Report grading **(the pdf has to be provided in the archive file):**

Data structures / storage of data sets [2 pts] in Column U and V for comments

Mention without penalizing if group names are missing. The emphasis should be more on “why” design decisions are made (e.g. of where to store the data) rather than “how” (e.g. only a list of attributes and methods without explanations).

[R1] Penalize **-0.5 pt** if the text is a bulky block without a clear structure of paragraphs.

[R2] Penalize **-1pt if the description of the class hierarchy is missing for “Lifeform” :** we expect to know why they chose such attributes/methods for the superclass and the derived classes algue, corail, scavenger (only a warning if the motivation is missing, no penalty)

[R3] Penalize **-0.5pt** per missing entity description : **simulation, shape**.

- Among the provided description, we expect to know where the sets of entities are stored: **algue, corail, scavenger**.
- We only ask what types have been designed for **shape**

=> **In column U remove the number of point indicated for each feature that is not achieved, but not more than 2 pts**.

In the spreadsheets column V report comment, note down the corresponding **code(s) : e.g. [R1],[R2],[R3]**

Execution grading:

The spreadsheet is organized with the same style of pre-filled “max-point” columns from W to AD. Column AE is the column for comments related to execution. It is to be filled in case of failure with the execution code(s) listed into brackets **[ ].** You may briefly add indication of what is wrong

Column [**Comp**] **=> obtaining an executable with `make`** gives 0.5pts
If it does not work at the first trial, remove the 0.5 point. Then try to fix the bug if it is obvious (less than 1 min search in the Makefile or the code). If you feel it requires more work, contact immediately **Yichen (in english)** so that he asks the group to upload a new version of the archive file (put me in CC).

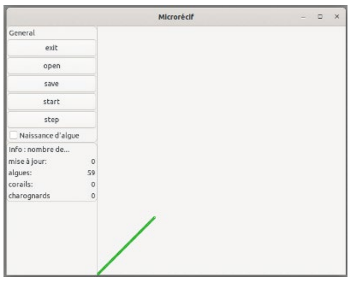
Column **[IE] Isolated Execution commands for correct files (2.00pt)**
Provide t23.txt on the execution command line as follows:

```
./projet t23.txt
```

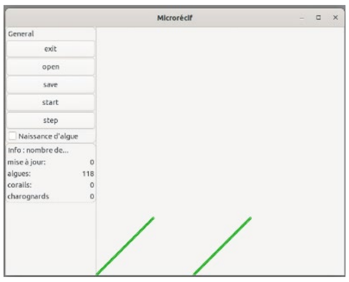
Check that the drawing and ***values*** in the left column are correct (Fig next page). No problem with the style ; we don’t grade if values are aligned or if there is a frame or the background color.
Quit the program with exit after each file ; do the same for **t24.txt, t25.txt** and **t26.txt**

- The requested colors are : light gray for world boundary, green circle for algue, blue for corail, red for scavenger.

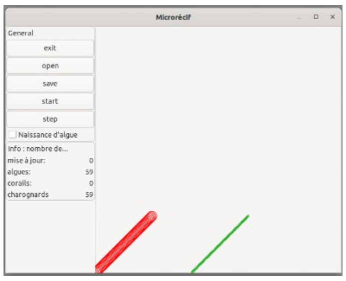
=> 0.5 pt for each successful case = correct values and drawing ;
=> general penalty of -0.5pt if the values do not appear unless we press start or step



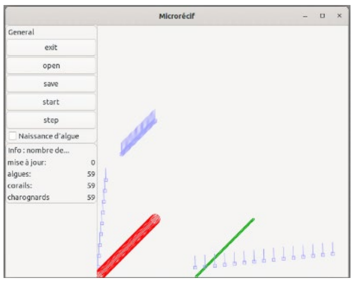
```
Info : nombre de...
mise à jour:      0
algues:          59
corails:          0
charognards       0
```
t23



```
Info : nombre de...
mise à jour:      0
algues:         118
corails:          0
charognards       0
```
t24

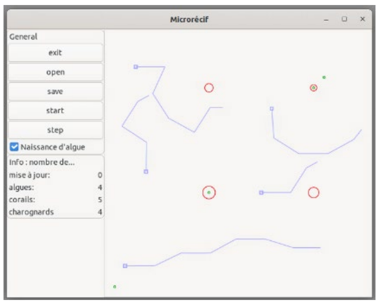


```
Info : nombre de...
mise à jour:      0
algues:          59
corails:          0
charognards      59
```
t25



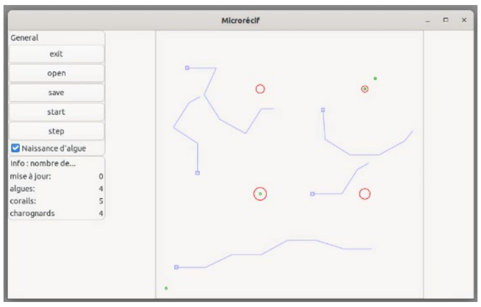
```
Info : nombre de...
mise à jour:      0
algues:          59
corails:         59
charognards      59
```
t26

Column **[DF] Distorsion-free for one correct file (1.00pt)**
Run with one of the correct file that displays correctly and change the window size in the 2 directions to check that we still see the whole simulation space and that there is no distorsion => squares remain squares. See the example next page (bottom).
=> 0.5 pt for each successful direction
=> -0.5 pt if the drawing area remains constant ; it never takes advantage of the increased window size



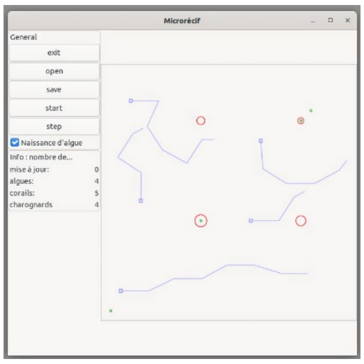
Info : nombre de...
mise à jour: 0
algues: 4
corails: 5
charognards 4

Fenêtre avec la taille initiale sur t27



Info : nombre de...
mise à jour: 0
algues: 4
corails: 5
charognards 4

Agrandissement horizontal



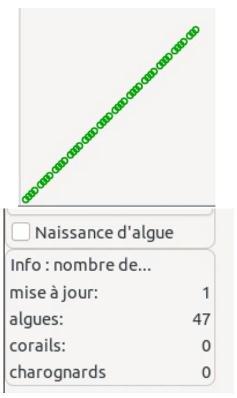
Info : nombre de...
mise à jour: 0
algues: 4
corails: 5
charognards 4

Agrandissement vertical

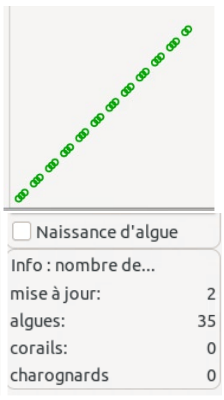
Column **[SB] Step Button and algue end-of-life management(1.5pt)**

Run the program with **`t23.txt.`** Check Step when the simulation is stopped and verify that the counter progresses by a single unit for each click (0.25pt). Verify that you get the correct number of remaining algue after 1 step (47), 2 steps(35), 3steps(23), 4 steps(11) & 5 steps (zero): 0.25pteach for correct value and drawing ; see next page.

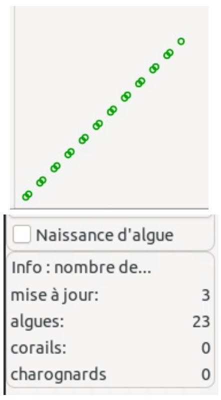
The algue creation is not enabled here ; the same for [SSKb].



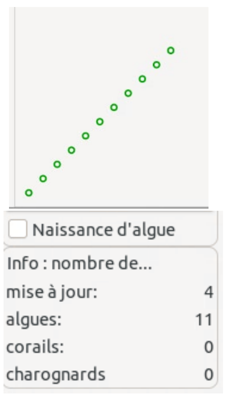
Info : nombre de...
mise à jour: 1
algues: 47
corails: 0
charognards 0



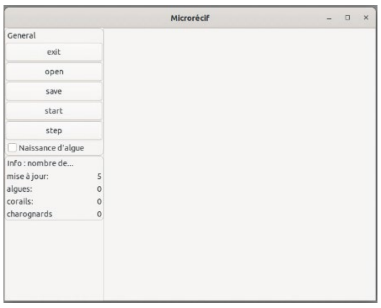
Info : nombre de...
mise à jour: 2
algues: 35
corails: 0
charognards 0



Info : nombre de...
mise à jour: 3
algues: 23
corails: 0
charognards 0



Info : nombre de...
mise à jour: 4
algues: 11
corails: 0
charognards 0



Info : nombre de...
mise à jour: 5
algues: 0
corails: 0
charognards 0

Column **[SSKb] Start/Stop and keyboard (1pt)**

(Continue to)Run the program with **`t23.txt.`** Check that **start** becomes **stop** and vice-versa (0.25pt). Check that **Start** triggers a timer who display the current value of a counter (0.25pt) and that Stop stops it. Check that **'s'** does the same as start/stop (0.25), and '1' the same as Step (0.25).

Column **[OSO] Open1-Save1-Open2-OpenSaved (2pts)**

Case 1 (1pt):

- start the program with file **`t26.txt.`**
- use the Save button to save the simulation with the name **ddd.txt** (0.5pt)
- use the Open button on file **`t24.txt.`** the drawing must be updated (0.25pt)
- use the Open button on file **`ddd.txt.`** the drawing must be updated (0.25pt)

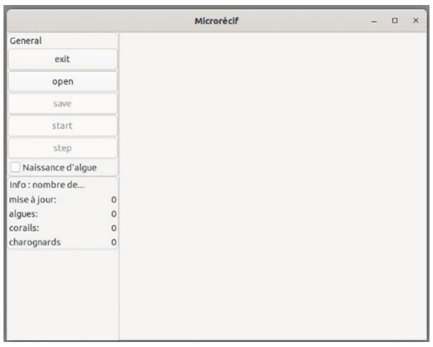
Case 2 (1pt): same as case 1 but use the start-stop-step button before loading **t24.txt**

Column **[ED] Error Detection and resulting behavior (1.5pt)**

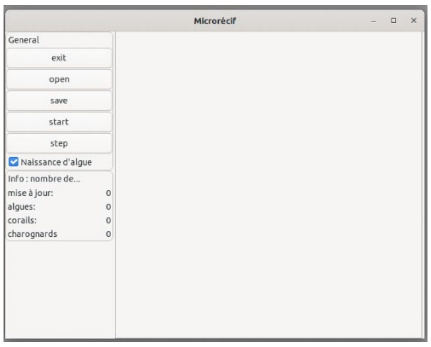
- Run the program with file **`t01.txt`** **=>** The error message should appear in the terminal (pop-up window is ok too) and the program should NOT quit and the drawing area should be empty (0.5pt). default values of 0 should be visible (0.25pt)
- Use the Open button on file **`t23.txt.`** the drawing must be updated (0.25pt)
- Use the Open button on file **`t01.txt.`** the error message should appear in the terminal (0.25pt); the drawing is empty with 0 values (0.25pt).

Column **[Prob] algue creation (2.5.pt)**

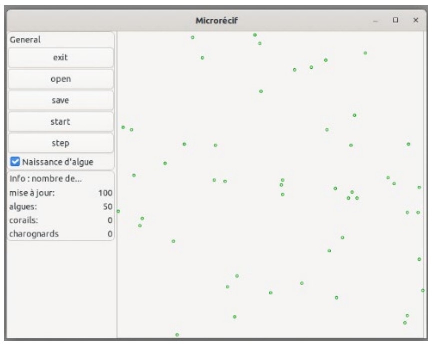
- Run the program with file **`t22.txt.`**Start the simulation ; let it run for **100** iterations by using Start/stop and Step. We get 50 algues created if the random generator has a seed of 1 (0.5pt) ; check that the location of the created entities is random (0.5pt) within the world only (0.5pt) by changing the window size to check the space out of the square.
- Reload t22.txt and start the simulation again ; the creation sequence should be the same (0.5pt)
- Save the simulation with the name **ddd.txt** after some algues are created (0.5pt)
- Open file t22.txt ; then open **ddd.txt** the saved simulation should be visible again (0.5pt)
-
-



Info : nombre de...
mise à jour: 0
algues: 0
corails: 0
charognards 0



Info : nombre de...
mise à jour: 0
algues: 0
corails: 0
charognards 0



Info : nombre de...
mise à jour: 100
algues: 50
corails: 0
charognards 0

-
- t01 t22 t22 après 100 mises à jour
-
-

ARCHITECTURE EVALUATION:

Nb points (max=0.5pts)	**Module role / separation of functionalities**
[A1] 0.5	Must use **argc** and **argv** .

[A2] Architecture features to check for the **Model sub-system**:

Nb points	**Module role / separation of functionalities**
[A2.1] 0.50	Simulation must declare a class ; simulation.h must NOT be included in the lower-level modules ;
[A2.2] 0.50	There must be NO dependency to GTKmm in any Model module
[A2.3] 0.50	The lifeform entities must be managed with a **hierarchy of classes** ; they can be defined in the same module or different modules

[A3] Architecture features to check for module **shape**:

Nb points	**Module role / separation of functionalities (same as rendu1)**
[A3] 0.50	The module **shape** has to be independent from higher level modules, including gui, and from GTKmm ; only the include of **graphic.h** is allowed Ex : including the appendix A = « constantes.h » in the shape module is a clear violation of the architecture specification.

[A4] Architecture features to check for module **gui**:

Nb points	**Module role / separation of functionalities**
[A4] 0.25	connection with the **Model** sub-system with **simulation.h only** but simulation.h can include other interfaces for its own class needs. OK to include shape.h and gtkmm.h

[A5] Architecture features to check for module **graphic** :
Check the report if this module is not present ; in such a case the gui module gather all the relevant information from the Model to manage the display with GTKmm.

ARCHI pt	**If the module graphic is present: Module role / separation of functionalities**
[A5] 0.5pt	Same rule as for [A3]: no dependency to higher level of the Model or to gui

The spreadsheet column AG shows the **default maximum of 2 point** for ARCHITECTURE.
=> **Remove the number of point indicated for each feature that is not achieved, but not more than 2 pts**.
In the spreadsheets column AH architecture violation comment, note down the corresponding **code(s) : e.g. [A1], [A2.1], [A2.3], [A3]** etc

4. CLASS ENCAPSULATION / MODULARIZATION: same as rendu1

[C0] Incomplete implementation: the max number of points is reduced in case of partial implementation. Do not waste time to figure out this in detail ; it should be obvious that a large fraction of the code is missing : ***Report the case to RB who will have a look and calibrate the reduced max.***

[C1] Encapsulation violation : using any **global variable** or making any **attribute public** is strictly forbidden in any modules, including **public** static attributes (no problem for methods and static methods).

It is allowed to have static variables in the implementation (.cc) of a module or variables declared in the unnamed namespace, or **private** static attribute (indicate a warning if there are too many of them). Indicate a BIG warning in case some static variables appear in the interface of a module.

[C2] Externalization of methods' definition : whenever a module interface shows a class interface, it should contain only method prototypes. The method definition must be externalized in the module implementation.

The only *accepted exception* of method definition in the class interface are the **constructors** or **getters** methods that fits onto the same line as the function prototype.

The spread sheet column AI shows the **default maximum of 2 points** for **CLASS**.

=> **Remove 1 point per public attribute or global variable** (max 2pt).

=> **Remove 1 point per interface that is not correctly externalized** (max 2 pt).

The total of removed points from C1 and C2 is maximum 2 pts.

In the spreadsheet column AJ class violation_comment, note down the corresponding **code [C1],[C2]** together with the **interface name** and the **public attribute name**. Indicate that it must be corrected in future assignments.

5. **CODING STYLE: less criteria for Rendu2 to spare time for execution tests**

[L1] Indentation rules have been ignored **more than 4 times** ; read carefully the conventions before considering this penalty because we accept some variants. Please note that we don't indent the **public/private** keywords in class declaration. Indicate only a **warning** if the whole code is consistent in the use of multiple brace styles (e.g. two styles are used but always in the same way, for the same control instructions).

Note: it is OK that "**case**" is not indented in the **switch** block but controlled instructions have to be indented.

[L2] There are **more than 4 wrapping lines** in the code (more than 87 char); Indicate only a warning if 4 wrapping lines or less.

[P2] Apart from two function/method of max 80 lines, all function/method size must not exceed 40 lines (+tolerance of 2 lines) with geany (with the default font size). Recommend to apply the principle of abstraction in case of too long function/method.

The spreadsheet column AK shows the default maximum of 5 points for STYLE

=> **remove 1 point max for [L1]**
=> remove 1 point max for [L2]
=> remove 1 point per function/method that is too long [P2]

In the spreadsheet column AL violation_list, note down the **code** representing the violated criteria followed by the **filename** and the **line number** it occurs. For instance **[L2]simulation.cc57,65,80-84** means that this set of lines are violating the wrapping criteria in the file simulation.cc. If the same type of violation occurs more than 5 times, you mention briefly how much larger the problem is in the violation comment column AM.

Keep the violation_list alphabetically sorted and separate each entry by a comma.

Global comment column AM