Schools

Develop an application to allow managing the info about schools in a region.
The application should allow the entry of information about municipalities, schools, branches, and communities. All the classes must be in the package "**`Schools`**".

R1 - Municipalities and Communities

The interaction with the system takes place through class **`Region`**.
The definition of a community can be performed using the **`newCommunity()`** function that accepts as arguments the name and an *`enum`* **`Type`** indicating whether it is a *`HILL`* or *`MOUNTAIN`* community.
The method returns a **`Community`** object.
The definition of a new municipality is done with the function **`newMunicipality()`** accepting as arguments the name, the province, the AFCode, the AFDescription, and an optional reference to the *`Community`* of belonging. The function returns a *`Municipality`* object. Both classes offer the getter methods for the data they are initialized with.
The collection of all municipalities and communities can be retrieve using the functions **`getMunicipalities()`** and **`getCommunities()`** respectively.
In addition, given a *`Municipality`* the relative community can be retrieved using the function **`getCommunity()`**, that returns an optional **`Community?`**, which is empty if the municipality does not belong to any community. Viceversa, given a *`Community`* the collection of municipalities can be obtained with function **`getMunicipalities()`**.

R2 - Schools and branches

Schools are defined using the function **`newSchool()`** that accepts as arguments, the name, the state code (unique), the school grade, the description, the management type, and the legal position.
The method returns an object of class **`School`**, which provides the required getter functions.
Schools can have one or more branches defines by means of **`newBranch()`** that accepts as arguments the unique regional code, the type of the branch, a reference to the *`Municipality`* where the branch is located, the address, the zip code, the locality, the telephone, the fax, and a reference to the School the branch belongs to.
The function returns a **`Branch`** object.
The class *`Branch`* provides the getter functions relative to the parameter used to create it; in particular it offers the functions **`getMunicipality()`** and **`getSchool()`** to retrieve the municipality where it is located and the school it belongs to.
The class *`School`* provides the function **`getBranches()`** to retrieve the collection of branches.

R3 - Queries

The function **`countSchoolsPerDescription()`** returns a dictionary containing as key the school description and as value the number of schools corresponding to that description.
The function **`countBranchesPerMunicipality()`** returns a dictionary containing as key the name of the municipality and as value the number of branches belonging to it.
The function **`averageBranchesPerMunicipality()`** returns a dictionary containing as key the name of the province and as value the average number of branches in the municipalities of that province.
The function **`countSchoolsPerMunicipality()`** returns a list of strings, each with the format "*`### - XXXXXX`*" where *`###`* represents the number of schools and *`XXXXXX`* represents the name of the municipality.
The function **`countSchoolsPerCommunity()`** returns a list of strings, each with the format "*`### - XXXXXX`*" where *`###`* represents the number of schools and *`XXXXXX`* represents the name of the community. sorted in decreasing ordered by number of schools. Only the schools having at least a branch in a municipality that belongs to a community.