Game 1024

Description (courtesy Serra Canca)

This is a game that is also available on the website http://1024game.org. **Play it to get familiar with it!**

Given a board of squares where some have numbers, and the rest are empty...

- You can use the arrow keys to move the numbered squares in any direction (left, right, up, and down.)
- All numbered squares will move in the direction that you select, at the same time, as far as they can go without taking over the space of another square that has a value.
    - Every time you move these squares, a random, empty square will gain the value of 1.
- After moving them, any pair of squares that have the ***same value***, in the ***selected direction*** that are ***next to each other*** will be ***merged*** (the values of the two squares will be added together and will become a single square).
    - The squares do not increase in size-- you will gain an empty square after a merge.
    - If you happen to have more than two squares of the same value in the same direction that you are moving in and they are all next to each other, the equal pairs closest to the edge of the direction you had selected will merge first.

      When there are more than two identical values in adjacent cells (omitting empty cells), then the cells in the given direction have a higher precedence. For example, given a row in a 4-column board, where x is a space, as follows.
      2 x 2 2
      If we press LEFT, the leftmost two identical value cells (ignoring empty cell in between) are merged and we get
      4 2 x x
      If we press RIGHT, the rightmost two identical value cells are merged and we get
      x x 2 4

- You win if any of the squares have a value of 1024 at any point, and you lose if there are no more empty squares!
- You lose if you are not able to merge anymore and all squares are taken up, and you only gain a 1 if the direction you have selected moves any of the squares!

Tasks

Task A (20 points)
Define constructors, destructor, and method print for Board.cpp.

- Implement constructors, we set max to be 0 and the target to be 32. Also, set numRows and numCols, initialize panel as a two-dimensional array of numRows rows and numCols.
    - For default constructor, set numRows and numCols to be 3, panel is two-dimensional array with 3 rows and 3 columns, set max to be 0 and target to be 32.
    - For Board(int m), set numRows to be m and numCols to be m if m is >= 1, otherwise, set numRows and numCols to be 3. Set panel to be a two-dimensional array with numRows rows and numCols columns, set max to be 0 and target to be 32.
    - For Board(int m, int n), set numRows by m and numCols by n if both m and n are >= 1, otherwise, set numRows and numCols to be 3. Set panel to be a two-dimensional array with numRows rows and numCols columns, set max to be 0 and target to be 32.
- Define destructor, release the memory space pointed by panel and remember to handle dangling pointer problem.
- Define print method, which prints panel as a tabular format, each number in a cell with 4-character wide, if the number is 0, do not print it out. See sample output in the project.

In this task, only submit Board.cpp to gradescope for grading. However, you should keep Board.hpp and Board.cpp in the same directory of local computer for compilation.

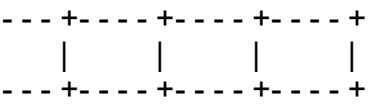
After finishing this task, your code does not run yet, use g++ -std=c++11 -c Board.cpp to check for compilation errors.

To test Task A, define TestBoard.cpp in the same directory as Board.hpp and Board.cpp, with the following contents.

```
#include "Board.hpp"

int main()
{
   Board game; //create a default Board object game
   game.print(); //test print method of game

   Board game2(4);
     //create a Board object with four rows
     //and four columns
   game2.print(); //test print methd of game2

   Board game3(3, 4);
     //create a Board object with three rows
     //and four columns
   game3.print(); //test print methd of game3

   return 0;
}
```

The expected output prints out a 3x3, 4x4, and 3x4 panel.

```
+- - - - +- - - - +- - - - +
|       |       |       |
+- - - - +- - - - +- - - - +
|       |       |       |
+- - - - +- - - - +- - - - +
|       |       |       |
+- - - - +- - - - +- - - - +
+- - - - +- - - - +- - - - +- - - - +
|       |       |       |       |
+- - - - +- - - - +- - - - +- - - - +
|       |       |       |       |
+- - - - +- - - - +- - - - +- - - - +
|       |       |       |       |
+- - - - +- - - - +- - - - +- - - - +
+- - - - +- - - - +- - - - +- - - - +
|       |       |       |       |
+- - - - +- - - - +- - - - +- - - - +
|       |       |       |       |
```



Submit only Board.cpp.

Pay attention to the following,

- Name source code Board.cpp. Note that board.cpp is not the same as Board.cpp since Linux is case-sensitive.
- Do not include main function Board.cpp. You should enclose main function in TestBoard.cpp, but no need to submit TestBoard.cpp to gradescope.
-

Task B (20 points)
Define code to select a random cell with value 1. That is, define selectRandomCell method of Board.cpp. Students may have questions on the parameters in the header of

void selectRandomCell(int& row, int& col)

Parameters row and col are row- and column-index of the chosen random empty cell. Use int& to pass by reference since we need to return both row and column. In our program, the returned row- and column-index are not used. When we improve our code later on, say, set text color in the newly chosen random cell to be red, we will need such information.

- Comments all constructors in the submission of Task A. But keep destructor ~Board() and method print and any helper function for print.
  Reason: gradescope autograde scripts provide a correct version of constructors, in case your constructors are incorrect in Task A, methods selectRandomCell and noAdjacentSameValue in your Task B can still be graded separately.
- Define method noAdjacentSameValue, which checks whether the game can be over or not. That is, if all cells are filled up (no empty cell, which represents zero, in the panel) and no two adjacent cells have the same value, return true, otherwise, return false.
  Note that, this method is called **after** finding every cell in the board is filled up. We can declare method noAdjacentSameValue as private, so the user of Board class cannot call this method directly, only selectRandomCell method can call method noAdjacentSameValue.
  If we detect that every cell is filled up, we only need to check whether there are adjacent same values in horizontal and/or vertical direction and current max < target to conclude that game is over.
  Hints: check whether there are adjacent same values in horizontal or

vertical direction. That is,

- In horizontal direction, for each row, check whether there are one pair of identical-value cells, if there is one, return false.

		…	

For a row with row index i,
Column index runs from 0 to numCols -1.
The first adjacent pair has column indices 0 and 1.
The second adjacent pair has column indices 1 and 2.
…
The last adjacent pair has column indices numCols-2 and numCols-1.
Since each adjacent pair (in horizontal direction) can be represented by left index and right index and these two indices are not independent (know one, know the other). That is, if the left index is j, then the right index is j+1.
We only need to represent either left or right index, but not both.
Without loss of generality, we choose left indices of each adjacent pair.
for a row with row index i,
for (int j = 0; j < numCols – 1; j++)
    Compare panel[i][j] and panel[i][j+1] have the same value or not

Pseudocode to check adjacent pairs in every row.

```
for (int i = 0; i < numRows; i++)
    for (int j = 0; j < numCols – 1; j++)
        Compare panel[i][j] and panel[i][j+1] have the same value or not.
        if panel[i][j] has the same value as of panel[i][j+1],
        then there are adjacent same value cells, return false,
        meaning the game is not over yet.
        Otherwise (panel[i][j] is not the same as panel[i][j+1]),
        There is no conclusion yet, we still need to check the rest cells,
        panel[i][j] does not equal to panel[i][j+1] only means
        this specific adjacent pair do not share the same value,
        but there might be same value adjacent pairs
        somewhere else. Warning: do not return true in the else part.
```

- In vertical direction, for each column, check whether there are one pair of identical-value cells, if there is one, return false. Note that the two adjacent cell in vertical direction are represented by panel[i][j] and panel[i+1][j], where i is row index ranging from 0 to numRows-2 and j is

column index ranging from 0 to numCols-1.

Pseudocode to check adjacent pairs in every row.

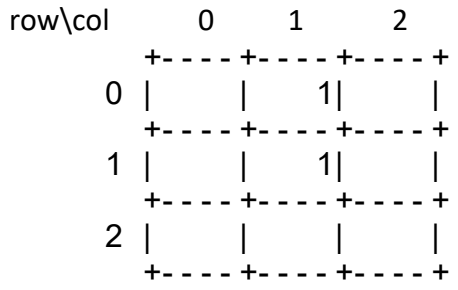
```
for (int j = 0; j < numCols; j++)
    for (int i = 0; i < numRows – 1; i++)
        Compare panel[i][j] and panel[i+1][j] have the same value or not.
        if panel[i][j] has the same value as of panel[i+1][j],
        then there are adjacent same value cells, return false,
        meaning the game is not over yet.
        Otherwise (panel[i][j] is not the same as panel[i+1][j]),
        There is no conclusion yet, we still need to wait to check the rest
cells,
        panel[i][j] does not equal to panel[i+1][j] only means
        this specific adjacent pair do not share the same value,
        but there might be same value adjacent pairs
        somewhere else. Warning: do not return true in the else part.
```

- If there is no return in either (1) or (2), then it means there is no adjacent same value pairs in any row and any column, if the current max is smaller than the target, return true (means the game is over).

- Define method selectRandomCell, which do the following
- Tally total number of empty cells (cells with number zero) in the panel.
- If there is no empty cell, if no adjacent cells with same value and max is smaller than target, print “Game over. Try again.” and exit the program.
- If there is at least one empty cell, put the cell’s location, represented by row and column, into a dynamically allocated array or a vector (will be introduced later) of integers. Challenge: how to represent the cell’s row and column information by an int? Think about how we label the seat numbers in a classroom with 3 rows and 4 columns. In the following illustration, row index goes from 0 to 2 vertically, while column index goes from 0 to 3 horizontally. Row 0 and column 0 is represented by 0, while the cell with row index 1 and column index 2 is represented by 6.

```
row\col     0      1      2      3
          +- - - +- - - +- - - +- - - +
      0 |   0 |   1 |   2 |   3 |
          +- - - +- - - +- - - +- - - +
      1 |   4 |   5 |   6 |   7 |
          +- - - +- - - +- - - +- - - +
      2 |   8 |   9 |  10 |  11 |
          +- - - +- - - +- - - +- - - +
```

For example, suppose we have the following board layout.



Then the number of empty cells is 7. We create an array with 7 integers.
Start label empty seats from zero. From first row, move from left to right, afterwards, go to the second row, move from left to right, and so on.
The first empty cell with row index 0 and column index 0 is represented by 0.
The second empty cell with row index 0 and column index 2 is represented by 2.
The third empty cell with row index 1 and column index 0 is represented by 3.
The fourth empty cell with row index 1 and column index 2 is represented by 5.
Similarly, the last three empty cells (in the last row) are presented by 6, 7, and 8, respectively.
That is, we have an array [0, 2, 3, 5, 6, 7, 8].

If you cannot represent the cell's row and column by an integer, here is another approach. Define a struct to represent row and column, then use an array or vector of this struct (a struct is a type) to record row and column information of each empty cell. You can use vector of this struct as well to avoid dynamic allocated memory processing (new, then delete and handle dangling pointer problem).

- Select a random element from the above array or vector, then convert the int back to its original row and column, set the corresponding cell by 1.
  For example, suppose a random integer 23 is generated, then 23 % 7 returns 2, where 7 is the size of the array. The remainder of any int divided by 7 falls in [0, 6], which is a range of valid indices of an array of size 7.

  The element indexed at 2 in the above array is 3, which corresponds to an empty cell with row index 1 and column index 0. You need to convert 3 back to row index 1 and column index 0 in a 3 x 3 panel. Afterwards, put number 1 to this cell.

- Print the updated panel with the added cell.

- With the added cell, if every cell is filled up and no adjacent cells with same values and max is smaller than target, print “Game over. Try again.” Then exit the program.

In this task, only submit Board.cpp to gradescope for grading. However, you should keep Board.hpp and Board.cpp in the same directory of local computer for compilation.

After finishing this task, your Board.cpp does not run yet, use g++ -std=c++11 -c Board.cpp to check compilation errors.

Submit only Board.cpp.

Task C (40 points)
Define code to press UP, DOWN, LEFT, RIGHT. That is, define pressUp, pressDown, pressLeft and pressRight methods of Board.cpp.

Note that selectRandomCell method is called in these above methods. In case your version of selectRandomCell in Task B has errors, you would fail Task C. To avoid that scenario, we provide a corrected version of selectRandomCell in gradescope scripts. As a result, do not include your definition of selectRandomCell and constructors/destructor in submission of Task C.

- From the submission of Task B, which includes print and selectRandomCell methods, add definition of pressLeft, pressRight, pressDown, and pressUp methods.
- When you define pressLeft, pressRight, pressDown, and pressUp methods, remember to call selectRandomCell method in the end. This is because, after an arrow key is pressed, a random empty cell is selected and number 1 is put in that cell, these steps are defined in selectRandomCell.
- Before submission, comment the definition of selectRandomCell since a correct version of it will be provided by gradescope scripts. If you include the definition of selectRandomCell method in submission of Task C, compiler would complain that that method is redefined.
  In short, only include codes for pressLeft, pressRight, pressUp, pressDown, and print methods in Task C.

  Submit only Board.cpp.

Task D (20 points)
Define code to play a game, that is, define start method in Board.cpp.

```
void Board::start()
{
```

```
   int round = 1;
   call selectRandomCell twice to generate two values

   while (true)
   {
         if (max equals target)
           print "Congratulation!" and leave the loop

         //reference: https://stackoverflow.com/questions/10463201/getch-and-arrow-codes
         if (getchar() == '\033')
         { // if the first value is esc
            getchar(); // skip the [
            switch(getchar()) { // the real value
                case 'A':
                    // code for arrow up
                    cout << "Round " << setw(4) << round << ": ";
                    cout << "Press UP. " << endl;
                    pressUp();
                    round++;
                    break;
                    //You finish the rest code
            } //end of switch
          } //end of if
     } //end of while
 }
```

In TestBoard.cpp, create Board object using different constructors and use setGoal method to change target (you need to implement setGoal method) and call start method to see how your code works.

An example of TestBoard.cpp is as follows.

```
#include <iostream>
#include "Board.hpp"
using namespace std;

int main()
{
   Board game(3, 3); //create a Board object with 3 rows and 3 columns.
   game.start();
   return 0;
}
```

You can use srand(time(NULL)); in constructors (when objects are created) so that rand() uses different random integer sequence at different time you run the code. But when you submit code to gradescope, only use srand(1); or simply ignore this line. Otherwise, grade scripts will not work due to random selection of cells.

How to work on the project

Download Board.hpp, TestBoard.cpp, makefile from blackboard, implement Board.cpp.

To run the project, in command line, enter
make
Then run
./run

Note

For grading purpose, choose srand(1); when submit this project, thus the same random sequence is chosen each time. You can change it to srand(time(NULL)); when you play with it.

Acknowledgment

Serra Canca provided project description. Nursima Donuk wrote autograder scripts.

Sample run

Here is a 3x3 board with target 32.

```
+----+----+----+
|    |    |    |
+----+----+----+
|    |   1|    |
+----+----+----+
|    |    |    |
+----+----+----+
+----+----+----+
|    |   1|    |
+----+----+----+
|    |   1|    |
+----+----+----+
|    |    |    |
+----+----+----+
^[[C
Round     1:  Press  RIGHT.  Goal:  32
```

```
+----+----+----+
|    |    |   1|
+----+----+----+
|   1|    |   1|
+----+----+----+
|    |    |    |
+----+----+----+
^[[A
Round    2: Press UP.  Goal:  32
+----+----+----+
|   1|    |   2|
+----+----+----+
|    |    |    |
+----+----+----+
|    |    |   1|
+----+----+----+
^[[D
Round    3: Press LEFT.  Goal:  32
+----+----+----+
|   1|   2|    |
+----+----+----+
|    |    |    |
+----+----+----+
|   1|   1|    |
+----+----+----+
^[[A
Round    4: Press UP.  Goal:  32
+----+----+----+
|   2|   2|    |
+----+----+----+
|    |   1|   1|
+----+----+----+
|    |    |    |
+----+----+----+
^[[D
Round    5: Press LEFT.  Goal:  32
+----+----+----+
|   4|    |    |
+----+----+----+
|   2|   1|    |
+----+----+----+
|    |    |    |
+----+----+----+
^[[B
Round    6: Press DOWN.  Goal:  32
+----+----+----+
```

```
|     |     |    1|
+----+----+----+
|    4|     |     |
+----+----+----+
|    2|    1|     |
+----+----+----+
^[[B
Round     7: Press DOWN.  Goal :  32
+----+----+----+
|     |     |     |
+----+----+----+
|    4|    1|     |
+----+----+----+
|    2|    1|    1|
+----+----+----+
^[[D
Round     8: Press LEFT.  Goal :  32
+----+----+----+
|     |     |     |
+----+----+----+
|    4|    1|     |
+----+----+----+
|    2|    2|    1|
+----+----+----+
^[[D
Round     9: Press LEFT.  Goal :  32
+----+----+----+
|    1|     |     |
+----+----+----+
|    4|    1|     |
+----+----+----+
|    4|    1|     |
+----+----+----+
^[[B
Round    10: Press DOWN.  Goal :  32
+----+----+----+
|     |     |     |
+----+----+----+
|    1|     |     |
+----+----+----+
|    8|    2|    1|
+----+----+----+
^[[C
Round    11: Press RIGHT.  Goal :  32
+----+----+----+
|     |     |    1|
```

```
+----+----+----+
|    |    |   1|
+----+----+----+
|   8|   2|   1|
+----+----+----+
^[[B
Round   12: Press DOWN.  Goal:  32
+----+----+----+
|    |    |   1|
+----+----+----+
|    |    |   1|
+----+----+----+
|   8|   2|   2|
+----+----+----+
^[[D
Round   13: Press LEFT.  Goal:  32
+----+----+----+
|   1|    |    |
+----+----+----+
|   1|   1|    |
+----+----+----+
|   8|   4|    |
+----+----+----+
^[[C
Round   14: Press RIGHT.  Goal:  32
+----+----+----+
|    |    |   1|
+----+----+----+
|    |   1|   2|
+----+----+----+
|    |   8|   4|
+----+----+----+
^[[A
Round   15: Press UP.  Goal:  32
+----+----+----+
|    |   1|   1|
+----+----+----+
|    |   8|   2|
+----+----+----+
|    |   1|   4|
+----+----+----+
^[[C
Round   16: Press RIGHT.  Goal:  32
+----+----+----+
|    |   1|   2|
+----+----+----+
```

```
|      |    8|    2|
+----+----+----+
|      |    1|    4|
+----+----+----+
^[[A
Round   17:  Press  UP.  Goal :  32
+----+----+----+
|    1|    1|    4|
+----+----+----+
|      |    8|    4|
+----+----+----+
|      |    1|     |
+----+----+----+
^[[A
Round   18:  Press  UP.  Goal :  32
+----+----+----+
|    1|    1|    8|
+----+----+----+
|    1|    8|     |
+----+----+----+
|      |    1|     |
+----+----+----+
^[[C
Round   19:  Press  RIGHT.  Goal :  32
+----+----+----+
|      |    2|    8|
+----+----+----+
|      |    1|    8|
+----+----+----+
|      |    1|    1|
+----+----+----+
^[[A
Round   20:  Press  UP.  Goal :  32
+----+----+----+
|      |    2|   16|
+----+----+----+
|      |    2|    1|
+----+----+----+
|      |     |    1|
+----+----+----+
^[[A
Round   21:  Press  UP.  Goal :  32
+----+----+----+
|      |    4|   16|
+----+----+----+
|      |     |    2|
```

```
+----+----+----+
|    1|    |    |
+----+----+----+
^[[C
Round   22: Press RIGHT. Goal: 32
+----+----+----+
|    |   4|  16|
+----+----+----+
|    |   1|   2|
+----+----+----+
|    |    |   1|
+----+----+----+
^[[B
Round   23: Press DOWN. Goal: 32
+----+----+----+
|    1|    |  16|
+----+----+----+
|    |   4|   2|
+----+----+----+
|    |   1|   1|
+----+----+----+
^[[C
Round   24: Press RIGHT. Goal: 32
+----+----+----+
|    |   1|  16|
+----+----+----+
|    1|   4|   2|
+----+----+----+
|    |    |   2|
+----+----+----+
^[[A
Round   25: Press UP. Goal: 32
+----+----+----+
|    1|   1|  16|
+----+----+----+
|    |   4|   4|
+----+----+----+
|    |    |   1|
+----+----+----+
^[[C
Round   26: Press RIGHT. Goal: 32
+----+----+----+
|    |   2|  16|
+----+----+----+
|    |    |   8|
+----+----+----+
```

```
|    1|     |    1|
+----+----+----+
^[[D
Round   27: Press LEFT. Goal: 32
+----+----+----+
|    2|   16|     |
+----+----+----+
|    8|    1|     |
+----+----+----+
|    2|     |     |
+----+----+----+
^[[C
Round   28: Press RIGHT. Goal: 32
+----+----+----+
|     |    2|   16|
+----+----+----+
|     |    8|    1|
+----+----+----+
|     |    1|    2|
+----+----+----+
^[[D
Round   29: Press LEFT. Goal: 32
+----+----+----+
|    2|   16|     |
+----+----+----+
|    8|    1|    1|
+----+----+----+
|    1|    2|     |
+----+----+----+
^[[D
Round   30: Press LEFT. Goal: 32
+----+----+----+
|    2|   16|     |
+----+----+----+
|    8|    2|     |
+----+----+----+
|    1|    2|    1|
+----+----+----+
^[[A
Round   31: Press UP. Goal: 32
+----+----+----+
|    2|   16|    1|
+----+----+----+
|    8|    4|     |
+----+----+----+
|    1|     |    1|
```

```
+----+----+----+
^[[C
Round   32: Press RIGHT.  Goal:  32
+----+----+----+
|   2|  16|   1|
+----+----+----+
|    |   8|   4|
+----+----+----+
|   1|    |   2|
+----+----+----+
^[[B
Round   33: Press DOWN.  Goal:  32
+----+----+----+
|   1|    |   1|
+----+----+----+
|   2|  16|   4|
+----+----+----+
|   1|   8|   2|
+----+----+----+
^[[C
Round   34: Press RIGHT.  Goal:  32
+----+----+----+
|    |   1|   2|
+----+----+----+
|   2|  16|   4|
+----+----+----+
|   1|   8|   2|
+----+----+----+
^[[D
Round   35: Press LEFT.  Goal:  32
+----+----+----+
|   1|   2|   1|
+----+----+----+
|   2|  16|   4|
+----+----+----+
|   1|   8|   2|
+----+----+----+
Game over. Try again.
laptopuser@laptopUsers-MBP code % vi Board.cpp
laptopuser@laptopUsers-MBP code % make
g++ -std=c++11 -c Board.cpp
g++ -std=c++11 -o run TestBoard.o Board.o
laptopuser@laptopUsers-MBP code % ./run
+----+----+----+
|    |    |    |
+----+----+----+
```

```
|      |    1|      |
+----+----+----+
|      |      |      |
+----+----+----+
+----+----+----+
|      |    1|      |
+----+----+----+
|      |    1|      |
+----+----+----+
|      |      |      |
+----+----+----+
^[[A
Round     1:  Press  UP.  Goal :  32
+----+----+----+
|      |    2|    1|
+----+----+----+
|      |      |      |
+----+----+----+
|      |      |      |
+----+----+----+
^[[D
Round     2:  Press  LEFT.  Goal :  32
+----+----+----+
|    2|    1|      |
+----+----+----+
|      |      |      |
+----+----+----+
|      |      |    1|
+----+----+----+
^[[A
Round     3:  Press  UP.  Goal :  32
+----+----+----+
|    2|    1|    1|
+----+----+----+
|      |      |      |
+----+----+----+
|      |    1|      |
+----+----+----+
^[[D
Round     4:  Press  LEFT.  Goal :  32
+----+----+----+
|    2|    2|      |
+----+----+----+
|      |    1|      |
+----+----+----+
|    1|      |      |
```

```
+----+----+----+
^[[D
Round     5:  Press  LEFT.  Goal :  32
+----+----+----+
|    4|    1|     |
+----+----+----+
|    1|     |     |
+----+----+----+
|    1|     |     |
+----+----+----+
^[[A
Round     6:  Press  UP.  Goal :  32
+----+----+----+
|    4|    1|     |
+----+----+----+
|    2|     |    1|
+----+----+----+
|     |     |     |
+----+----+----+
^[[A
Round     7:  Press  UP.  Goal :  32
+----+----+----+
|    4|    1|    1|
+----+----+----+
|    2|     |     |
+----+----+----+
|     |    1|     |
+----+----+----+
^[[D
Round     8:  Press  LEFT.  Goal :  32
+----+----+----+
|    4|    2|     |
+----+----+----+
|    2|     |     |
+----+----+----+
|    1|     |    1|
+----+----+----+
^[[D
Round     9:  Press  LEFT.  Goal :  32
+----+----+----+
|    4|    2|    1|
+----+----+----+
|    2|     |     |
+----+----+----+
|    2|     |     |
+----+----+----+
```

```
^[[A
Round    10:  Press UP.  Goal :  32
+----+----+----+
|    4|    2|    1|
+----+----+----+
|    4|    1|     |
+----+----+----+
|     |     |     |
+----+----+----+
^[[A
Round    11:  Press UP.  Goal :  32
+----+----+----+
|    8|    2|    1|
+----+----+----+
|     |    1|     |
+----+----+----+
|    1|     |     |
+----+----+----+
^[[C
Round    12:  Press RIGHT.  Goal :  32
+----+----+----+
|    8|    2|    1|
+----+----+----+
|     |     |    1|
+----+----+----+
|    1|     |    1|
+----+----+----+
^[[A
Round    13:  Press UP.  Goal :  32
+----+----+----+
|    8|    2|    2|
+----+----+----+
|    1|     |    1|
+----+----+----+
|     |     |    1|
+----+----+----+
^[[D
Round    14:  Press LEFT.  Goal :  32
+----+----+----+
|    8|    4|     |
+----+----+----+
|    2|     |     |
+----+----+----+
|    1|    1|     |
+----+----+----+
^[[D
```

```
Round   15: Press LEFT. Goal: 32
+----+----+----+
|   8|   4|    |
+----+----+----+
|   2|    |   1|
+----+----+----+
|   2|    |    |
+----+----+----+
^[[A
Round   16: Press UP. Goal: 32
+----+----+----+
|   8|   4|   1|
+----+----+----+
|   4|    |    |
+----+----+----+
|    |    |   1|
+----+----+----+
^[[B
Round   17: Press DOWN. Goal: 32
+----+----+----+
|   1|    |    |
+----+----+----+
|   8|    |    |
+----+----+----+
|   4|   4|   2|
+----+----+----+
^[[D
Round   18: Press LEFT. Goal: 32
+----+----+----+
|   1|    |    |
+----+----+----+
|   8|   1|    |
+----+----+----+
|   8|   2|    |
+----+----+----+
^[[A
Round   19: Press UP. Goal: 32
+----+----+----+
|   1|   1|    |
+----+----+----+
|  16|   2|    |
+----+----+----+
|    |   1|    |
+----+----+----+
^[[D
Round   20: Press LEFT. Goal: 32
```

```
+----+----+----+
|    2|    |    |
+----+----+----+
|   16|    2|    |
+----+----+----+
|    1|    |    1|
+----+----+----+
^[[A
Round    21:  Press  UP.  Goal :  32
+----+----+----+
|    2|    2|    1|
+----+----+----+
|   16|    |    1|
+----+----+----+
|    1|    |    |
+----+----+----+
^[[D
Round    22:  Press  LEFT.  Goal :  32
+----+----+----+
|    4|    1|    |
+----+----+----+
|   16|    1|    1|
+----+----+----+
|    1|    |    |
+----+----+----+
^[[A
Round    23:  Press  UP.  Goal :  32
+----+----+----+
|    4|    2|    1|
+----+----+----+
|   16|    1|    |
+----+----+----+
|    1|    |    |
+----+----+----+
^[[C
Round    24:  Press  RIGHT.  Goal :  32
+----+----+----+
|    4|    2|    1|
+----+----+----+
|    1|   16|    1|
+----+----+----+
|    |    |    1|
+----+----+----+
^[[A
Round    25:  Press  UP.  Goal :  32
+----+----+----+
```

```
|    4|    2|    2|
+----+----+----+
|    1|   16|    1|
+----+----+----+
|    1|     |     |
+----+----+----+
^[[D
Round   26: Press LEFT. Goal: 32
+----+----+----+
|    4|    4|    1|
+----+----+----+
|    1|   16|    1|
+----+----+----+
|    1|     |     |
+----+----+----+
^[[D
Round   27: Press LEFT. Goal: 32
+----+----+----+
|    8|    1|     |
+----+----+----+
|    1|   16|    1|
+----+----+----+
|    1|     |    1|
+----+----+----+
^[[C
Round   28: Press RIGHT. Goal: 32
+----+----+----+
|     |    8|    1|
+----+----+----+
|    1|   16|    1|
+----+----+----+
|    1|     |    2|
+----+----+----+
^[[A
Round   29: Press UP. Goal: 32
+----+----+----+
|    2|    8|    2|
+----+----+----+
|     |   16|    2|
+----+----+----+
|    1|     |     |
+----+----+----+
^[[A
Round   30: Press UP. Goal: 32
+----+----+----+
|    2|    8|    4|
```

```
+----+----+----+
|    1|   16|    |
+----+----+----+
|    |    1|    |
+----+----+----+
^[[B
Round   31: Press DOWN. Goal: 32
+----+----+----+
|    |    8|    |
+----+----+----+
|    2|   16|    1|
+----+----+----+
|    1|    1|    4|
+----+----+----+
^[[D
Round   32: Press LEFT. Goal: 32
+----+----+----+
|    8|    |    1|
+----+----+----+
|    2|   16|    1|
+----+----+----+
|    2|    4|    |
+----+----+----+
^[[A
Round   33: Press UP. Goal: 32
+----+----+----+
|    8|   16|    2|
+----+----+----+
|    4|    4|    |
+----+----+----+
|    |    1|    |
+----+----+----+
^[[D
Round   34: Press LEFT. Goal: 32
+----+----+----+
|    8|   16|    2|
+----+----+----+
|    8|    |    1|
+----+----+----+
|    1|    |    |
+----+----+----+
^[[A
Round   35: Press UP. Goal: 32
+----+----+----+
|   16|   16|    2|
+----+----+----+
```

```
|     1|      |     1|
+----+----+----+
|      |      |     1|
+----+----+----+
^[[D
Round    36:  Press  LEFT.  Goal :  32
+----+----+----+
|    32|     2|      |
+----+----+----+
|     2|      |      |
+----+----+----+
|     1|      |     1|
+----+----+----+
Congratulation!
```