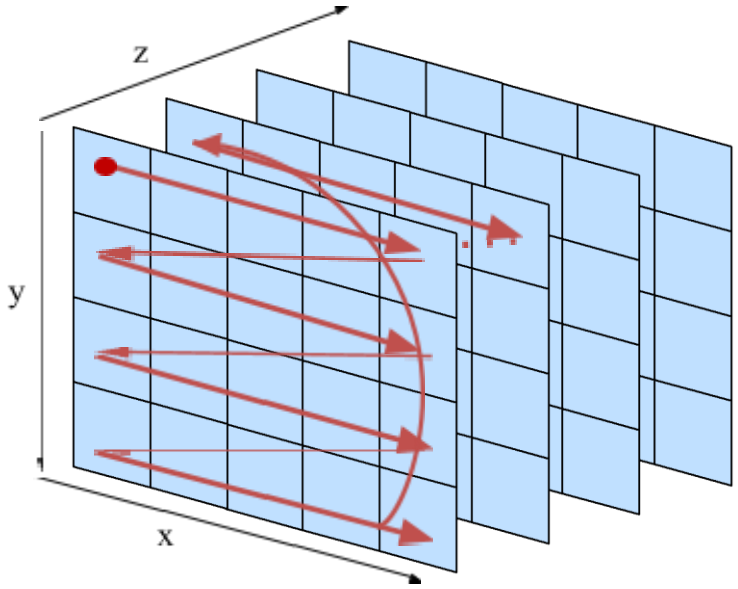
Matrix3D

The project requires the design and implementation of a template **Matrix3D** class which implements a 3-dimensional matrix of cells containing data of type **T**. The dimension of the matrix is chosen by the user during the construction phase of the object. Each cell can be accessed by giving the coordinates of the plane (z), row (y) and column (x) in this order.



The **Matrix3D** class, in addition to the methods necessary for its correct use (such as, for example, knowing the number of rows, columns, floors, ...), must have the following functionalities:

- It must be possible to convert a **Matrix3D** defined on a type **U** to a **Matrix3D** defined on a type **T**. Obviously the conversion will be possible if an element of type **U** is convertible/castable to an element of type **T**.

- It must be possible to read and write the value of a cell at position (z,y,x) via **operator()**. Ex: G(1,2,3).

- The appropriate read and write iterators must be implemented to access all the values contained in the matrix. The values are iterated in the order indicated by the red arrows in the drawing

- A method must be implemented, **slice** which returns a **sub-Matrix3D** containing the values in the ranges of coordinates z1..z2, y1..y2 and x1..x2.

- An **operator==** method must be implemented to verify that two **Matrix3D** objects contain the same values.

- A **fill** method must be implemented which allows to fill a **Matrix3D** with values taken from a sequence of data identified by generic iterators. Filling must occur in the order of iteration of the array data. The old values will be overwritten.

A generic function **transform** must also be implemented which, given a Matrix3D **A** (on types **T**) and a generic functor **F**, returns a new Matrix3D **B** (on types **Q**) whose elements are obtained by applying the functor to the elements of **A**:

B(i,j,k) = F(A(i,j,k))

Test the class and the global function both on primitive types and on custom types and with different functors F.

Use error handling via assertions or exceptions where appropriate.

Note 1: In the design of the class, the use of external libraries and container data structures of the **std library** such as **std::vector**, **std::list** and the like is prohibited. Their use in test code in main is allowed.

Note 2: Apart from **nullptr**, you cannot use any other **C++11** and later constructs.