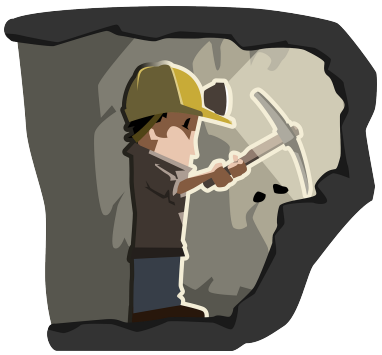
Lectura del plano de la mina



Hemos encontrado un plano de una antigua mina de diamantes. En el plano se encuentran dibujados los lugares donde se encuentran los diamantes, así como diversos túneles ya cavados, galerías que se han tupido con arena y hay que volver a cavar y lugares donde hay piedras que tendrán que moverse o muros por los que no podremos pasar. Vamos a ir a cavar las galerías tupidas y recuperar todas las gemas que podamos.

Tu tarea en este problema es leer el plano de la mina y la posición en la que se encuentra el minero y mostrarlo por pantalla.

Requisitos de implementación Este problema está asociado a la práctica que se debe realizar en la asignatura. Por ello debe resolverse siguiendo las siguientes indicaciones.

Se debe crear un módulo `mina`, con dos ficheros: `mina.h` donde se definirán las siguientes estructuras de datos y prototipos, y un fichero `mina.cpp` donde se implementarán las funciones siguientes.

Se usará un tipo enumerado `tElemento` para definir los elementos que pueden encontrarse en la mina. Estos pueden ser: `TIERRA`, `PIEDRA`, `MURO`, `GEMA`, y `SALIDA`. Ademas en el plano tendremos `LIBRE` para representar los túneles ya cavados, y `MINERO` para representar donde se encuentra el minero.

El plano lo guardaremos en memoria en una matriz cuyos valores son los elementos del tipo enumerado.

Definiremos una mina, de tipo `tMina`, como un plano y la posición del minero. Para definir la posición del minero se puede utilizar el tipo `pair` de la STL, un struct con dos campos enteros, o dos valores independientes.

Debes implementar las siguientes funciones:

1. `tTElemento char2elem (char c);`, transforma un carácter en un valor del tipo enumerado `tElemento`.

2. `std::ostream& operator<< (std::ostream& out, tElemento const& e);` para escribir los valores del tipo enumerado. El operador se sobrecarga para el tipo enumerado, no para la matriz. La matriz se muestra con las funciones siguientes.

3. `void cargar_mina(std::istream& fichero, tMina& mina);` lee el plano de la mina del *fichero* y lo guarda en memoria. Actualiza la posición del minero en la mina.

4. `void dibujar1_1(tMina const& mina);`, muestra por pantalla el plano en escala 1:1

5. `void dibujar3_1(tMina cons & mina) ;` muestra por pantalla el plano en escala 3:1

En la función `resuelveCaso` se gestionará la lectura y escritura de datos. La función `main` gestionará la lectura de casos, como se está haciendo en los programas del juez. En el main se redireccionará el fichero de entrada como hacemos para resolver los problemas del juez, y se llamará a las funciones que leen o escriben datos en el módulo mina con el flujo `cin`. Estas funciones reciben el flujo de datos sobre el que deben actuar.

Para subir al juez varios archivos se deben seleccionar todos ellos y arrastrarlos conjuntamente.

Entrada

La entrada comienza con el número de minas que queremos explorar. La descripción de cada mina comienza con una línea en que se dan los parámetros del juego: escala a la que se mostrará el plano, 1 si se muestra a escala 1:1 y 2 si se muestra a escala 3:1; a continuación un 2 para indicar que la entrada será por fichero. En la línea siguiente se muestran las dimensiones del plano, el primer valor será el número de filas n y el segundo el número de columnas m. A continuación se muestran n líneas cada una con m valores indicando el elemento que se encuentra en ese punto de la mina. Los caracteres que describen la mina en el fichero de entrada son: el minero (J), tierra(T), piedra (P), muro (M), gema (G), salida (S). Los túneles ya cavados se muestran con un guión (-).

Salida

Para cada mina se dibuja el plano a la escala requerida. Los caracteres que representan los elementos en el plano de salida son: piedra (@), muro (X), tierra (.), minero (M), gema (G), salida(S). Los túneles ya cavados se representan con el guión (-).

Entrada de ejemplo

```
5
1 2
7 10
GM-J-MMMMM
GM---MMTGT
TM-G-MMTTT
GTTG--MTTT
TTTTTTTTTT
TGTTMGTMST
TTTTMTTMTT
2 2
7 10
JMGMMMMMMM
-MGMGTTMTS
-MGMMMTTTT
--GP-MMPMM
TTTP-TTTTT
TGTP-TTGTT
TTTP-TTTTT
1 2
6 7
-JPPSTT
TPTMTTT
GP-MTTG
TM-MTTT
TM-MTTT
TM-MTTT
2 2
6 7
PJTPPPP
TPTMPPP
TP-MPPP
MMGTTTT
MMMMMMT
MMMMSTT
1 2
5 2
ST
MT
TT
TM
TJ
```

Salida de ejemplo

```
GX-M-XXXXX
GX---XX.G.
.X-G-XX...
G..G--X...
..........
.G..XG.XS.
....X..X..

MMMXXXGGGXXXXXXXXXXXXXXXXXXXX
MMMXXXGGGXXXXXXXXXXXXXXXXXXXX
MMMXXXGGGXXXXXXXXXXXXXXXXXXXX
---XXXGGGXXXGGG......XXX...SSS
---XXXGGGXXXGGG......XXX...SSS
---XXXGGGXXXGGG......XXX...SSS
---XXXGGGXXXXXXXXX.............
---XXXGGGXXXXXXXXX.............
---XXXGGGXXXXXXXXX.............
------GGG@@@---XXXXXX@@@XXXXXX
------GGG@@@---XXXXXX@@@XXXXXX
------GGG@@@---XXXXXX@@@XXXXXX
.........@@@---................
.........@@@---................
.........@@@---................
...GGG...@@@---......GGG......
...GGG...@@@---......GGG......
...GGG...@@@---......GGG......
.........@@@---................
.........@@@---................
.........@@@---................

-M@@S..
.@.X...
G@-X..G
.X-X...
.X-X...
.X-X...

@@@MMM...@@@@@@@@@@@@
@@@MMM...@@@@@@@@@@@@
@@@MMM...@@@@@@@@@@@@
...@@@...XXX@@@@@@@@@@
...@@@...XXX@@@@@@@@@@
...@@@...XXX@@@@@@@@@@
...@@@---XXX@@@@@@@@@@
...@@@---XXX@@@@@@@@@@
...@@@---XXX@@@@@@@@@@
XXXXXXGGG.............
XXXXXXGGG.............
XXXXXXGGG.............
XXXXXXXXXXXXXXXXXX...
XXXXXXXXXXXXXXXXXX...
XXXXXXXXXXXXXXXXXX...
XXXXXXXXXXXXSSS......
XXXXXXXXXXXXSSS......
XXXXXXXXXXXXSSS......

S.
X.
..
.X
.M
```

Autor: Isabel Pita.