Maze generation problem

AAPP 1st challenge AA 2022/23

Mauro Famà 10631287

Structure of the problem

0	1	2
3	4	5

the maze is represented by a matrix

Structure of the problem

0	1	2
3	4	5

the maze is represented by a matrix

Every “cell” of the maze can be identified by its coordinates or its position

e.g. element in position (1,2) is 4

Structure of the problem

0	1	2
3	4	5

the maze is represented by a matrix

Every “cell” of the maze can be identified by its coordinates or its position

e.g. element in position (1,2) is 4

It is possible to swap between these two identifiers with simple math operations:

(i,j) -> i*columns + j

k -> (k mod columns, k/rows)

Disjoint sets

Cells are represented by their positions in the matrix (not coordinates). I used an array of integers called parent[]. If we are dealing with N cells, the i’th element of the parent[] array is the parent of the i’th cell, which is the i’th element of the array. These relationships create one or more virtual trees.

Disjoint sets

Cells are represented by their positions in the matrix (not coordinates). I used an array of integers called parent[]. If we are dealing with N cells, the i'th element of the parent[] array is the parent of the i'th cell, which is the i'th element of the array. These relationships create one or more virtual trees.

At the end of the execution of the algorithm, we'll have the parent[] array representing all the disjoint sets created, thus the maze.

parent[]

0	1	2	4	0	0

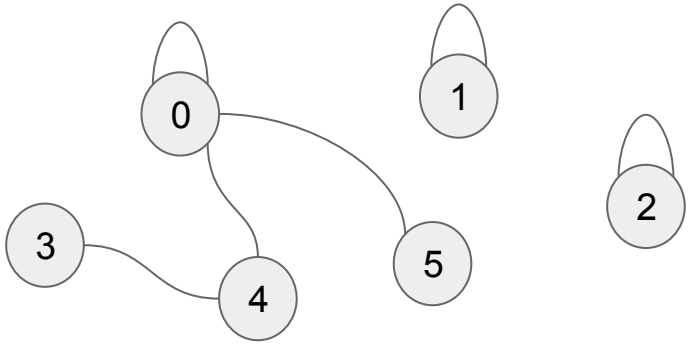
Disjoint sets

Cells are represented by their positions in the matrix (not coordinates). I used an array of integers called parent[]. If we are dealing with N cells, the i'th element of the parent[] array is the parent of the i'th cell, which is the i'th element of the array. These relationships create one or more virtual trees.

At the end of the execution of the algorithm, we'll have the parent[] array representing all the disjoint sets created, thus the maze.

parent[]

0	1	2	4	0	0
0	1	2	3	4	5



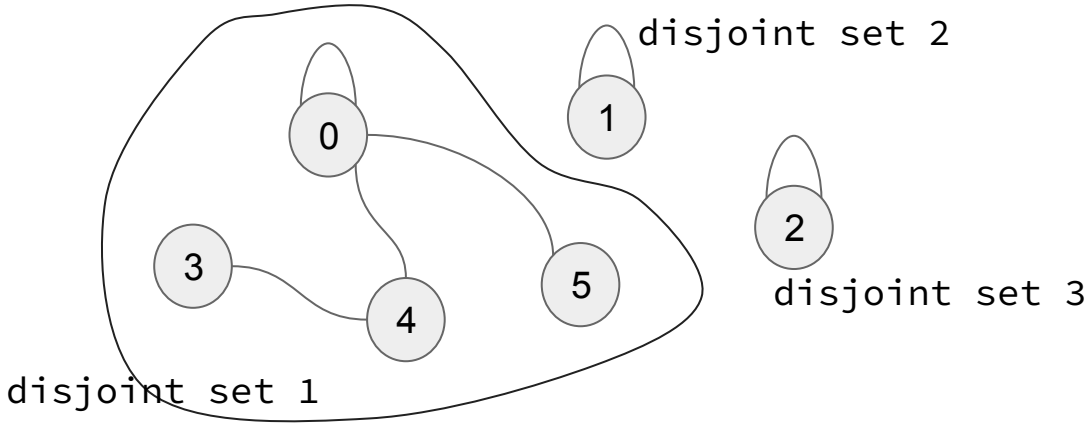
Disjoint sets

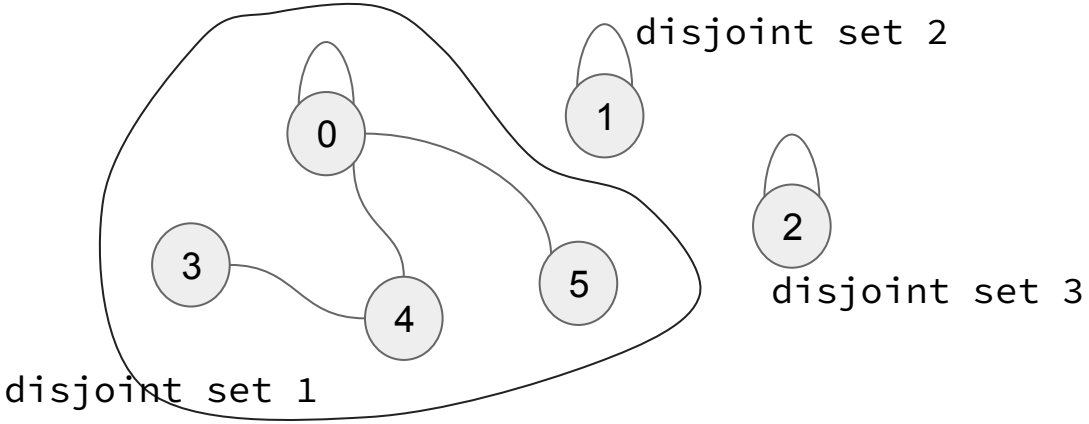
Cells are represented by their positions in the matrix (not coordinates). I used an array of integers called parent[]. If we are dealing with N cells, the i'th element of the parent[] array is the parent of the i'th cell, which is the i'th element of the array. These relationships create one or more virtual trees.

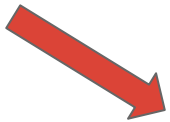
At the end of the execution of the algorithm, we'll have the parent[] array representing all the disjoint sets created, thus the maze.

parent[]

0	1	2	4	0	0







0	1	2
3	4	5

Disjoint set functions - Find

```
// Finds set of item x
int Find(int x){
   // Finds the representative of the set that x is an element of
   if (parent[x] != x) {
       // if x is not the parent of itself then x is not the representative of his set
       parent[x] = Find(parent[x]);
       // so we recursively call Find on its parent and move i's node directly under the representative of this set
   }
   return parent[x];
}
```

Disjoint set functions - Un ion

```
// Do union of two sets represented by x and y.
void Union(int i, int j) {
   // Find the representatives (or the root nodes) for the set that includes i
   int irep = this->Find(i);
   // And do the same for the set that includes j
   int jrep = this->Find(j);
   // Make the parent of i’s representative be j’s  representative effectively moving all of i’s set into j’s set)
   this->parent[irep] = jrep;
}
```

DISJOINT SET FUNCTIONS - SAMESET

```
// Return true if two elements are in the same set, false otherwise

bool SameSet(int x, int y){

   return Find(x) == Find(y);

}
```