Задача: Защита башен

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда на 10 ходов
Ограничение по памяти:	64 мегабайта

Игровое поле представляет из себя матрицу размером 20×20 клеток. Верхняя левая клетка поля имеет координаты $(0, 0)$, нижняя правая — $(19, 19)$. В позиции $(2, 2)$ находится башня первого игрока, в позиции $(17, 17)$ — башня второго игрока. Каждая башня имеет 100 единиц здоровья.

В начальный момент игры в клетке $(0, 0)$ находится первый игрок, в клетке $(19, 19)$ — второй игрок. Игрок занимает однку клетку. Игрок не имеет жизней, т.е. он не убиваемый. Игрок может перемещаться в соседние, смежные по стороне клетки поля. Игрок не может перемещаться на ту клетку, которая является занятой башней или пушкой (будет описано ниже).

Время от времени на игровом поле, в случайно выбранной клетке, свободной от игроков, башен и пушек, появляется монетка номиналом 1 золотой. Если в клетке уже была монетка, то так одна монетка там и остается. Если игрок, на своем ходе перемещается в клетку, в которой есть монетка, то монетка исчезает из этой клетки, а к счетчику монеток игрока добавляется единица. Изначально счетчик монеток каждого игрока пуст. Монетки появляются на поле случайным образом, с частотой примерно 1 монетка на 3 ходов (здесь ходом считается ход одного и ответ другого).

Набрав достаточное количество монет, игрок может построить пушку. Дадим описание пушки. Стоя в некоторой клетке, игрок может за один ход построить пушку в любой свободной смежной по стороне или углу клетке (т.е. в одной из 8 соседних клеток). Как только пушка построена, она начинает занимать клетку, в которой она находится. В игре есть 3 вида пушек.

Каждый вид обладает следующими характеристиками.

- **Стоимость:** C золотых. (Такая стоимость вычитается с денежного счета игрока, когда он строит пушку. Игрок не может построить пушку, если у него меньше денег).
- **Жизни:** H единиц здоровья. (Когда жизни пушки падают меньше 1, то пушка уничтожается, клетка, которую она занимала, становится пустой).
- **Радиус:** R клеток. (Величина, на которую пушка может стрелять. Пусть пушка имеет координаты (x_1, y_1). Пушка сможет выстрелить в клетку с координатами (x_2, y_2), если выполняется условие $|x_1 - x_2| + |y_1 - y_2| \leq R$. Такое расстояние еще называется Манхэттенским).
- **Мощность:** P единиц урона. (Каждый выстрел пушки отнимает это значение от жизней цели. Целью пушки может быть либо башня противника, либо пушка противника).

Первый вид пушки.

- **Стоимость:** 3 золотых.
- **Жизни:** 9 единиц здоровья.
- **Радиус:** 1 клетка.
- **Мощность:** 2 единицы урона.

Второй вид пушки.

- **Стоимость:** 7 золотых.
- **Жизни:** 12 единиц здоровья.
- **Радиус:** 2 клетки.
- **Мощность:** 3 единицы урона.

Третий вид пушки.

- **Стоимость:** 12 золотых.
- **Жизни:** 5 единиц здоровья.
- **Радиус:** 3 клетки.
- **Мощность:** 4 единиц урона.

На каждом ходу игрок может сделать одно из трех действий:

1. остаться стоять на месте;
2. переместиться в свободную смежную по стороне клетку;
3. построить пушку, указав координаты соседней свободной клетки для строительства.

После того, как сделает ход первый, а затем второй игрок, автоматически стреляют пушки. Делают они это одновременно, по описанному ниже правилу.

Для каждой пушки вычисляется, может ли она стрелять в башню противника (если позволяет радиус). Если башня находится в радиусе досягаемости, то пушка наносит урон башне противника. От ее жизни отнимается количество единиц урона данной пушки. Если же пушка не достает до башни противника, то выбирается самая близкая пушка противника, по расстоянию. Если таких пушек несколько, то выбирается сначала та, которая расположена раньше по часовой стрелке, считая началом клетку над клеткой пушки (т.е. $(x, y - r)$, где r упомянутый выше минимальный радиус). Пушка наносит урон выбранной пушке противника. От жизней пушки отнимается величина урона. Если жизни пушки стали меньше либо равны нулю, то пушка исчеазет (однако она уже могла выстрелить, ведь стреляют одновременно все). Если в радиусе досягаемости пушки нет ни одной другой пушки противника, то пушка ничего не делает.

Кроме денежного счета, у игрока есть счет очков. За каждую подобранную золотую монетку игрок получает 1 очко счета. За каждую построенную пушку первого вида, игрок получает 2 очка счета. За каждую построенную пушку второго типа, игрок получает 4 очка счета. За каждую построенную пушку третьего типа игрок получает 7 очков счета.

Если после очередной серии выстрелов из пушек, жизни одной из башен становятся меньше либо равны нулю, башня считается разрушеной, победителем считается игрок, разрушивший башню. Если игроки одновременно разрушили башню, то победителем считется игрок, набравший наибольший счет очков.

Если за 2222 ходов башни противников не были разрушены, то победитлем считается тот, кто набрал наибольший счет. Если счет игроков равный, игра завершается ничьей.

Игра запускается несколько раз. Половину запусков Ваша программа будет ходить первой, половину — второй.

Формат входного файла

Перед каждым ходом в стандартный поток ввода Вашей программе будет передано текущее состояние поля, 20 строк по 20 чисел в каждой. Завершаться передача состояния будет пустой строкой. Каждое из этих чисел a_{ij} соответствует одной клетке поля.

- Если $1 \le a_{ij} \le 100$, то это единицы жизни башенки. Башенки всегда находятся в клетках $(2, 2)$ и $(17, 17)$.
- Если $201 \le a_{ij} \le 209$, то $(a_{ij} - 200)$ — единицы жизни пушки первого типа первого игрока.
- Если $301 \le a_{ij} \le 312$, то $(a_{ij} - 300)$ — единицы жизни пушки второго типа первого игрока.
- Если $401 \le a_{ij} \le 405$, то $(a_{ij} - 400)$ — единицы жизни пушки третьего типа первого игрока.
- Если $a_{ij} = 500$, то здесь находится первый игрок.

- Если $601 \le a_{ij} \le 609$, то $(a_{ij} - 600)$ — единицы жизни пушки первого типа второго игрока.
- Если $701 \le a_{ij} \le 712$, то $(a_{ij} - 700)$ — единицы жизни пушки второго типа второго игрока.
- Если $801 \le a_{ij} \le 805$, то $(a_{ij} - 800)$ — единицы жизни пушки третьего типа второго игрока.
- Если $a_{ij} = 900$, то здесь находится второй игрок.
- Если $a_{ij} = 1000$, то в клетке находится монетка.
- Если $a_{ij} = 0$, то клетка свободна.

Формат выходного файла

На каждом ходу Вам требуется вывести команду.

1. `S` — остаться стоять на месте.
2. `M W` — переместиться в свободную смежную по стороне клетку. Где `W` — это одно из четырех направлений для перемещения: `U` — вверх, `L` — влево, `R` — вправо, `D` — вниз.
3. `B X Y T` — построить пушку вида T $(1 \le T \le 3)$, указав координаты соседней свободной клетки для строительства. Где (X, Y) может быть одним из восьми значений: $(X-1, Y)$, $(X-1, Y+1)$, $(X, Y+1)$, $(X+1, Y+1)$, $(X+1, Y)$, $(X+1, X-1)$, $(X, Y-1)$, $(X-1, Y-1)$.

Если вы выведете не корректную команду, либо сделаете ход в занятую клетку, либо попытаетесь построить пушку в занятой клетке, либо попытаетесь построить пушку, не имея средств, то Ваш ход будет проигнорирован и Вы останетесь стоять на месте.

После вывода хода выведите пустую строку (то есть после команды нужно вывести два перевода строки). Не забывайте сбрасывать буфер выходного потока после каждого хода. В C/C++ это можно сделать с помощью `fflush(stdout);`, в Pascal с помощью `flush(output);`

Примеры

стандартный ввод
500 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 100 100 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 900
стандартный вывод
M R

Пример реализованной стратегии, на каждом ходу делающей случайный ход (C++):

```
#include <ctime>
#include <cstdlib>
#include <cstdio>
using namespace std;
const int SZ = 20;
int field[SZ][SZ];
const char dir[5] = "ULRD";
const int build_dx[] = {1, -1, 0,  0, 1,  1, -1, -1};
const int build_dy[] = {0,  0, 1, -1, 1, -1,  1, -1};
const int move_dx[] = {-1,  0, 0, 1};
const int move_dy[] = { 0, -1, 1, 0};
bool good (int x, int y) {
  return (x >= 0 && x < 20 && y >= 0 && y < 20);
}
int main() {
    srand((unsigned)time(NULL));
    int money = 0, x, y;
    while (1) {
        for (int i = 0; i < SZ; i++) {
            for (int j = 0; j < SZ; j++) {
                scanf("%d", &field[i][j]);
                if (field[i][j] == 500) {
                  x = i;
                  y = j;
                }
            }
        }
        int build_direction = rand () % 8;
        x = x + build_dx[build_direction];
        y = y + build_dy[build_direction];
        if (good (x, y) && field[x][y] == 0 && money >= 7) {
        printf ("B %d %d 2\n", y, x);
        money -= 7;
        } else {
            int move_direction = rand () % 4;
         x = x - build_dx[build_direction] + move_dx[move_direction];
         y = y - build_dy[build_direction] + move_dy[move_direction];
         if (good (x, y) && (field[x][y] == 0 || field[x][y] == 1000)) {
         if (field[x][y] == 1000) {
        money++;
         }
         printf("M %c\n", dir[move_direction]);
       } else {
                printf("S\n");
         }
        }
        printf("\n");
        fflush(stdout);
    }
    return 0;
}
```