TEST PLAN

March 31, 2019

Abstract

In this document we will go over all tests, including unit tests as well as scenario tests, documenting their name along with their purpose and how we intend to check this.

Benjamin Vandersmissen benjamin.vandersmissen@student.uantwerpen.be
Lars Van Roy lars.vanroy@student.uantwerpen.be
Evelien Daems evelien.daems@student.uantwerpen.be
Frank Jan Fekkes franciscus.fekkes@student.uantwerpen.be

Contents

1 Introduction

For the rest of this document we will give a short summary of every newly created test that will check if the new features work as we expected. These new additions include the two new contact types, being Daycare and PreSchool, new data formats, being JSON and HDF5 as well as global schemes that will be used by all students. There will also be a visual output for the data via QT and an improvement to the generation of the population by factoring in new parameters that will make the generation of the population more diverse, so that we can check even more parameters for their influence on the spread of diseases.

2 Daycare and PreSchool

2.1 Generators

2.1.1 DaycareGenerator.ZeroLocationTest

What will it test? This test will check whether the DaycareGenerator can function correctly with no locations given.

How will we achieve this? We will achieve this by creating a geogrid with no locations and handing it over to the Generator, then checking whether the size of the geogrid is zero.

2.1.2 DaycareGenerator.OneLocationTest

What will it test? This test will check whether the DaycareGenerator can function correctly with one location given.

How will we achieve this? We will achieve this by creating a geogrid with one location and handing it over to the Generator, then comparing the amount of contactPools for the location with the expected amount of contactPools.

2.1.3 DaycareGenerator.MultipleLocationtest

What will it test? This test will check whether the DaycareGenerator can function correctly with multiple locations given.

How will we achieve this? We will achieve this by creating a geogrid with multiple locations and handing it over to the Generator, then comparing the amount of contactPools for each location with the expected amount of contactPools in that location.

2.1.4 PreSchoolGenerator.ZeroLocationTest

What will it test? This test will check whether the PreSchoolGenerator can function correctly with no locations given.

How will we achieve this? We will achieve this by creating a geogrid with no locations and handing it over to the Generator, then checking whether the size of the geogrid is zero.

2.1.5 PreSchoolGenerator.OneLocationTest

What will it test? This test will check whether the PreSchoolGenerator can function correctly with one location given.

How will we achieve this? We will achieve this by creating a geogrid with one location and handing it over to the Generator, then comparing the amount of contactPools for the location with the expected amount of contactPools.

2.1.6 PreSchoolGenerator.MultipleLocationtest

What will it test? This test will check whether the PreSchoolGenerator can function correctly with multiple locations given.

How will we achieve this? We will achieve this by creating a geogrid with multiple locations and handing it over to the Generator, then comparing the amount of contactPools for each location with the expected amount of contactPools in that location.

2.2 Populators

2.2.1 DaycarePopulator.NoPopulation

What will it test? This test will check whether the DaycarePopulator doesn't crash when it gets a geogrid with an empty location.

How will we achieve this? We will achieve this by creating a geogrid with an empty location and passing it to the DaycarePopulator, then checking if no exceptions occur.

2.2.2 DaycarePopulator.OneLocationTest

What will it test? This test will check whether the DaycarePopulator is well-behaved when working with a geogrid with one location.

How will we achieve this? We will achieve this by creating a geogrid with one location and passing it to the DaycarePopulator, then comparing a number of parameters with the expected values, such as the amount of pools and the age range for each pool.

2.2.3 DaycarePopulator.TwoLocationTest

What will it test? This test will check whether the DaycarePopulator is well-behaved when working with a geogrid with two or more locations.

How will we achieve this? We will achieve this by creating a geogrid with two locations and passing it to the DaycarePopulator, then comparing a number of parameters for each location with the expected values, such as the amount of pools and the age range for each pool.

2.2.4 PreSchoolPopulator.NoPopulation

What will it test? This test will check whether the PreSchoolPopulator doesn't crash when it gets a geogrid with an empty location.

How will we achieve this? We will achieve this by creating a geogrid with an empty location and passing it to the PreSchoolPopulator, then checking if no exceptions occur.

2.2.5 PreSchoolPopulator.OneLocationTest

What will it test? This test will check whether the PreSchoolPopulator is well-behaved when working with a geogrid with one location.

How will we achieve this? We will achieve this by creating a geogrid with one location and passing it to the PreSchoolPopulator, then comparing a number of parameters with the expected values, such as the amount of pools and the age range for each pool.

2.2.6 PreSchoolPopulator.TwoLocationTest

What will it test? This test will check whether the PreSchoolPopulator is well-behaved when working with a geogrid with two or more locations.

How will we achieve this? We will achieve this by creating a geogrid with two locations and passing it to the PreSchoolPopulator, then comparing a number of parameters for each location with the expected values, such as the amount of pools and the age range for each pool.

3 Data formats

3.1 JSON

3.1.1 HouseholdJSONReader.validJSON

What will it test? This test will check is a given JSON file containing a household configuration is read properly and retrieved the expected values.

How will we achieve this? A JSON formatted string is given to the reader. Then all households are checked if they have the same size and if each household contain the right values.

3.1.2 HouseholdJSONReader.invalidJSON

What will it test? It will test if a wrongly formatted JSON is given to the household reader.

How will we achieve this? Given a wrongly formatted input, the reader function is called. Then it will be checked if an exception is throw and if it is the expected error that occured.

3.1.3 GeoGridJSONReaderTest.locationsTest

What will it test? It will test if the reader has extracted the right location information from a JSON file.

How will we achieve this? Given a JSON file we can read, the test will compare the ID's with the expected value and for each location all attributes will be tested on their contents.

3.1.4 GeoGridJSONReaderTest.commutesTest

What will it test? It will test if the reader has extracted the right commuting information from a JSON file.

How will we achieve this? Given a JSON file we can read, the test will consider all locations and in specific if their commutes object contains the right location ID and percentage of commuters.

3.1.5 GeoGridJSONReaderTest.contactPoolTest

What will it test? This test will check if each pool has the right type according to the internal coding of the different types of centers.
How will we achieve this? Given a JSON file we extract all the contactPools and their information and then we will check for each if the numeric class is available in the code. All types were specified in the JSON file and thus we can check if all types are compatible with their numerical values.

3.1.6 GeoGridJSONReaderTest.peopleTest

What will it test? This test will check if a person within a contactpool can be linked to a defined person with the same ID.
How will we achieve this? From a JSON formatted file we will check if

the person within a center has a pointer to the correct object with the right information by checking all its attributes againts the expected values.

3.1.7 GeoGridJSONReaderTest.intTest

What will it test? It will test if their occurs an error if a double is given in the JSON file.
How will we achieve this? The test simply follows the same steps as the peopleTest and sees if the error occurs or everything can be simply casted to the right types.

3.1.8 GeoGridJSONReaderTest.emptyStreamTest

What will it test? This test will examine if an idle input can be properly handled by the reader.
How will we achieve this? We will simply pass along an empty stream to the reader and observe its response in the form of an exception.

3.1.9 GeoGridJSONReaderTest.invalidTypeTest

What will it test? This test will examine if a correct response is produces when a wrong type is used for a contact pool in the JSON format.
How will we achieve this? A JSON file is given to the reader with a wrong type for a contactPool. Then it is checkes if an Exception has been throw to indicate something has gone wrong.

3.1.10 GeoGridJSONReaderTest.invalidPersonTest

What will it test? This test is it is possible to reference a non-existing person within a contact pool without defining this person with its attributes.
How will we achieve this? A JSON file with a wrong person in a contact pool is given to the reader to test its response. If it returns an Exception indicating something has gone wrong the test will succeed.

3.1.11 GeoGridJSONReaderTest.invalidJSONTest

What will it test? A test to see what will happen if a a random word is given within a JSON formatted file.
How will we achieve this? By passing a random string within a JSON file the test will determine if a proper Exception is thrown and the execution is aborted.

3.1.12 GeoGridJSONWriterTest.locationTest

What will it test? A test to determine if the location information retrieved from a GeoGrid is correctly formatted in a JSON file.

How will we achieve this? The test will create a GeoGrid and passes this information to the writer which will create a JSON formatted output. The test will convert this JSON and a comparison file to an XML format and check if they are the same.

3.1.13 GeoGridJSONWriterTest.contactPoolTest

What will it test? A test to determine if the contact pool information retrieved from a GeoGrid is correctly formatted in a JSON file.

How will we achieve this? The test will create a GeoGrid and passes this information to the writer which will create a JSON formatted output. The test will convert this JSON and a comparison file to an XML format and check if they are the same.

3.1.14 GeoGridJSONWriterTest.peopleTest

What will it test? This test will look in specific if the people are correctly represented in the resulting JSON file.

How will we achieve this? The test will create a population and passes this information to the writer which will create a JSON formatted output. The test will convert this JSON and a comparison file to an XML format and check if they are the same.

3.1.15 GeoGridJSONWriterTest.commutesTest

What will it test? This test will look in specific if the commuting information is correctly represented in the written JSON file.

How will we achieve this? The test will create a population and passes this information to the writer which will create a JSON formatted output. The test will convert this JSON and a comparison file to an XML format and check if they are the same.

3.2 HDF5

3.2.1 HDF5Reader.locationsTest

What will it test? This test will check whether the locations are read correctly from the HDF5 file.

How will we achieve this? We will achieve this by comparing the locations read from the HDF5 input file with the correct locations.

3.2.2 HDF5Reader.commutesTest

What will it test? This test will check whether the commutes are read correctly from the HDF5 file.

How will we achieve this? We will achieve this by comparing the commutes read from the HDF5 input file with the correct commutes.

3.2.3 HDF5Reader.contactCentersTest

What will it test? This test will check whether the contactCenters are read correctly from the HDF5 file.

How will we achieve this? We will achieve this by comparing the contactCenters read from the HDF5 input file with the correct contactCenters.

3.2.4 HDF5Reader.peopleTest

What will it test? This test will check whether the people are read correctly from the HDF5 file.

How will we achieve this? We will achieve this by comparing the people read from the HDF5 input file with the correct people.

3.2.5 HDF5Reader.invalidHDF5Test

What will it test? This test will check whether the HDF5Reader accepts invalid formed files or not.

How will we achieve this? We will achieve this by feeding the HDF5Reader a number of invalid files and check whether or not an exception is thrown by the HDF5Reader.

3.2.6 HDF5Writer.locationsTest

What will it test? This test will check whether the locations are written correctly to the HDF5 file.

How will we achieve this? We will achieve this by comparing the resulting HDF5 file with the correct HDF5 file.

3.2.7 HDF5Writer.commutesTest

What will it test? This test will check whether the commutes are written correctly to the HDF5 file.

How will we achieve this? We will achieve this by comparing the resulting HDF5 file with the correct HDF5 file.

3.2.8 HDF5Writer.contactCentersTest

What will it test? This test will check whether the contactCenters are written correctly to the HDF5 file.

How will we achieve this? We will achieve this by comparing the resulting HDF5 file with the correct HDF5 file.

3.2.9 HDF5Writer.peopleTest

What will it test? This test will check whether the people are written correctly to the HDF5 file.

How will we achieve this? We will achieve this by comparing the resulting HDF5 file with the correct HDF5 file.

4 QT

No tests will be made for QT, as this is hard to test via code. The testing of QT will happen via simulations done by ourselves and outsiders checking if our interface is obvious and well made

5 Improved population generation

5.1 Demographic profile

5.1.1 DemographicProfile.age

What will it test? This test will check whether a given demographic profile for the young/old ratio is used.

How will we achieve this? We will achieve this by generating a population using a demographic profile for these ratios. We can check if the profile was used by comparing the generated population with the expected percentages.

5.1.2 DemographicProfile.cities

What will it test? This test will check whether a given demographic profile for the distribution of central cities compared to other cities is used.

How will we achieve this? We will achieve this by generating a population using a demographic profile for the different central cities versus other cities. We can check if the profile was used by comparing the generated population with the expected percentages.

5.2 Workplace contactpools

5.2.1 WorkplaceCSVReader.Test1

What will it test? This test will check whether we are able to read Workplace distribution CSV's.

How will we achieve this? We will achieve this by passing a possible CSV configuration to the reader. The returned format can then be checked in order to ensure that the configuration is read correctly.

5.2.2 WorkplaceSizePopulatorTest.NoPopulationTest

What will it test? The test will check whether the workplace size populator still works when there is no population available.

How will we achieve this? We can easily check this by passing a geogrid to the populator who has a population size of 0. If there are no exception thrown we know that the function did finish correctly.

5.2.3 WorkplaceSizePopulatorTest.OneWorkplaceTypeTest

What will it test? The test will check whether the workplace size populator can handle one single workplace type.

How will we achieve this? We can achieve this by passing a single workplace size to populator. We can check if this process was successful by analyzing the altered object.

5.2.4 WorkplaceSizePopulatorTest.TwoWorkplaceTypesTest

What will it test? The test will check whether the workplace size populator can handle two workplace types.

How will we achieve this? We can achieve this by passing two workplace sizes to the populator. In order to check if this was successful we can analyze the altered object by checking whether there are two different pools.

5.2.5 WorkplaceSizePopulatorTest.MultipleWorkplaceTypesTest

What will it test? This test will check whether the workplace size populator can handle multiple types and sizes.

How will we achieve this? We will achieve this by altering the configuration of the workplace sizes in such way that there are multiple. We can check if this was successful by analyzing the new populations data.

5.2.6 WorkplaceSizeGeneratorTest.OneWorkplaceTypeTest

What will it test? This test will check whether the workplace size generator can handle one single type of workplace.

How will we achieve this? We can achieve this by passing one single type to the generator. We can check whether this was successful by checking the resulting populations data.

5.2.7 WorkplaceSizeGeneratorTest.ZeroWorkplaceTypesTest

What will it test? This test will check whether the workplace size generator can handle zero types of workplaces.

How will we achieve this? We can achieve this by passing no types to the generator. To check if this was successful we can check whether the generated population has any workplace data.

5.2.8 WorkplaceSizeGeneratorTest.FiveWorkplaceTypesTest

What will it test? This test will check whether the workplace size generator can handle multiple types of workplaces.

How will we achieve this? We achieve this by passing five different types to the generator. The generated population will, if the process was successful, have five different workplaces, all with their expected sizes.