarily guarantee that your program is correct !!!

CREDITS
=======

```
Main menue with all tools available ........................  5 points
Tools available in any order and repetition ................ 10 points
Tool-1 enables setting N and M .............................  5 points
Tool-2 enables setting DELAY TIME ..........................  5 points
Tool-3 computes structurally correct mazes ................. 30 points
Tool-3 displays the maze while it is growing ............... 10 points
Tool-4 writes maze to a file exactly as in example-2 .......  5 points
Tool-5 reads unknown maze and highlights longest path ...... 20 points
Technical constraints completely obeyed .................... 10 points
----------------------------------------------------------------------
                                                  maximal 100 points
```

Problem Chosen for the first session (5 hours)

***TASK 4.1.3 "ISLANDS IN THE SEA"
===============================

The SEA is represented by an N times N grid. Each ISLAND is a "*" on that grid. The task is to reconstruct a MAP of islands only from some CODED INFORMATION about the horizontal and vertical distribution of the islands. To illustrate this code, consider the following map:

```
*   * *       1 2
  * * *   *   3 1
*   *   *     1 1 1
  * * * * *   5
* *   *   *   2 1 1
      *       1

1 1 4 2 2 1
1 2   3   2
1
```

The numbers on the right of each row represent the order and size of the groups of islands in that rows. For example, "1 2" in the first row means that this row contains a group of one island followed by a group of two islands; with sea of arbitrary length to the left and right of each island group. Similarly, the sequence "1 1 1" below the first column means that this column contains three groups with one island each, etc.

PROBLEM STATEMENT
=================

Implement a program which repeats the following steps until a given input file containing several information blocks has been read completely:

1. Read the next information block from an ASCII input file
   (for the data structure of that file see also the examples below)
   and display it on the screen.
   Each information block consists of the size of the square grid, followed by the row constraints and the column constraints. Each constraint for a single row or column appears on a single line as a sequence of numbers separated by spaces and terminated by 0.
2. Reconstruct the map (or all of the maps, if more then one solution is possible, see Example-4) and display it/them on the screen.
3. Write the map(s) to the end of an ASCII output file. Each blank must be represented by a pair of spaces. Each island should be represented by a '*' followed by a space. Different maps satisfying the same constraints should be separated by a blank line. If there is no map satisfying the constraints, indicate it by a line saying "no map". The solutions to the different information blocks must be separated by a line saying "next problem".

TECHNICAL CONSTRAINTS
=====================

```
Constraint-1: N must be not less than 1 and not larger than 8.
Constraint-2: Put your solution program into an ASCII text file named
              "C:\IOI\DAY-1\413-PROG.xxx". Extension .xxx is:
              - .BAS for BASIC programs, .C   for C      programs,
              - .LCN for LOGO  programs, .PAS for PASCAL programs.
Constraint-3: The name of the ASCII input file for reading the coded
              information from must be "C:\IOI\DAY-1\413-SEAS.IN".
Constraint-4: The name of the ASCII output file for writing the
              map(s) to must be "C:\IOI\DAY-1\413-SEAS.OU".

EXAMPLE(S)
==========
6            Example-1 (the problem above): 6 is the size of the grid.
1 2 0        <-- The start of the first line constraint
3 1 0
1 1 1 0
5 0
2 1 1 0
1 0
1 1 1 0      <-- The start of the first column constraint
1 2 0
4 0
2 3 0
2 0
1 2 0

4            Example-2. Solution: columns: 1 2 3 4
0                                   row 1:
1 0                                 row 2:     *
2 0                                 row 3:   * *
0                                   row 4:
0
1 0
2 0
0

2            Example-3. Note that there is no map
0                       satisfying the constraints.
0
2 0
2 0

2            Example-4. Note that there are two different maps
1 0                     satisfying the constraints.
1 0
1 0
1 0

SAMPLE FILES
============
We provided these correct example files for your convenience:
"C:\IOI\DAY-1\413-SEAS.IN" and "C:\IOI\DAY-1\413-SEAS.OU".

WARNING: Successful execution of your program with these examples
does not necessarily guarantee that your program is correct !!!

CREDITS
=======
Read an information block from
the input file and display it ..............................  5 points
Process all information blocks one by one
```

```
until the input file is read completely ................... 10 points
Reconstruct one map for each information
block (if it has a solution) and display it ............... 35 points
Write the solution map to the output file .................  5 points
Reconstruct all possible maps (if there
are several solutions) and display them ................... 20 points
Write all solution maps correctly
separated to the output file .............................. 10 points
Identify information blocks having no solution ............  5 points
Technical constraints completely obeyed ................... 10 points
----------------------------------------------------------------------
                                                 maximal 100 points
```