Test plan

geogen::GeoGrid::GeoGrid():

Test Title	Test summary	Test steps	Test data	Expected result
Happy day GeoGen constructor.	Tests if a geogen is constructed properly if the input files are correct.	1) run popgen-sim with default_geogen-config.xml	- default_geogen-config.xml -flanders_cities.csv	Variables initialized according to the config file. Cities all loaded into GeoGen with correct data.
Faulty row Geogen	Tests if one of the input's on the csv list is wrong	1) run popgen-sim with default_geogen-config.xml	- default_geogen-config.xml flanders_cities_faulty.csv	Skips row of csv file, outputs warning that row is faulty. Continue's running GeoGen
Faulty column structure city csv extra column	checks expected result, if we have an extra unwanted column in the csv	1) run popgen-sim with default_geogen-config.xml	- default_geogen-config.xml flanders_cities_faulty_extra.csv	Extra column get's ignored.
Faulty column structure city csv file, lost a column	Checks expected result if we miss a city column.	1) run popgen-sim with default_geogen-config.xml	- default_geogen-config.xml flanders_cities_faulty_less.csv	Geogen terminated, missing data.
Happy day row count	Count amount of csv rows, check len map	1) run popgen-sim with default_geogen-config.xml 2) make a unit test in test enviorment and assert equal	- default_geogen-config.xml -flanders_cities.csv	Assert equal csv rows and map size city's in unit test
Antwerp correct data test	Check if the data filled in for id Antwerp in city's is correct	1) run popgen-sim with default_geogen-config.xml 2) make a unit test in test environment and assert equal to expected value's of antwerp	- default_geogen-config.xml -flanders_cities.csv	Test should pass all data should be as expected data

Test plan

geogen::GeoGrid::generate_schools(...):

Test Title	Test summary	Test steps	Test data	Expected result
Happy day, generate schools	Generate schools with all correct parameters. -schooled fractal: 0.7 -random seed 0	1) run popgen-sim with default_geogen-config.xml	- default_geogen-config.xml	6058 schools created, placed according to discrete distribution in the city's
Negative total pop	Total population is negative	1) run popgen-sim with default_geogen-config.xml	- default_geogen-config.xml	Exception thrown value error, negative pop
schooled fractal is above 100%	schooled fractal is above 100%	1) run popgen-sim with default_geogen-config.xml	- default_geogen-config.xml	Exception thrown value error, schooled fractal above 100%
Negative school size	School size is negative	1) run popgen-sim with default_geogen-config.xml	- default_geogen-config.xml	Exception thrown value error, school size is negative
Hand calculated total schools	Calculate total schools to be placed, check against schools generated	1) run popgen-sim with default_geogen-config.xml 2) check school vec size against hand calculated school amount	default_geogen-config.xml	Set boundry in unit test enviroment, and assert equal value's, value's should be equal
High populated vs low populated school amount	Check if the 5 highest populated cities have more schools then the 5 lowest	1) run popgen-sim with default_geogen-config.xml 2) check in unit test if the 5 highest populated cities have more schools then the lowest	default_geogen-config.xml	Highest populated(5).school_amount >Lowest populated(5).school_amount >

Test plan

geogen::GeoGrid::generate_colleges():

Test Title	Test summary	Test steps	Test data	Expected result
Happy day scenario for generate_colleges(N) default value for N =10	Tests if colleges are assigned to the correct cities and the number of generated colleges for each of those cities.	- Read input - Find N largest cities - Generate colleges according to fraction of students. - Check against expected values	Happy day scenario so considering GeoGrid's constructor read a valid input file which is done by parser::parse_cities()	N largest cities should have X colleges where X= #students / 3000 #students is the number of students for a particular city.
Alternate scenario for generate_colleges(N)	Tests bad input or no input at all.	- Read input - Run as if it was a happy day scenario	Bad input file or no input at all	Again, largest cities should have X colleges, considering some cities were read correctly. If no cities are left due to bad input, no colleges should be present.
Unit test for adjustLargestCities(...)	Tests function used by generate_colleges() to ensure correct behavior.	- Add N cities - Add the N+1th city and check against expected behavior.	A number of manually created cities as well as a city that should be added.	The first N cities should be added immediately. The N+1th city should replace the city with the smallest population if that city has a bigger population.
Unit test for assignCollege(...)	Tests function used by generate_colleges() to ensure correct behavior.	- Generate colleges for a particular city. - Check against expected number of colleges.	A number of students or fraction of the population representing the students for the particular city.	Number of colleges should be equal to #students / 3000

findSmallest(...) will not be tested since this is a trivial function and it's being used by adjustLargestCities(…), therefore if the tests pass for adjustLargestCities(…), findSmallest(…) should be working correctly.

Test plan

geogen::GeoGrid::generate_communities():

Test Title	Test summary	Test steps	Test data	Expected result
Happy day generate_communities()	Tests if communities are generated correctly given a certain input.	- Read input - Generate communities - Check against expected values	Any legal input file containing the total population size, the number of cities and the population per city.	Cities with a larger population will have more communities than cities with a smaller population.
Equal_cities	Tests if all cities will contain the same number of communities if they have a population of equal size.	- Read input - Generate communities - Check against expected values	Any legal input file containing the total population size, the number of cities and the population per city.	Cities with equal populations have an equal number of communities.
Extreme_cities	If a city has a negligible population it will contain no communities.	- Read input - Generate communities - Check against expected values	Any legal input file containing the total population size, the number of cities and the population per city.	Cities with insignificant populations will have no communities.

Test plan

geogen::GeoGrid::generate_workplaces():

Test Title	Test summary	Test steps	Test data	Expected result
Happy day generate_workspaces()	Tests if workspaces are generated successfully with given required inputs	- Read input - Calculate the number of people working in a region = active population in the region - out commuters + in commuters - Generate workspaces - Check if the expected value is met	Any legal file(s) containing the information of the total population size, the population per region, the population distribution per age, commuting behaviour	Workspaces are spread according to the number of active population active in that region (active population + in commuters - out commuters) and the average workers for workspace is respected
Commuter vs local active population test	Tests if two regions have the same number of workspaces; if the both regions have the same number of active population but a region has active population without incoming and outgoing commuters and another region where all the active population consists of only incoming commuters (nobody from the own region)	- Read input - Calculate the number of people working in a region = active population in the region - out commuters + in commuters - Generate workspaces - Check if the expected value is met	Any legal file(s) containing the information of the total population size, the population per region, the population distribution per age, commuting behaviour	The same number of workspaces for the both regions
Extreme region	Test if a region with no effective active population will have workspaces	- Read input - Calculate the number of people working in a region = active population in the region - out commuters + in commuters - Generate workspaces - Check if the expected value is met	Any legal file(s) containing the information of the total population size, the population per region, the population distribution per age, commuting behaviour	That region will have no workspaces