HospitalGeneratorRDF

Starting point

H-Outbreak gives us the following relations:

Bed - Ordinary room - Ordinary Service

Bed - ICU Unit

Bed - ER Unit

Radiology Bed

Surgery Bed

To create its hospital, it has hardcoded the following values:

Parámetro	Valor
No. Radiology Beds	4
No. Surgery Beds	3
No. ICU Rooms	1
No. ER Rooms	1
No. Ordinary services	8

Y pide los siguientes parámetros:

- No. ER Beds
- No. ICU Beds
- No. Rooms of each ordinary Service
- No. Beds for ordinary Service

What do we need?

- All Hospitalization Units
- Services for ICU, ER, Radiology and Surgery
- Everything in the Location hierarchy above Room
- Relative spatial relationships

HospitalGeneratorRDF Parameters

- *index*: First number for ids (preferably greater than the last id of the H-Outbreak hospital entities)
- *huPerService*: Number of Hospitalization Units per Normal Service
- *n_floors*: Number of hospital Floors
- *n_huPerFloor*: Number of Hospitalization Units per Floor
- *nRows*, *nColumns*: Number of Rows and Columns that the Floor Grid will have

Location Attributes

All Locations have 2 attributes:

- ***id***: Sequential number
- ***description***: It is a String that allows the entity to be unambiguously located in the location hierarchy. Each description is formed from the description of its father by adding its information.

  For example:

  b0_r1_c0_a0A_f1

  That is the description of the bed 0 from room 1 from corridor 0 from area 0A from floor 1.

Attributes of Services and Hospitalization Units

Services and Hospitalization Units have 3 attributes:

- ***id***: Sequential number
- ***abrev***: It is similar to the Location descriptions.
  Services are numbered with [0,7]
  Hospitalization Units with letters
- ***description***: Extended version of the abbreviations

  For example:

  S2 : Service 2

  HU2A : HospUnit 2A → First Hospitalization Unit from Service 2

Steps to generate hospitals

1. Create the 4 Special Services and their Hospitalization Unit:

Service	Hospitalization Unit
Emergency Service (*ER*)	Emergency Unit (*ERU*)
Intensive Care Unit Service (*ICU*)	Intensive Care Unit (*ICUU*)
Radiology Service (*RAD*)	Radiology Unit (*RADU*)
Surgery Service (*SURG*)	Surgery Unit (*SURGU*)

2. Parse the file with the H-outbreak hospital. Here the corresponding entities are created, they are related to each other, their id is assigned and the following additional actions are performed:

1. For each normal Service (S0-S7), as many Hospitalization Units are created as indicated in *huPerService*.
2. Each *ER Room* is connected with the ERU Hospitalization Unit
3. Each *ICU Room* is connected with the *ICUU Hospitalization Unit*
4. A Room is created for each *Surgery Bed*, and the Room is associated with *SURGU*
5. A Room is created for each *Radiology Bed*, and the Room is associated with *RADU*

3. Distribute the Rooms of each service (S0-S7) equally among its Hospitalization Units.

- If a Service has more Hospitalization Units than Rooms, all of them are assigned to *Unit A* and the rest of the Units are eliminated.

4. Calculate how many floors to create.

- On Floor 0 there will only be the *ER*, *ICU*, *RAD* and *SURG Rooms*
- The number of Floors for the rest of the Services is calculated based on the parameter *n_huPerFloor*.
  - Initially, each Plant will have *n_huPerFloor* Units.
  - But the user is also asked how many Plants he wants.
  - If *n_huPerFloor * n_floors* is different from the number of Hospitalization Units, the user is asked to choose which parameter to maintain. The other will be modified accordingly.
  - If the resulting *n_huPerFloor* is not an exact value, an extra Floor is NOT created with the leftover Units, but rather they are assigned to Floor 1

5. Create Floor 0 with the following layout:

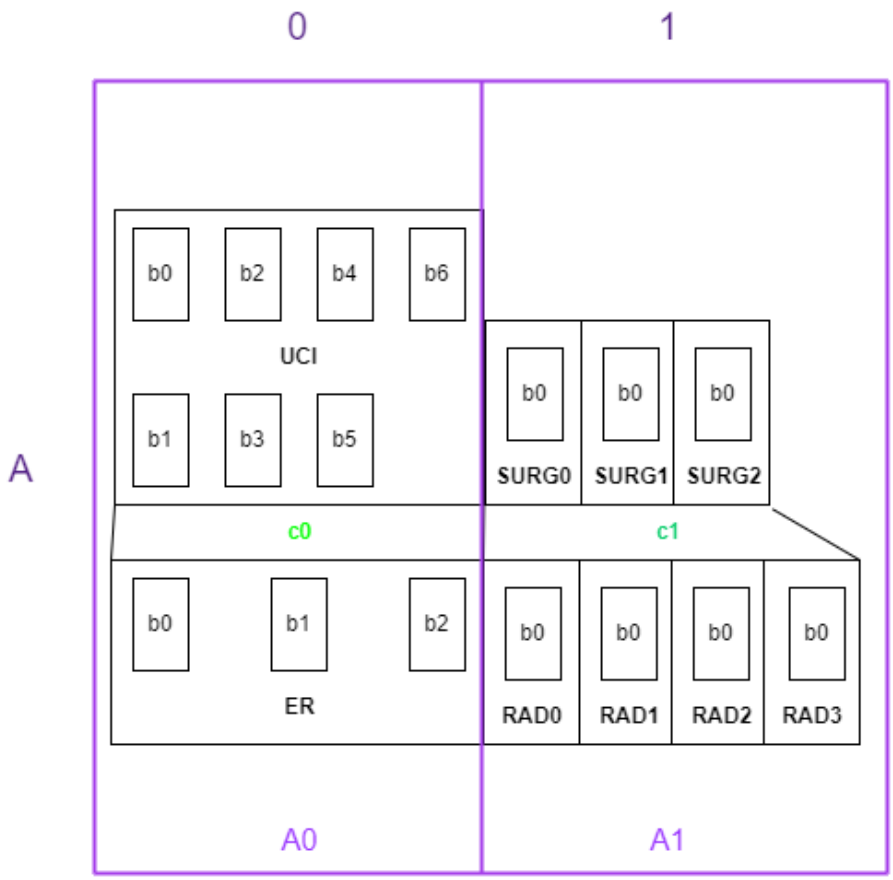


Figure 1 - Grid of Floor 0 with its Corridors, Rooms and Beds

6. Create the rest of the Floors.
Each Floor is based on a grid with as many rows and columns as specified by the user.
Each grid cell is an Area.

The process of creating each Floor is based on choosing the layout of the Corridors and Rooms of each of the Areas of the grid.

4 types of basic parts have been designed for the cells according to the position of their corridors. The number of types of pieces has been expanded by rotating the pieces by 90º, 180º and 270º. In addition, variations of each piece have been created with different numbers of rooms in each of its corridors.

- In short, each piece has:
  - A fixed number and arrangement of Corridors
  - A fixed number of Rooms
  - A fixed arrangement of Rooms in the Corridors

- Pieces nomenclature:
  - The Types of pieces (layout of the Halls) are named with CAPITAL LETTERS.
  - The Variants of each Type (number of rooms) are named with NUMBERS.

  For example:
  B0: It is a type B piece, with the smallest number of rooms
  I2: It is a piece with the third variant of type I

The 66 pieces that have been created are shown below:

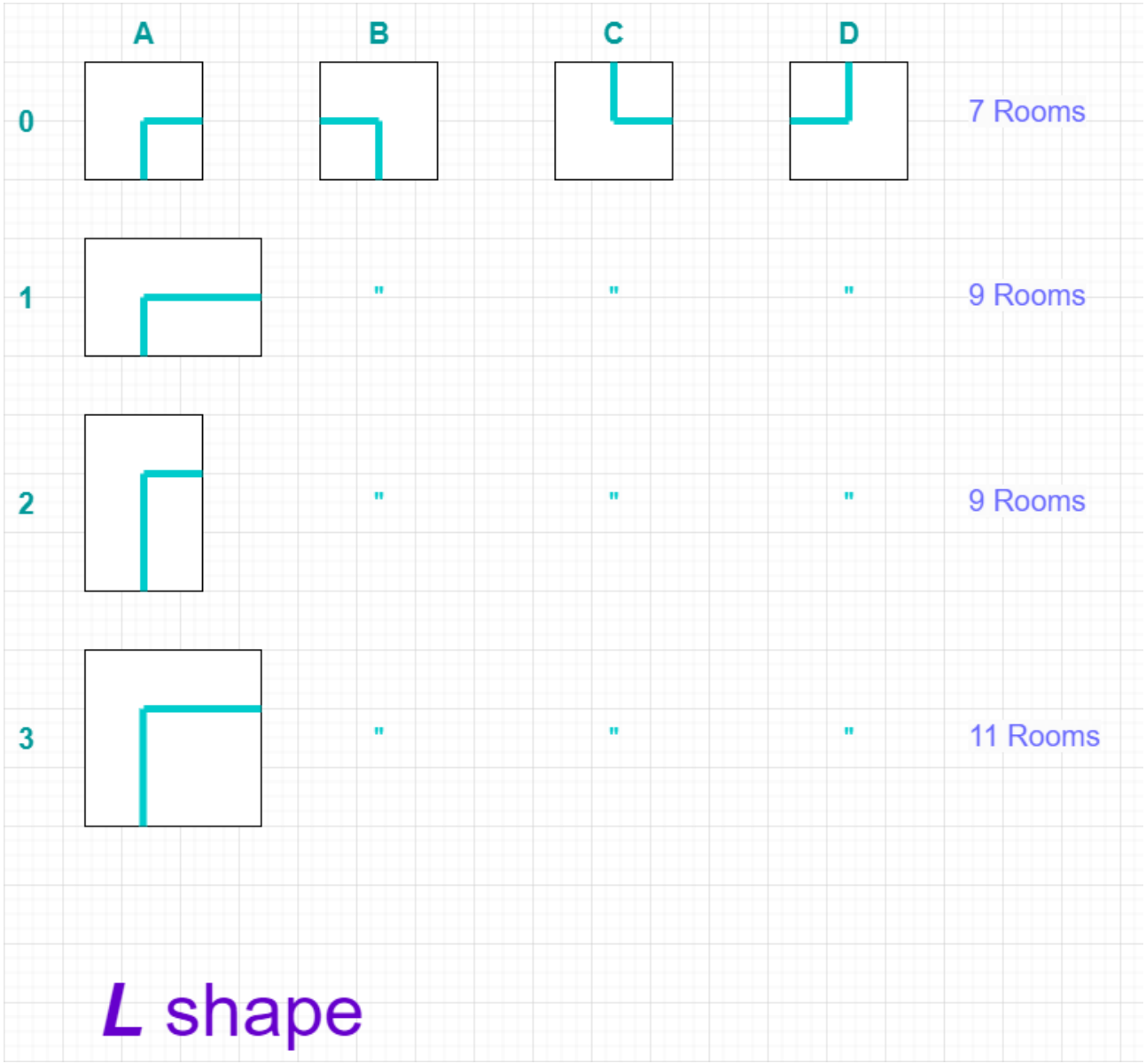


Figure 2 - L-shaped pieces (types A-D)

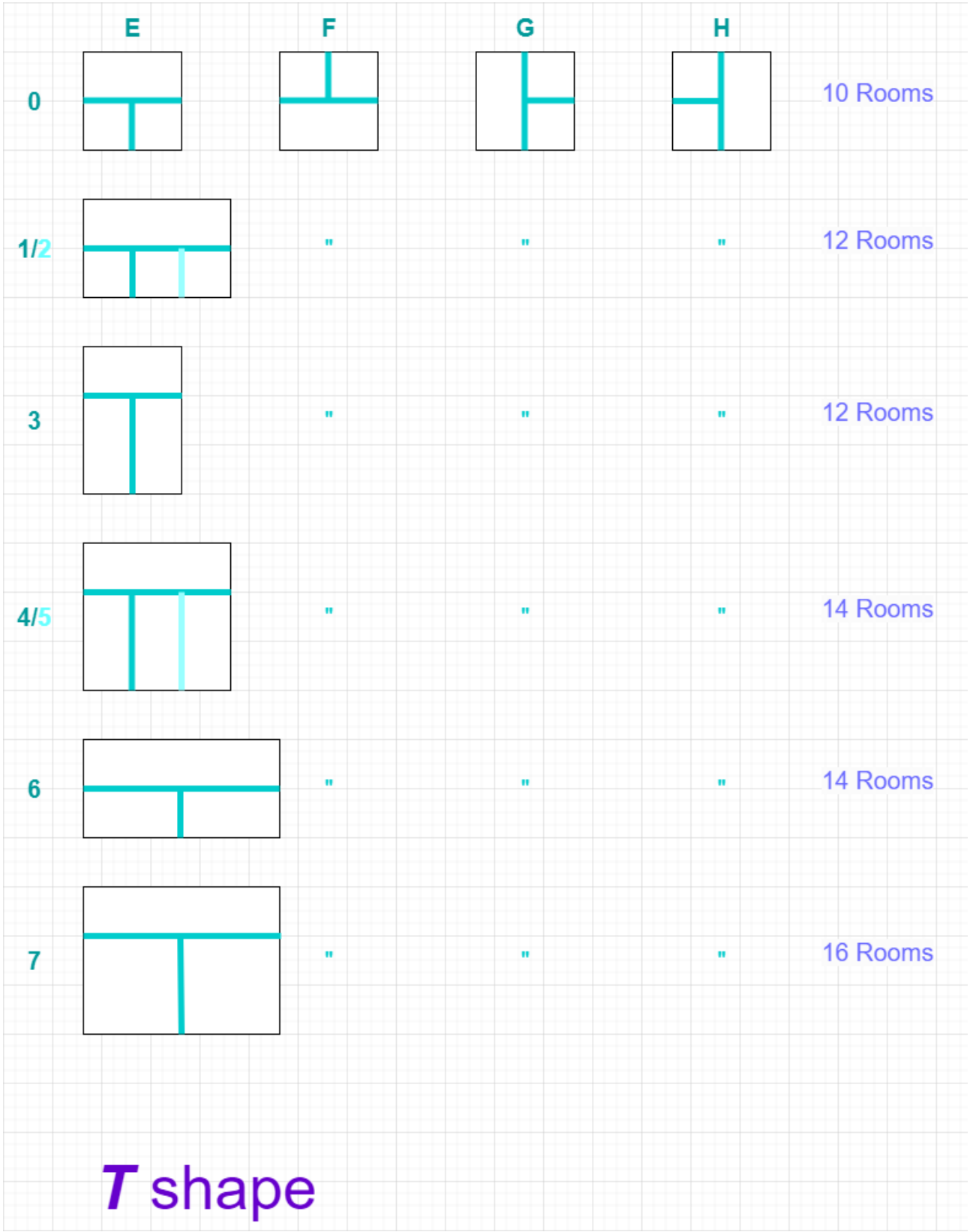


Figure 3 - T-shaped pieces (yipes E-H)

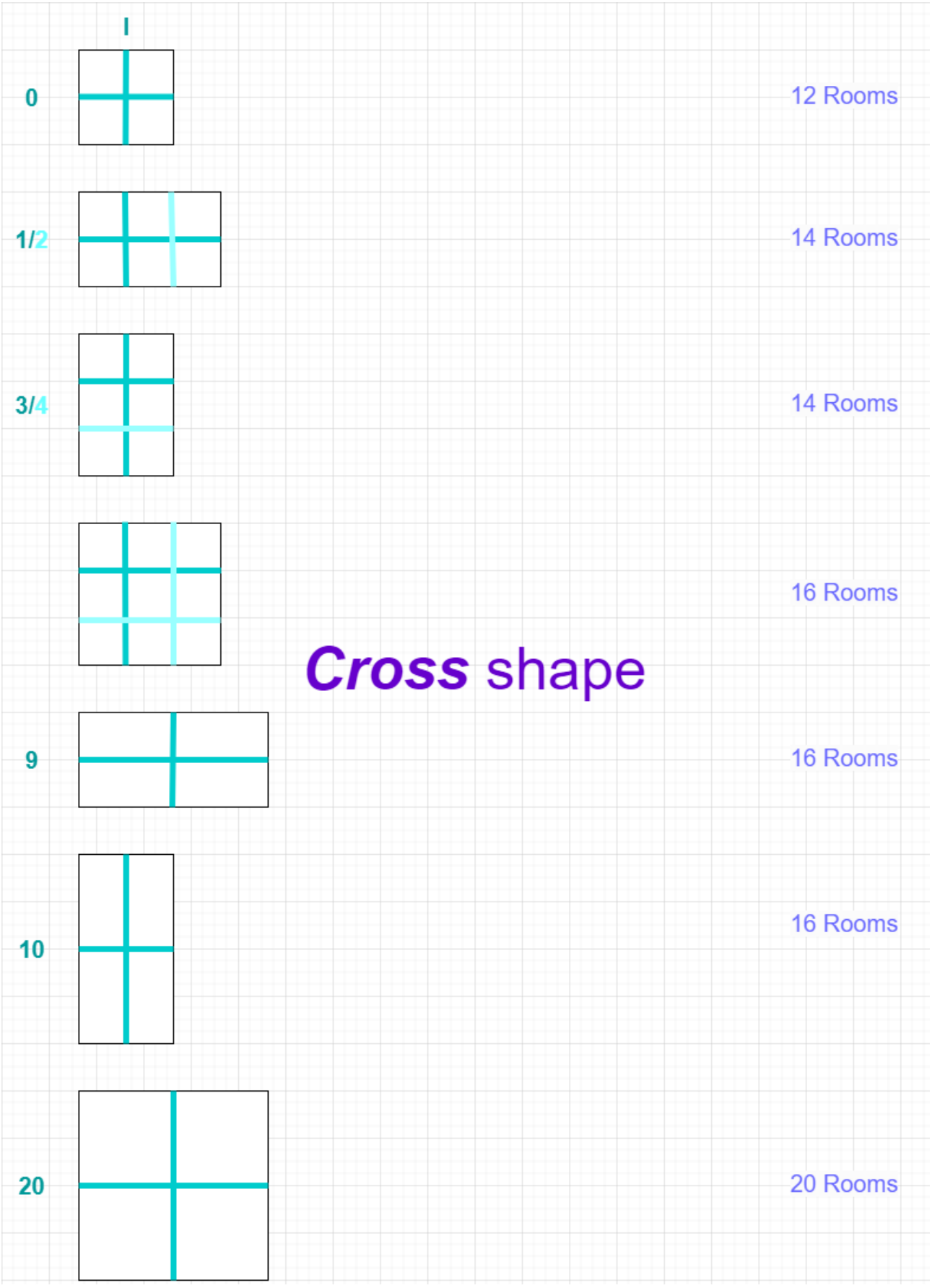


Figure 4 - Cross-shaped pieces (type I)

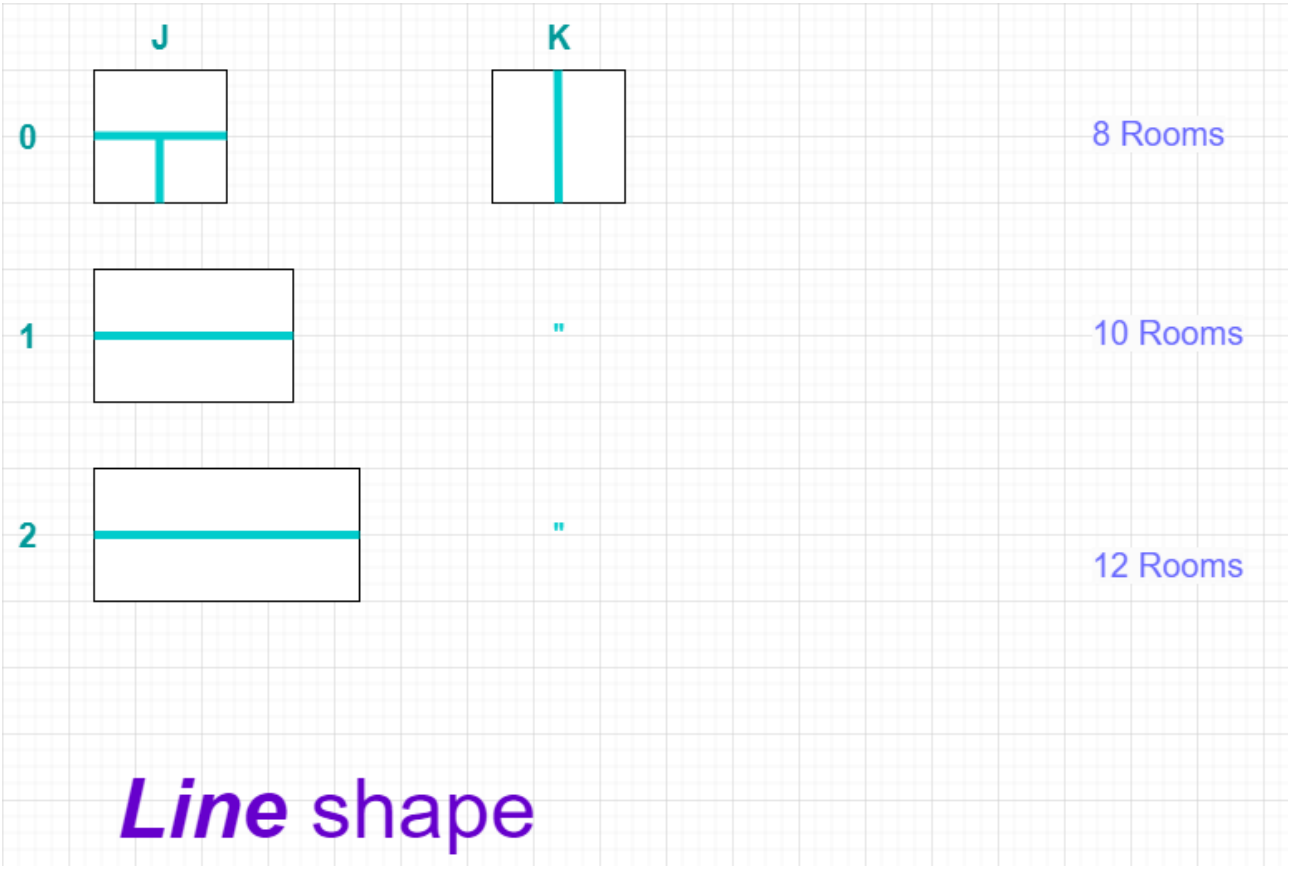


Figure 5 - Line shaped pieces (J-K types)

The selection of which piece to use in each cell of the grid is done through backtracking, which will give as a solution a semi-random configuration whose number of rooms is equal to the number of rooms that the floor must have.

The bactracking algorithm runs through all the cells as if they were in a horizontal line. In each "step" of bactracking the piece to put in the cell is chosen. The pieces tested for the cell range from those with the fewest Rooms to those with the most. As there are pieces that have the same number of Rooms, a set is made with all of them and only one of them is chosen randomly.

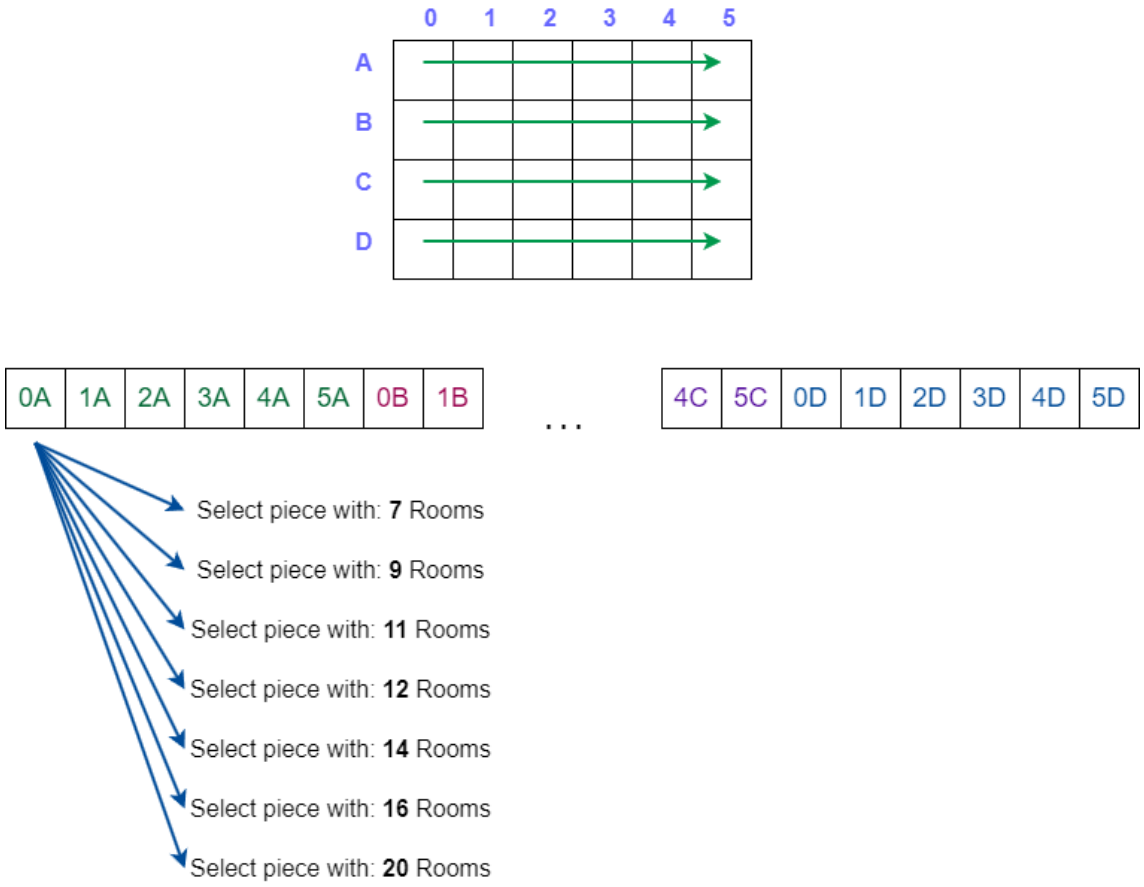


Figure 6 - Schematic representation of the backtracking algorithm

The algorithm will try to make the grid size *nRows* × *nColumns*. However, if the number of rooms is greater than the maximum (or less than the minimum) number of rooms possible with that size, then the algorithm will try larger (or smaller) grid sizes.

The configuration is *semi-random* because there are a set of restrictions on the type of piece that can be put in some cells. The basis of the restrictions is that every Corridor must be connected to at least one other Corridor.

Different restrictions have been created depending on the number of Rows in the grid.

1 Row	
Cell	**Available piece types**
First cell	A, E, G, J, I
Last cell	B, E, H, J, I
Rest of cells	E, I, J
2 Rows	
Cell	**Available piece types**
Top-left corner	A, E, G, I
Top-right corner	B, E, H, I
Bottom-left corner	C, F, G, I
Bottom-right corner	D, F, H, I
Rest of Top Row	E, I, J
Rest of Bottom Row	F, I, J

> 2 Rows	
Cell	**Available piece types**
Top-left corner	A, E, G, I
Top-right corner	B, E, H, I
Bottom-left corner	C, F, G, I
Bottom-right corner	D, F, H, I
Rest of Top Row	E, I, J
Rest of Bottom Row	F, I, J
Rest of Left Column	G, I
Rest of Right Column	H, I
Central cells	J, E, F

These restrictions can be summarized as follows:

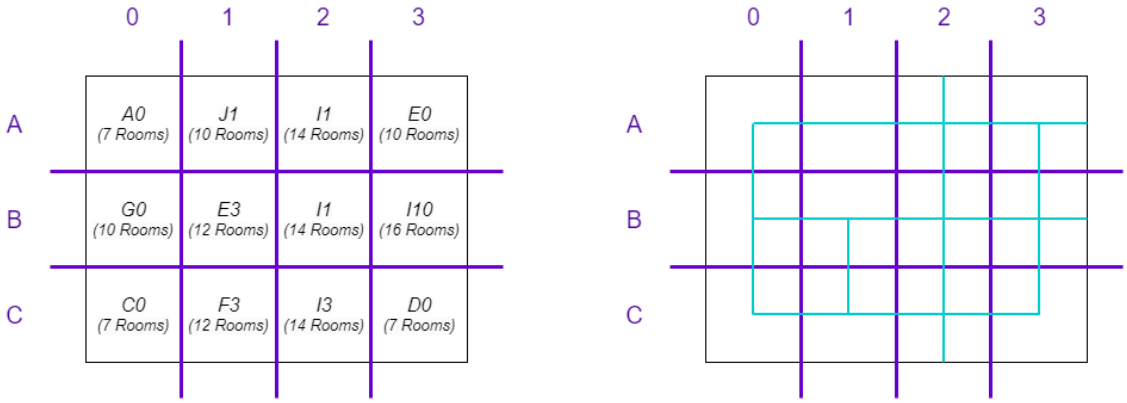
- If the piece in the cell immediately above has a Down Corridor, the piece chosen for the current cell must have an Up Corridor. The down corridor will be optional.
- All pieces that are not in the first or last column must be crossed from left to right by horizontal corridors.
- Pieces that are in the first or last column can have a single horizontal corridor, pointing towards the interior of the grid.

The result is an array of Pieces whose sum of Rooms gives what is required for the Floor.

If a configuration with the desired size cannot be found, an error will be returned and *HospitalGeneratorRDF* will close.

If the Floor has less than **7** rooms, an error will also be returned.

The minimum number of Rooms that a Floor must have for the backtracking algorithm to start is **21**. If it is **between 7 and 21**, then a base case will be returned.



Rooms = 7+10+14+10+10+12+14+16+7+12+14+7 = **133**

Figure 7 - Configuration example for a 133 Room Floor with 3 rows and 4 columns

NOTE: *Rows* correspond to **Unit**

Columns correspond to **Block**

NOTE: The area nomenclature is:

*a**B**lock**U**nit*

- **Blocks** are named with NUMBERS.
- **Units** are named with CAPITAL LETTERS

For example:

a*2D*: Area located in **Unit 2 Block D**, that is, *third Column* and *fourth Row*

7. Once you have the structure of the Areas of each Floor, the Corridors are created following the scheme of each selected Piece.
An Area will have a maximum of 4 hallways: *Left*, *Right*, *Top* and *Bottom*.
The Corridors of an Area will be numbered following this order. If any of them are missing, the next one will take their number.

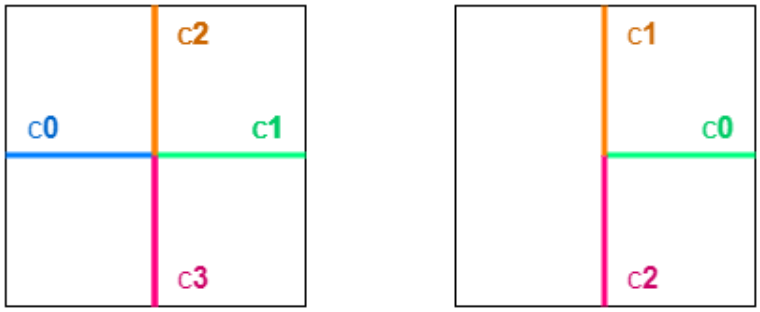


Figure 8 - Examples of hallway numbering

Each Corridor keeps who its neighboring Corridors are and their relative position. This is coded with the following numbering:

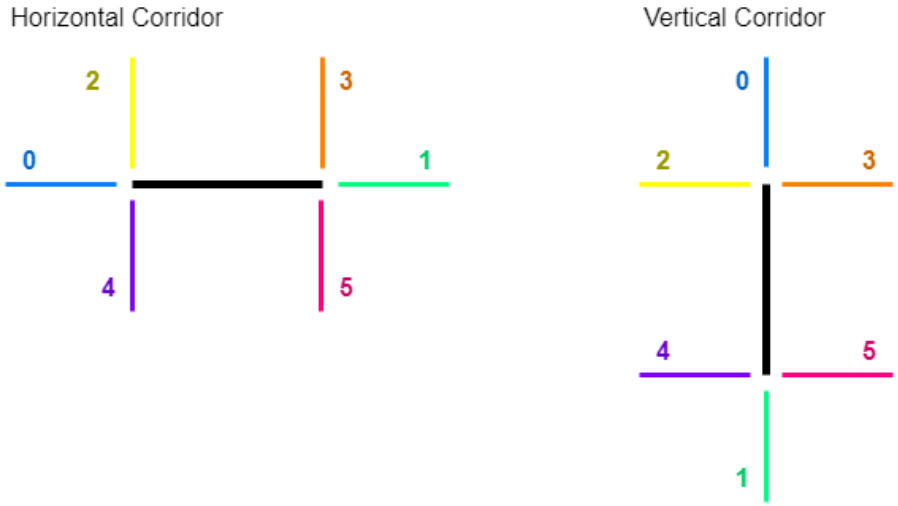


Figure 9 - Numbering of the positions in which a Corridor can be a neighbor of another

8. Assign each Area its Rooms.
The Areas are traversed from left to right and from top to bottom.

Each Piece has a fixed arrangement of Rooms, making it easy to assign to each Room which Rooms are next to it and which are opposite it.

Two Rooms will only be considered adjacent if they belong to the same Corridor or if they are in adjacent Corridors. Similarly, two Rooms will only be considered opposite if they belong to the same Corridor.
Two Rooms will belong to the same Corridor when their doors open towards the same corridor.

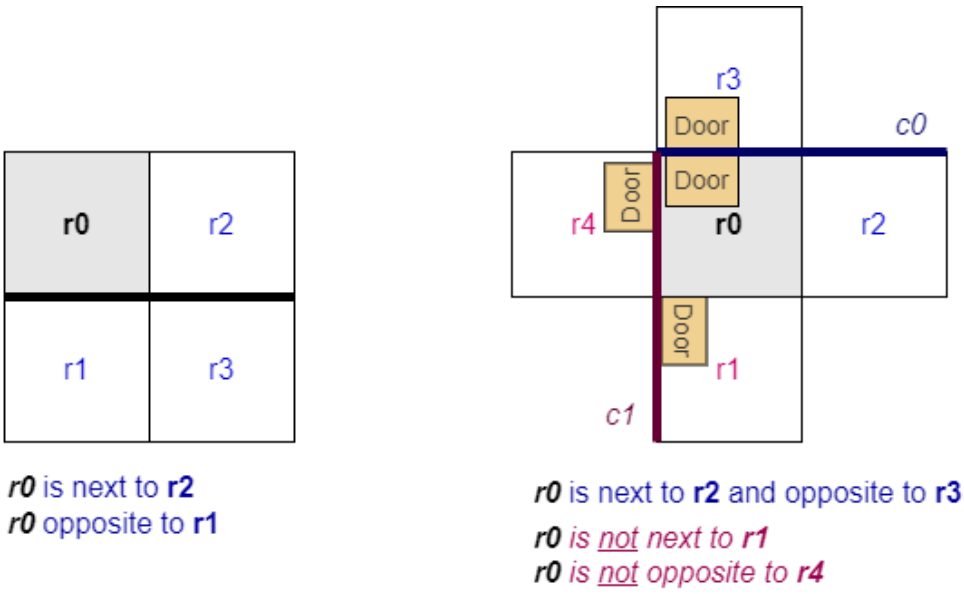


Figure 10 - Example of rooms that are (or not) next to and opposite

The nomenclature of the Rooms is not local to the Corridor, but global to the Area.

For example: `r5_c2_a0A_f1`

- It is not Room 5 of Hall 2 of Area 0A from Floor 1
- It is Room 5 of Area 0A from Floor 1 and is located in Corridor 2

Rooms are numbered following this order:

1. Top corridor: *Left → Right* and *Top → Bottom*
2. Bottom corridor: *Left → Right* and *Top → Bottom*
3. Left corridor: *Top → Bottom* and *Left → Right*
4. Right corridor: *Top → Bottom* and *Left → Right*

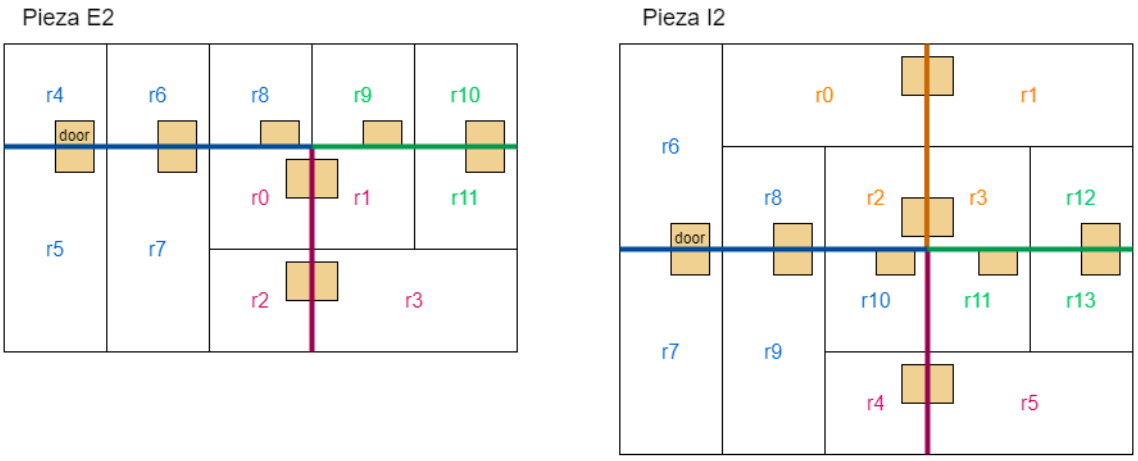
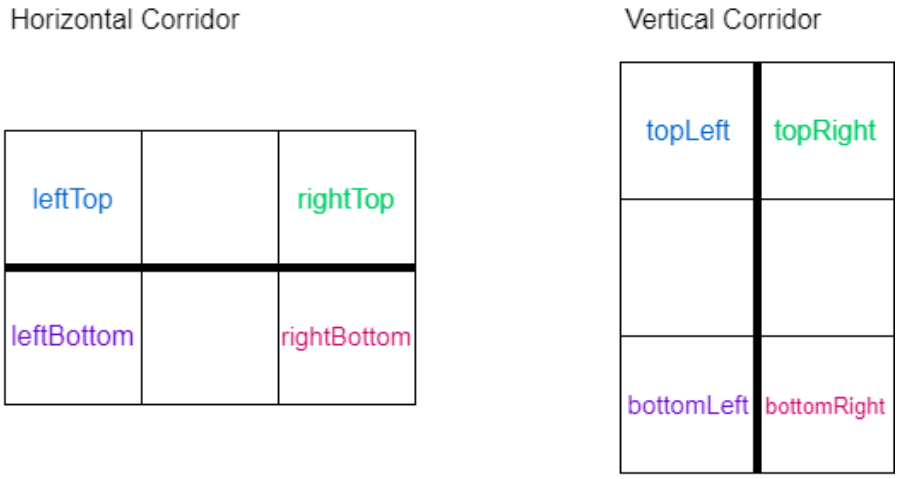


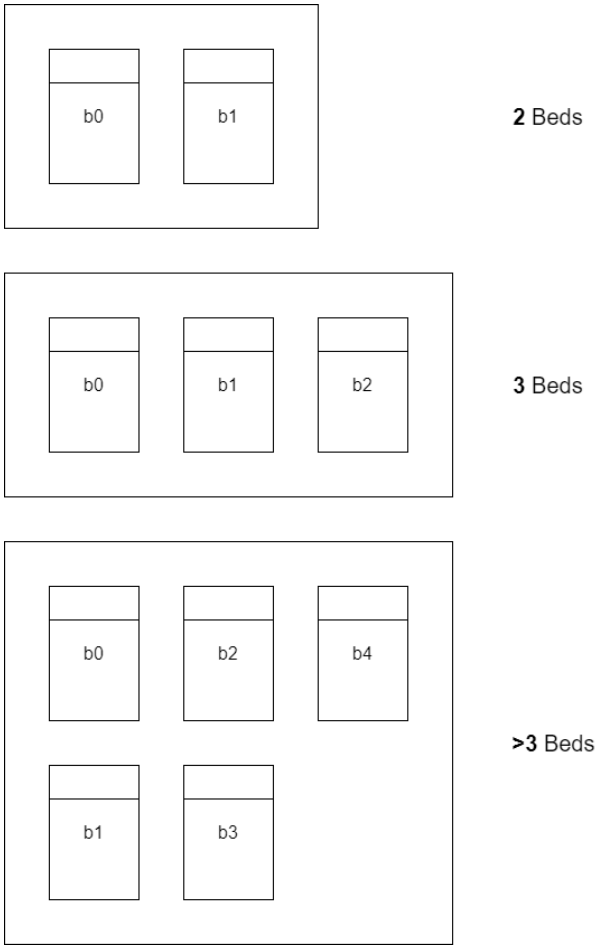
Figure 11 - Example of pieces E2 and I2 with the numbering of their Rooms

Information is also added to the Corridors about which Rooms they have at their ends, with the following nomenclature:



9. As each Room already has its Beds (from the entry parse), now the program put the Beds in their position within the Room, indicating which Beds are next to and opposite.

There are 3 types of layouts according to the number of beds:



10. Create the files with the nodes and edges of the graph that represents the hospital.