| | | | | |
|---|---|---|---|---|
| **329472** | **5** | Warning [L1] shape.cc 74,81 | Super! Le code est limpide, rien à redire | **10** |
| **330605** | **5** | Very clean code with good style | Good start for the project | **8** |
| **339673** | **5** | Very clean code with good style | Good start for the project | **8** |
| **339966** | **4** | [L1]simulation.h, shape.h, lifeform.h : surindentation des classes, simulation.cc15-91, shape.cc32-51, lifeform.cc11-19, 32-36, 54-61 | Le travail est bien réalisé, mais attention à garder votre code propre et lisible, ce qui en facilitera l'amélioration et le debug. | **8** |
| **341435** | **5** | Ok | Bon travail, votre code est très dense et pourrez etre plus lisible.<br>La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **10** |
| **344311** | **5** | OK | The code is well written and very easy to understand. Bravo! Good modularity, especially for decodage_ligne. For the future:<br>- A little tip: if you use unique_ptr instead a normal pointers you don't need to disaloccate all the instances<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Pay attention to some narrowing conversion as they can be the source of bugs<br>- Some member functions can be made static | **10** |
| **344324** | **5** | OK | The code is more or less readable and understandable, good job! Good modularity. But:<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing and indentation<br>- Some functions (such decodage_ligne) are massive, hard to read and convoluted try to break them in smaller functions. Especially since it is clear that you removed all the space to remain under the 80 maximum function size<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Try to name your variables and functions with better names | **10** |
| **344767** | **4** | [L1]shape.h 15-21,24-36,42-43, | Bon code, commentaires pertinent, bonne utilisation des classes. Bravo | **9** |

| | | | | |
|---|---|---|---|---|
| **346202** | **4** | [L1]simulation.cc38-41,shape.cc113,118,119<br>[L2]lifeform.h31,32 Warning : soyez plus consistent dans votre brace style. | Code modulaire, bonnes décisions d'architecture. Attention au style et l'indentation qui ne sont pas très constants. | **9** |
| **346300** | **4** | [L1]:shape.h20-30 | The code is convoluted and very hard to read, try to simplify it and organize it:<br>- Please remove the build files before uploading it on moodle<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing and indentation<br>- Some functions (such decodage_ligne) are massive, hard to read and convoluted try to break them in smaller functions. Especially since it is clear that you removed all the space to remain under the 80 maximum function size<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Try to name your variables and functions with better names<br>- In project.cc , you are checking argv, but your if statement is evaluated without any consequence due to a wrong ; | **6** |
| **347458** | **3** | [P5]: 256 (simulation.cc), 255 (lifeform.cc),<br>[L2]:lifeformcc64,84,96,124... | The code is convoluted and very hard to read, try to simplify it and organize it:<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing and indentation<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Make all the getters as const, so you can avoid possible mistakes<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Some functions (such decodage_ligne) are massive, hard to read and convoluted try to break them in smaller functions<br>- Why you define S2d in shape and you don't use it in the other classes for the coordinates?<br>- There are a lot of narrowing and automatic conversion, pay attention to it, because they can be the source of invisible mistakes<br>- Try to name your variables and functions with better names | **7** |
| **348518** | **5** | Warning indentation shape.cc l.21,25 | Excellent code très bien structuré avec un beau style, continuez ainsi! Attention cependant au variables globales | **9** |

| | | | | |
|---|---|---|---|---|
| **355624** | **4** | [L1]lifeform.h (do not indent public/private), but in generall the if-else with single stament are wrongly indented and the spacing is very incosistent | The code is convoluted and hard to read, try to simplify it and organize it:<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing<br>- Try to name your variables and functions with better names (do not use abbrevations)<br>- Some functions (such lecture) are massive, hard to read and convoluted try to break them in smaller functions. Especially since it is clear that you removed all the space to remain under the 80 maximum function size<br>- Some constructors do not inizialize all the attributes/fields | **9** |
| **355678** | **3** | [L1]projet.cc 20-21; simulation.h 21-23; simulation.cc 51, 73, 80-82,....<br>[P2] simulation.cc decodage:_ligne | Bon usage des typedef, sauf le typedef nommé v. N'utilisez pas de nom de variable/type à une/deux lettres. Si un module doit déjà être inclu dans le .h, pas besoin de l'inclure à nouveau dans le .cc. Attention aux variable globales, c'est une très mauvaise pratique qui devra être corrigé pour le rendu 2. N'hésitez pas à demander à un.e assistant.e de l'aide. Il y a de bonne base mais qui mérite d'être retravaillée, courage | **5** |
| **355690** | **5** | [L2] constantes.h:5 | Bon travail, style très bien respecté de manière générale. Cela dit faites attention à l'architecture et à l'externalisation des méthodes, car cela vous a coûté beaucoup de points. Tâchez de résoudre tout ceci pour le prochain rendu et votre projet frisera la perfection! | **6** |
| **355754** | **4** | [L1]simulation.cc38-41,shape.cc113,118,119<br>[L2]lifeform.h31,32 Warning : soyez plus consistent dans votre brace style. | Code modulaire, bonnes décisions d'architecture. Attention au style et l'indentation qui ne sont pas très constants. | **8** |
| **355769** | **4** | [L1]lifeform.cc 135,143,145,163, | Pour le rendu 2 il faudra convertir lifeform en hierarchie de classe. Attention au nom de paramètre à une lettre. Bon code dans l'ensemble, bon travail | **9** |
| **355770** | **5** | OK, Warning : Bracket not consistent in simulation.cc line 163 | Excellent travail, tout est bien réalisé, BRAVO | **10** |
| **355786** | **4** | [L1] Mauvaises indentation dans les déclarations de classes : shape.h20,27; lifeform.h 11,14, 21, 30, 40,46; simulation.h; Mauvais alignements : lifeform.h24; lifeform.cc22, simulation.cc111-112; Mauvais placement des brackets dans shape.cc | Un bon code, mais le style est inconsistent dans le programme, ce qui en rend la lecture difficile | **9** |

| | | | | |
|---|---|---|---|---|
| **355993** | **4** | [L1]:shape.h20-30 | The code is convoluted and very hard to read, try to simplify it and organize it:<br>- Please remove the build files before uploading it on moodle<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing and indentation<br>- Some functions (such decodage_ligne) are massive, hard to read and convoluted try to break them in smaller functions. Especially since it is clear that you removed all the space to remain under the 80 maximum function size<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Try to name your variables and functions with better names<br>- In project.cc , you are checking argv, but your if statement is evaluated without any consequence due to a wrong ; | **6** |
| **356066** | **4** | [L1] lifeform.cc: 25-29,35-44,49-54,70-71,79 80,141-143,153-156; projet.cc: 9-10; simulation.cc: 131-144; simulation.h: 21,30 | Bien joué! code lisible et clair, sauf à quelques endroits notamment: check_collision qui contient beaucoup de blocs de code imbriqués, pensez à la décomposer. Faites attention au double indentation pour le prochain rendu, et à l'externalisation des méthodes. | **5** |
| **356068** | **5** | OK | The code is more or less readable and understandable, good job! Good modularity. But:<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing and indentation<br>- Some functions (such decodage_ligne) are massive, hard to read and convoluted try to break them in smaller functions. Especially since it is clear that you removed all the space to remain under the 80 maximum function size<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Try to name your variables and functions with better names | **10** |
| **356074** | **5** | OK | Excellent! Votre code est clair, cohérent et homogène. Vous avez de bons reflexes de programmation et vous faites bien attention aux conventions. Continuez ainsi! | **10** |
| **356179** | **5** | warning [L2] : wrapping lines simulation.cc l.116 l.51; shape.cc l.39, lifeform.h l.29 | Très bon travail, très bonne architecture et modularisation. Vous devez vraiment faire attention aux conventions | **6** |
| **356332** | **5** | ok | code clair avec un bon usage des enums, et bonne décomposition en fonctions | **9** |

| | | | | |
|---|---|---|---|---|
| **356447** | **3** | [L1]projet.cc 20-21; simulation.h 21-23; simulation.cc 51, 73, 80-82,....<br>[P2] simulation.cc decodage:_ligne | Bon usage des typedef, sauf le typedef nommé v. N'utilisez pas de nom de variable/type à une/deux lettres. Si un module doit déjà être inclu dans le .h, pas besoin de l'inclure à nouveau dans le .cc. Attention aux variable globales, c'est une très mauvaise pratique qui devra être corrigé pour le rendu 2. N'hésitez pas à demander à un.e assistant.e de l'aide. Il y a de bonne base mais qui mérite d'être retravaillée, courage | **5** |
| **356459** | **3** | [L1] simulation.cc:19-22,41,68,72,... lifeform.cc:17,... etc<br>[L2] simulation.cc:17 simulation.h:12,15 lifeform.cc:27,76,... | Bon travail mais très peu de rigueur pour le style, en particulier pour l'indentation et la longueur max des lignes. Il faut aussi faire plus attention à l'architecture: projet.cc doit contenir le strict minimum et shape doit être totalement indépendant du modèle. Essayez d'améliorer ces points d'ici le prochain rendu ! | **6** |
| **356483** | **5** | OK | The code is well written and easy to understand good job! Good that you divided the function to read the files in many smaller functions! It is a little bit strange to see PascalCase functions names with sometimes funny names (GodFunctions, AllJudgment). The only problem was that struct in lifeform as they have all their attributes publics. In any case for next time fix the classes and:<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Pay attention to old c casting and automatic casting | **7** |
| **356497** | **5** | Ok | Un très bon travail, le rendu est propre, bravo ! | **10** |
| **356509** | **4** | [L1] toutes vos classes ont une indentation de trop. Les mots de clé private/public ne doit pas avoir d'indentation. | Code excellent et très bonne modularisation. Faites attention à l'indentation dans les classes. | **9** |
| **356568** | **5** | [L2] lifeform.h: l.32,l.75,lifeform.cc l.171,l.38 | Très bon travail, code très lisible, bien modularisé et très homogène. Pour la prochaine fois revoyez le principe d'abstraction: une méthode rempli une fonction. | **9** |
| **356665** | **4** | [L1] simulation.cc : 53, lifeform.cc : 154,218,220,226,...,239,... Attention aux accolades de tes if (voir conventions du cours). Mets tous les typedef de lifeform.h en haut du fichier, après les enum. | Très bon code avec un bon style. Attention à votre fonction "decodage_ligne" : celle-ci devrait plutôt se trouver dans le module simulation et devrait faire appel à des méthodes plus spécifiques de lifeform. Si vous avez un peu de marge, essayez d'aérer un peu plus cette fonction. | **9** |
| **356696** | **5** | Warning [L1] simulation.cc : 68, lifeform.cc : 16 ; Warning [L2] lifeform.h : 46,65. | Excellent code ! Continuez ainsi. | **10** |

| | | | | |
|---|---|---|---|---|
| **356848** | **4** | [L1] projet.cc 22.30, shape.cc 20,47, lifeform.h 21. Remarque: essayez d'éviter trop de niveaux d'indentation dans vos méthodes isValid | Certains #include ne sont pas nécessaires dans vos .h, essayez de les minimiser. Bien structuré et présentation claire et lisible, bravo ! | **8** |
| **356960** | **4** | [L1]lifeform.h12,21,29,... | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. Attention les mots clés public/private ne doivent pas être indentés. | **9** |
| **358053** | **2** | [L1] lifeform.cc 80,84-105, 117, 120, .... [L2] lifeform.cc 67-68, 94, 103, 142, ... [P2] simulation.cc testCorail, project.cc lire_fichier | Code illisible. Les conventions ne sont pas du tout respectées, le code n'est pas indenté. On retrouve régulièrement des lignes de 100-130 caractères... | **6** |
| **358079** | **4** | [L1]lifeform.h (do not indent public/private), but in generall the if-else with single stament are wrongly indented and the spacing is very incosistent | The code is convoluted and hard to read, try to simplify it and organize it:<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing<br>- Try to name your variables and functions with better names (do not use abbrevations)<br>- Some functions (such lecture) are massive, hard to read and convoluted try to break them in smaller functions. Especially since it is clear that you removed all the space to remain under the 80 maximum function size<br>- Some constructors do not inizialize all the attributes/fields | **9** |
| **358229** | **4** | [L1] : simulation.h private/public ne doivent pas etre indenté simulation.cc mauvaise indentation L.209-225 | Très bon travail, continuez ainsi. Pensez à mettre vos noms en haut des fichiers | **8** |
| **358422** | **4** | [L1] shape.cc 33, lifeform.h 19,30, simulation.cc 219-222,235-238, lifeform.cc 96-98,143-152,155,159-164,167 | Attention pour la suite, argv[1] est accédé sans vérifier argc. Faites attention à votre indentation pour la suite, ça aidera aussi à rendre votre code plus facilement lisible. Autrement, la structure est bien respectée et le code bien modularisé. | **8** |

| | | | | |
|---|---|---|---|---|
| **358556** | **5** | OK, but the spacing is very inconsistent thereby making your code hard to read | The code is convoluted and hard to read, try to simplify it and organize it:<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing<br>- Try to name your variables and functions with better names (do not use abbrevations)<br>- Class names should always be capitalized, i.e PascalCase<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Instead of doing else return xxx; you can just do return xxx; | **9** |
| **359309** | **3** | [L1]simulation.h 20,51; simulation.cc 38, lifeform.h 18,37,lifeform.h 118-120, all of lifeform.cc<br>[L2]simulation.cc 53, 73,82, 122,128,156,... | Attention à bien utiliser des setter/getter pour modifier/accéder aux attributs (nbSim_). Vous avez créé un un enum l'état de la lecture, utilisez-le. Les mots-clés public,private et protected ne doivent pas être indentés. De bonnes idées, bon travail | **7** |
| **359317** | **5** | OK | The code is well written and very easy to understand. Bravo! Good modularity, especially for decodage_ligne. For the future:<br>- A little tip: if you use unique_ptr instead a normal pointers you don't need to disaloccate all the instances<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Pay attention to some narrowing conversion as they can be the source of bugs<br>- Some member functions can be made static | **10** |
| **359350** | **5** | [L1]: simulation.cc: fonction message_error, instanceDel; lifeform.h: constructeur lifeform; lifeform.cc: message_error | C'est un très bon travail qui témoigne de très bonnes connaissances en programmation. Par ailleurs, faites attention à vos indentions car vous utiliser parfois deux type d'accolades, parfois même dans les même fonctions. | **8** |
| **360542** | **5** | warning : plusieurs styles d'accolades regardez le constructeur de Simulation dans simulation.cc, warning : manque d'indentations les arguments ne sont pas alignés simulation.cc:232-233 | Code excellent et très bonne modularisation. | **9** |

| | | | | |
|---|---|---|---|---|
| **360573** | **5** | warning [L2] : wrapping lines simulation.cc l.116 l.51; shape.cc l.39, lifeform.h l.28 | Très bon travail, très bonne architecture et modularisation. Vous devez vraiment faire attention aux conventions | **6** |
| **360696** | **4** | [L1] : simulation.h private/public ne doivent pas etre indenté simulation.cc mauvaise indentation L.209-225 | Très bon travail, continuez ainsi. Pensez à mettre vos noms en haut des fichiers | **8** |
| **360804** | **5** | Warning : soyez consistant dans votre indentation des if | Excellent ! Code très lisible. Bonne architecture et modularisation. Continuez ainsi. | **10** |
| **360955** | **4** | [L1] : simulation.h l.15,18 shape.h l.16,26 lifeform.h l.24,47,54 public/private doivent etre indenté warning : le style de vos accolades doivent etre consistant (différents if dans shape | Bon travail, continuez ainsi | **8** |
| **360961** | **5** | warning : plusieurs styles d'accolades regardez le constructeur de Simulation dans simulation.cc, warning : manque d'indentations les arguments ne sont pas alignés simulation.cc:232-233 | Code excellent et très bonne modularisation. | **9** |
| **360966** | **5** | Warning : soyez consistant dans votre indentation des if | Excellent ! Bon respect des convention, code très lisible. Bonne architecture et modularisation. Continuez ainsi. | **10** |
| **360985** | **4** | [L1] : simulation.h l.15,18 shape.h l.16,26 lifeform.h l.24,47,54 public/private doivent etre indenté warning : le style de vos accolades doivent etre consistant (différents if dans shape | Bon travail, continuez ainsi | **8** |

| | | | | |
|---|---|---|---|---|
| **361003** | **3** | [P2] decodage_ligne (144 lines), [L1] lifeform.h34-39, 62-67 | The code is convoluted and very hard to read. It looks like that it was rushed through... try to simplify it and organize it:<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing and indentation<br>- Some functions (such decodage_ligne) are massive, hard to read and convoluted try to break them in smaller functions, especially since it is above the maximal threshold<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Try to name your variables and functions with better names<br>- You can only define getters and constructors in the interface and they must be in only one line | **5** |
| **361119** | **3** | [P5]: 256 (simulation.cc), 255 (lifeform.cc), [L2]:lifeformcc64,84,96,124... | The code is convoluted and very hard to read, try to simplify it and organize it:<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing and indentation<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Make all the getters as const, so you can avoid possible mistakes<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Some functions (such decodage_ligne) are massive, hard to read and convoluted try to break them in smaller functions<br>- Why you define S2d in shape and you don't use it in the other classes for the coordinates?<br>- There are a lot of narrowing and automatic conversion, pay attention to it, because they can be the source of invisible mistakes<br>- Try to name your variables and functions with better names | **7** |
| **361158** | **5** | Warning [L1] simulation.cc : 68, lifeform.cc : 16 ; Warning [L2] lifeform.h : 46,65. | Excellent code ! Continuez ainsi. | **10** |

| | | | | |
|---|---|---|---|---|
| **361167** | **5** | Excellent ! | Un travail remarquable, une lisiblité excellente, BRAVO<br>Cependant ATTENTION : l'encapsulation de vos classes de lifeform est laisse à désirer étant donné que la vérification de la validité des paramètres est externe à la classe. En effet, la classe n'a pas de contrôle sur la validité de ses paramètres | **10** |
| **361170** | **5** | OK, Warning : Bracket not consistent in simulation.cc line 163 | Excellent travail, tout est bien réalisé, BRAVO | **10** |
| **361248** | **4** | [L1] lifeform.cc: 7-9, 210; lifeform.h: 19, 62; simulation.cc: 26-50, 84, 103, 128, ...<br>Alignez le code par rapport aux paranthèse de la fonction | Bien joué! Juste quelques points: les noms calculs_angle_i ne sont pas très parlants, essayez d'éviter de faire des fonctions qui continent trop de blocs if-elseif-else, ca a tendance à rendre le code assez opaque et pensez à enregistrer les données des entités dans un vecteur qui n'est pas local à la fonction lecture. A part ces quelques points bien joué! | **9** |
| **361295** | **4** | [L1] lifeform.h: 25-29, 33-36, 40-52, 57-61, ...<br>Ne pas indenter public et private<br>[L1] simulation.cc: 77-82,104-195,121-126; simulation.h:4-86<br>[L2] warning: shape.cc 68 | Bon travail! Le code est clair et lisible. Dommage pour les quelques fautes d'indentations. Bien joué! | **9** |
| **361373** | **4** | [L1] : le fichier lifeform.h est mal configuré -> indentation de 3 espaces | Très bon travail, bon style de code, bonne architecture et bonne modularisation, continuez ainsi | **9** |
| **361525** | **4** | [L1] : shape.cc l.21,22 lifeform.cc l.35,71,125,136<br>warning [L2] : simulation.cc l.128 | Bon travail, faites attention à votre utilisation des attribut static. | **8** |
| **361740** | **5** | OK | The code is well written and easy to understand good job! Good that you divided the function to read the files in many smaller functions! It is a little bit strange to see PascalCase functions names with sometimes funny names (GodFunctions, AllJudgment). The only problem was that struct in lifeform as they have all their attributes publics. In any case for next time fix the classes and:<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Pay attention to old c casting and automatic casting | **7** |
| **361977** | **3** | [L1] simulation.cc:19-22,41,68,72,... lifeform.cc:17,... etc<br>[L2] simulation.cc:17 simulation.h:12,15 lifeform.cc:27,76,... | Bon travail mais très peu de rigueur pour le style, en particulier pour l'indentation et la longueur max des lignes. Il faut aussi faire plus attention à l'architecture: projet.cc doit contenir le strict minimum et shape doit être totalement indépendant du modèle. Essayez d'améliorer ces points d'ici le prochain rendu ! | **6** |

| | | | | |
|---|---|---|---|---|
| **362065** | **4** | [L1]lifeform.h12,21,29,... | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. Attention les mots clés public/private ne doivent pas être indentés. | **9** |
| **362182** | **5** | OK | Attention pour la suite, argv[1] est accédé sans vérifier argc. Préférez <cmath> à <math.h>, et essayez d'enlever les #include inutiles dans les .h. Pour la suite, faites attention à l'externalisation des définitions des constructeurs. Autrement, le code est très clair et lisible, rien à redire sur le style! | **8** |
| **362197** | **5** | [L1] simulation.cc:165 [WARNING | Très beau travail, rendu très propre, bravo et continuez comme ça ! | **10** |
| **362281** | **5** | Excellent ! | Un travail remarquable, une lisiblité excellente, BRAVO<br>Cependant ATTENTION : l'encapsulation de vos classes de lifeform est laisse à désirer étant donné que la vérification de la validité des paramètres est externe à la classe. En effet, la classe n'a pas de contrôle sur la validité de ses paramètres | **10** |
| **362291** | **5** | Warning [L1] shape.cc 74,81 | Super! Le code est limpide, rien à redire | **10** |
| **362360** | **4** | [L1] lifeform.h, il faut indenter les attributs et les methodes dans tout le code comme vous l'avez fait dans simulation.h ([L17]Conventions de programmation) | Bonne base de projet | **9** |
| **362397** | **5** | OK, but the spacing is very inconsistent thereby making your code hard to read | The code is convoluted and hard to read, try to simplify it and organize it:<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing<br>- Try to name your variables and functions with better names (do not use abbrevations)<br>- Class names should always be capitalized, i.e PascalCase<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Instead of doing else return xxx; you can just do return xxx; | **9** |
| **362436** | **4** | [L1] simulation.h 23,25, simulation.cc 48,162, lifeform.h 17,28-31, 60, etc. | Certains #include ne sont pas nécessaires dans vos .h, essayez de les minimiser. Plusieurs instructions très longues: essayez de les diviser pour les rendre plus claires et faites attention à l'alignement sur plusieurs lignes. Autrement, le code est assez clair et l'architecture bien respectée. | **7** |

| | | | | |
|---|---|---|---|---|
| **362702** | **5** | OK | The code is very readeable and understandable, good job! Good modularity. For the future:<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Pay attention to some narrowing conversion as they can be the source of bugs<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- You define all your vectors as static global vector, but it would be better to be an attribute of your class Simulation so you can easily access them and pass them around<br>- In project.cc you call the method lecture through an instance of simulation, but since it is a static method, there is no need istantiate one (or change it to a non static method)<br>- You could use pointer/unique_ptr to store your instances in the vectors for performance (since you avoid copying them) | **10** |
| **362952** | **5** | [L2] constantes.h:5 | Bon travail, style très bien respecté de manière générale. Cela dit faites attention à l'architecture et à l'externalisation des méthodes, car cela vous a coûté beaucoup de points. Tâchez de résoudre tout ceci pour le prochain rendu et votre projet frisera la perfection! | **6** |
| **362974** | **5** | | Super! Le code est très propre. Rien à redire. | **10** |
| **362981** | **4** | Warning [L1] trop de styles d'indentation differentes dans simulation.cc. Warning [L2] lifeform l.26,45. [P5] dans simulation.cc trop de variables a 1 caractère, on ne comprend pas a quoi elle servent. | Bonne base de projet | **8** |
| **363062** | **4** | [L1] simulation.cc : 53, lifeform.cc : 154,218,220,226,...,239,... Attention aux accolades de tes if (voir conventions du cours). Mets tous les typedef de lifeform.h en haut du fichier, après les enum. | Très bon code avec un bon style. Attention à votre fonction "decodage_ligne" : celle-ci devrait plutôt se trouver dans le module simulation et devrait faire appel à des méthodes plus spécifiques de lifeform. Si vous avez un peu de marge, essayez d'aérer un peu plus cette fonction. | **9** |
| **363081** | **5** | [L2] lifeform.h: l.32,l.75,lifeform.cc l.171,l.38 | Très bon travail, code très lisible, bien modularisé et très homogène. Pour la prochaine fois revoyez le principe d'abstraction: une méthode rempli une fonction. | **9** |
| **363238** | **5** | Warning [L1] simulation.cc 274,276, lifeform.cc 58,178 | L'inclusion de fichiers du Modèle dans le module shape est à corriger. Style: les noms de classe devraient commancer par une majuscule, et faites attention à l'alignement de paramètres sur plusieurs lignes. Sinon, le code est propre et bien lisible. | **8** |

| | | | | |
|---|---|---|---|---|
| **363255** | **3** | [L1]simulation.h 20,51; simulation.cc 38, lifeform.h 18,37,lifeform.h 118-120, all of lifeform.cc [L2]simulation.cc 53, 73,82, 122,128,156,... | Attention à bien utiliser des setter/getter pour modifier/accéder aux attributs (nbSim_). Vous avez créé un un enum l'état de la lecture, utilisez-le. Les mots-clés public,private et protected ne doivent pas être indentés. De bonnes idées, bon travail | **7** |
| **363263** | **5** | Warning: inconsistence du brace style | Code clair, bonnes décisions d'architecture et bon style. Attention le module shape doit être indépendant. Vous pouvez utliser plus de commentaires par exemple pour délimiter des espaces du code pour l'implémentation des différentes entités. | **9** |
| **363267** | **5** | Ok | Bon travail, votre code est très dense et pourrez etre plus lisible. La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **10** |
| **363275** | **4** | [L1] Mauvaises indentation dans les déclarations de classes : shape.h20,27; lifeform.h 11,14, 21, 30, 40,46; simulation.h; Mauvais alignements : lifeform.h24; lifeform.cc22, simulation.cc111-112; Mauvais placement des brackets dans shape.cc | Un bon code, mais le style est inconsistent dans le programme, ce qui en rend la lecture difficile | **9** |
| **363304** | **5** | OK | Code très lisible et très constant, les noms de fonctions sont clairs. Mais attention pour la prochaine fois à la modularisation. | **9** |
| **363429** | **4** | [L1] lifeform.cc : 31,34,37,132, shape.cc : 90. Warning [L2] lifeform.cc : 128, shape.cc : 83,90. Attention à tes if à une seule instruction : si tu décides de mettre l'instruction entre accolade respecte les conventions (tout sur la même ligne ou saut de ligne pour l'instruction et le "}") | Très bon code, continuez ainsi ! | **8** |
| **363518** | **4** | Warning [L1] trop de styles d'indentation differentes dans simulation.cc. Warning [L2] lifeform l.26,45. [P5] dans simulation.cc trop de variables a 1 caractère, on ne comprend pas a quoi elle servent. | Bonne base de projet | **8** |
| **363566** | **5** | [L1] simulation.cc:165 [WARNING | Très beau travail, rendu très propre, bravo et continuez comme ça ! | **10** |

| | | | | |
|---|---|---|---|---|
| **364037** | **5** | [L1]simulation.h13 | Code clair et modulaire, très bonnes décisions d'architecture et très bon style. Vous pouvez utliser plus de commentaires par exemple pour délimiter des espaces du code pour l'implémentation des différentes entités. Attention le module shape doit être indépendant. | **9** |
| **364053** | **5** | [L1]: simulation.cc: fonction message_error, instanceDel; lifeform.h: constructeur lifeform; lifeform.cc: message_error | C'est un très bon travail qui témoigne de très bonnes connaissances en programmation. Par ailleurs, faites attention à vos indentions car vous utiliser parfois deux type d'accolades, parfois même dans les même fonctions. | **8** |
| **364063** | **5** | Warning: inconsistence du brace style | Code clair, bonnes décisions d'architecture et bon style. Attention le module shape doit être indépendant. Vous pouvez utliser plus de commentaires par exemple pour délimiter des espaces du code pour l'implémentation des différentes entités. | **9** |
| **364095** | **4** | [L1] lifeform.cc: 25-29,35-44,49-54,70-71,79 80,141-143,153-156; projet.cc: 9-10; simulation.cc: 131-144; simulation.h: 21,30 | Bien joué! code lisible et clair, sauf à quelques endroits notamment: check_collision qui contient beaucoup de blocs de code imbriqués, pensez à la décomposer. Faites attention au double indentation pour le prochain rendu, et à l'externalisation des méthodes. | **5** |
| **364173** | **4** | [L1] lifeform.cc: 7-9, 210; lifeform.h: 19, 62; simulation.cc: 26-50, 84, 103, 128, ... Alignez le code par rapport aux paranthèse de la fonction | Bien joué! Juste quelques points: les noms calculs_angle_i ne sont pas très parlants, essayez d'éviter de faire des fonctions qui continent trop de blocs if-elseif-else, ca a tendance à rendre le code assez opaque et pensez à enregistrer les données des entités dans un vecteur qui n'est pas local à la fonction lecture. A part ces quelques points bien joué! | **9** |
| **371921** | **4** | [L1] simulation.h 23,25, simulation.cc 48,162, lifeform.h 17,28-31, 60, etc. | Certains #include ne sont pas nécessaires dans vos .h, essayez de les minimiser. Plusieurs instructions très longues: essayez de les diviser pour les rendre plus claires et faites attention à l'alignement sur plusieurs lignes. Autrement, le code est assez clair et l'architecture bien respectée. | **7** |
| **371946** | **4** | [L1] simulation.cc 26-27, lifeform.h (général) double indentation, lifeform.h 121, lifeform.cc 125-126,286-287. simulation.cc plusieurs styles d'accolades incohérents | Inclusion de constantes.h dans shape à corriger, et faites attention à l'externalisation des définitions de constructeurs. Sinon, le code est bien lisible, aéré et bien structuré. | **7** |
| **371969** | **4** | Warning [L1] essayez d'unifier vos styles d'accolades pour les instructions de controôle. [L1] shape.h 16-32 double indentation, shape.cc 96,109,123 style d'accolades différent du reste, shape.cc 114, lifeform.cc 44,65-66 | Attention pour la suite, argv[1] est accédé sans vérifier argc. Globalement excellent travail, particulièrement clair et bien structuré, bravo! | **9** |

| | | | | |
|---|---|---|---|---|
| **371989** | **5** | Ok | Travail de très grande qualité. Très beau style de code. Excellente modularisation et architecture.<br>Continuez ainsi | **9** |
| **372017** | **5** | OK | Code excellent et bonne modularisation. | **10** |
| **372018** | **4** | [L1] simulation.h 19-20,30,32,34,36,45, shape.h 19, shape.cc 89, etc. | Attention pour la suite, argv[1] est accédé sans vérifier argc. Faites attention à l'alignement des instructions sur plusieurs lignes. Sinon, le code est très clair et bien structuré, bravo ! | **9** |
| **372048** | **5** | warning : une indentation de trop dans le fichier shape.cc ligne 8 à 10 | Code excellent et bien modulariser, pensez à revoir la classe Simulation cela pourrait vous aider pour le rendu 2 | **9** |
| **372051** | **4** | [L1]lifeform.h:19,51,63,77,84,107,simulation.h:13,26 | Code globalement clair et très structuré, pensez à ne pas indenter public et private dans les classes, je n'ai pas compris à quoi servait #define _USE_MATH_DEFINES, est-ce util? | **8** |
| **372071** | **3** | [L1] projet.cc : 10, simulation.cc : 50,65,67,72,76,78,...,199-203,222-227,..., lifeform.cc : 91,104,113,131,..., shape.cc : 19,54,58,107,108. [P2] simulation.cc : decodage. WARNING : Ne changez pas de style d'accolade au cours du code ! Beaucoup de lignes trop ou pas assez indentée. | Bon code. Attention à bien garder le même style de code tout le programme, à mettre vos accolades au bon endroit, etc (voir conventions de programmation du cours). Pour vos fonctions/méthodes/constructeurs avec beaucoup de paramètres,je vous recommande de suivre la règle [L22] et de mettre le plus de paramètres sur une seule ligne pour plus de lisibilité. | **8** |
| **372082** | **5** | OK | Code excellent et bonne modularisation. Le fichier entities.cc n'est pas utile (il y a seulement include entities.h). | **9** |
| **372083** | **5** | OK | Code très lisible et très constant, les noms de fonctions sont clairs. Mais attention pour la prochaine fois à la modularisation. | **9** |
| **372118** | **5** | [L1]simulation.h13 | Code clair et modulaire, très bonnes décisions d'architecture et très bon style. Vous pouvez utliser plus de commentaires par exemple pour délimiter des espaces du code pour l'implémentation des différentes entités. Attention le module shape doit être indépendant. | **9** |
| **372124** | **3** | [L1]projet.cc 11-16, simulation.cc 28,62,119, lifeform.h 33,50,... [L2]lifeform.h 32,33,49,50; lifeform.cc 50,60; simulation.cc 39 | Attention, pour le rendu 2 une classe simulation sera nécessaire. Bon usage des enum. Attention aux constante. (epsil de shape.cc dans segmentSuperposition meriterait d'être une constexpr au début de shape.cc) Attention aux quelques erreurs d'inattention, sinon bonne base pour la suite. | **8** |
| **372178** | **5** | OK | Bel effort sur les commentaire, peut-être un peu trop nombreux. (bravo pour la mention du forum où un morceau code à été repris, très bonne pratique). Belle décomposition. Toute petite erreur d'indentation dans simulation.cc ligne 190, sinon très bon travail. | **10** |

| | | | | |
|---|---|---|---|---|
| **372192** | **4** | Warning [L1] essayez d'unifier vos styles d'accolades pour les instructions de controôle. [L1] shape.h 16-32 double indentation, shape.cc 96,109,123 style d'accolades différent du reste, shape.cc 114, lifeform.cc 44,65-66 | Attention pour la suite, argv[1] est accédé sans vérifier argc. Globalement excellent travail, particulièrement clair et bien structuré, bravo! | **9** |
| **372212** | **4** | [L1] Warning lifeform.cc l.36-54 3 indentations au lieu de 1 [L2] Simulation.h l.22-24 et Simulation.cc l.73,128,134265 | Excellent code, tres propre et tres bien structuré. Ca semble refeleter un très bon niveau de programmation. Attention cependant au nb max de character par ligne. | **9** |
| **372226** | **5** | Attention : shape.cc ligne 15 : double indentation; lifeform.cc ligne 73 | C'est un bon travail, attention cependant à vos tests et à votre style de code qui n'est pas toujours constant | **10** |
| **372331** | **5** | [L1]lifeform.h, double indentation dans la définition des classes Warning : traiterCorail dans simulation.cc fait 42 lignes, vous avez le droit à qu'une seule fonction de plus de 40 lignes (LectureFichier) | Code modulaire, très bonnes décisions d'architecture et style OK. | **10** |
| **372387** | **4** | [L1] lifeform.cc : 118-128, simulation.cc : 109-111, 160-192 ; Warning [L2] dans lifeform.cc (l.113,114,131) | Très bon code, très lisible et compréhensible. Attention à ne pas changer de style d'indentation au cours du code. Pensez à faire un fichier "constantes.h" pour les constantes du projet (pour ne pas les avoir dans lifeform.h) | **7** |
| **372414** | **5** | [L1]lifeform.h, double indentation dans la définition des classes Warning : traiterCorail dans simulation.cc fait 42 lignes, vous avez le droit à qu'une seule fonction de plus de 40 lignes (LectureFichier) | Code modulaire, très bonnes décisions d'architecture et style OK. | **10** |
| **372417** | **4** | [L1] : simulation.h l.14,15 simulation.cc l.58,68,112,143,165 lifeforms.cc l.12-18 | Bon travail !<br>Bon style de code dans l'ensemble, essayez tout de meme de découper d'avantage votre code. La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **9** |
| **372427** | **5** | Ok | Bravo! Du beau travail! Seule petite critique essayez si possible d'éviter des fonctions du style verif_intersection dans lifeform qui sont trop imbriqués, ca peut rendre le code assez dur à lire | **10** |

| | | | | |
|---|---|---|---|---|
| **372454** | **4** | [L1] : simulation.h l.11,15; simulation.cc l.127,140; shape.cc l.69<br>warning [L2] : simulation.cc l.74 | Très bon travail, code très lisible et bonne modularisation.<br>Rq : t'as pas besoin de mettre un ";" à la fin de tes méthodes | **8** |
| **372460** | **4** | [L1] simulation.cc : 191-194, lifeform.cc : 78-82 | Très bon style de code, pour le prochain rendu pensez à bien externaliser vos constructeurs et méthodes | **6** |
| **372468** | **5** | Very clean code and style | Very solid base projects, well done, keep up the good work | **10** |
| **372474** | **5** | [L1] shape.cc:11,22,37,47 [WARNING]<br>[L2] constantes.h:4 [WARNING] | Bon code et bon style (hors quelques lignes où le style d'indentation ne colle pas avec le reste du code), mais faites attention à l'externalisation des méthodes: il ne faut pas définir de méthodes dans les fichiers .h! | **7** |
| **372570** | **5** | warning : ligne trop longue shape.h:37 | Code excellent et bonne modularisation. Le fichier lifeform.cc est assez long, peut-être le découper en plusieurs modules?? | **10** |
| **372602** | **4** | [P5] simulation.cc (a lot of magic numbers) | The code is more or less readable and understandable, good job! Good modularity. For the future:<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Pay attention to some narrowing conversion as they can be the source of bugs<br>- Try to name your variables and functions with better names<br>- You can only define getters and constructors in the interface and they must be in only one line<br>- Pay attention to magic numbers, defined them as constexpr | **8** |
| **372625** | **4** | [L1] lifeform.cc:139-143 | Beau travail et beau style, faites néanmoins attention à l'architecture car shape doit être indépendant du modèle. | **8** |
| **372633** | **4** | [P5] simulation.cc (a lot of magic numbers) | The code is more or less readable and understandable, good job! Good modularity. For the future:<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Pay attention to some narrowing conversion as they can be the source of bugs<br>- Try to name your variables and functions with better names<br>- You can only define getters and constructors in the interface and they must be in only one line<br>- Pay attention to magic numbers, defined them as constexpr | **8** |

| | | | | |
|---|---|---|---|---|
| **372689** | **5** | Ok | Bravo! Du beau travail! Seule petite critique essayez si possible d'éviter des fonctions du style verif_intersection dans lifeform qui sont trop imbriqués, ca peut rendre le code assez dur à lire | **10** |
| **372757** | **4** | [L1] Simulation.h 11, lifeforms.cc 88, 169-172, 208-209,212, etc. Général: le style d'indentation d'accolades est à corriger. Warning [L2] lifeforms.cc 140. Warning [P5] Simulation::compteur, Simulation::total et Simulation::i très peu explicites pour des attributs static de classe. | Le style d'accolades est à corriger (utiliser une des variantes proposées dans les conventions). Cela rend votre code assez difficile à lire pour une personne externe, surtout lorsqu'il y a plusieurs niveaux d'indentation. Vos attributs static de Simulation peuvent probablement être transférés dans le .cc ou même dans une fonction; cela les rendra plus compréhensibles. Sinon, la structure générale est bien respectée et le code est bien modularisé. | **6** |
| **372791** | **4** | [L2]ignorance totale de la taille max des lignes à 87 caractères | bonne structuration ; code très lisible mais la contrainte de taille max de ligne est totalement ignorée. | **8** |
| **372830** | **4** | [L1] simulation.h:13-20 shape.cc:61,73,... | Bon travail, bon respect général du style et de l'architecture. Attention, l'indentation est parfois hasardeuse (tantôt 2 espaces, tantôt 4, ...) et il ne faut pas indenter les mots-clés public et private. Attention aussi à ne pas laisser d'attributs publics quels qu'ils soient | **8** |
| **372924** | **4** | [L1] Ne pas indenter les mots clef "public" et private". Concernant les constantes, fais un fichier constantes.h comme demandé dans la donnée du projet plutôt que de les mettre seulement dans les fonctions où tu en as besoin. | Bon code dans l'ensemble. Module lifeform très léger. Il faudrait déléguer certaines parties du code simulation qui sont trop spécifiques au module lifeform. | **6** |
| **373043** | **4** | [L1]lifeform.cc 120,126,127,200,210 | N'oubliez pas de réécrire lifeform à l'aide de l'héritage pour le rendu 2. Plutôt que d'écrire un if sans corps et utiliser le else, utilisez directement un if not() (simulation.cc, 55). Cela rend la condition plus lisible. Code très agréable à lire, bien aéré et compréhensible. Attention à utiliser des symbole plutôt que des indice pour rendre le code plus lisible (orientation dans shape.cc)L'overload des opérateur est une bonne idée. | **9** |
| **373083** | **5** | Warning: [L1] lifeform.cc: 116-121 | Bien joué! Je vois que vous êtes déjà prêt pour le prochain rendu. Pour ce qui est des noms de fonctions/variables essayez de rester sur 1 seul langue. Aussi pour les prochains rendus, je ne suis pas sûr qu'un vector contenant toutes les entités vous simplifiera la vie. A part ces deux points, super! | **10** |
| **373154** | **4** | [L1] shape.cc 33, lifeform.h 19,30, simulation.cc 219-222,235-238, lifeform.cc 96-98,143-152,155,159-164,167 | Attention pour la suite, argv[1] est accédé sans vérifier argc. Faites attention à votre indentation pour la suite, ça aidera aussi à rendre votre code plus facilement lisible. Autrement, la structure est bien respectée et le code bien modularisé. | **8** |

| | | | | |
|---|---|---|---|---|
| **373274** | **4** | [L1] : simulation.h l.14,15 simulation.cc l.58,68,112,143,165<br>lifeforms.cc l.12-18 | Bon travail !<br>Bon style de code dans l'ensemble, essayez tout de meme de découper d'avantage votre code.<br>La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **9** |
| **373282** | **5** | OK | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. | **10** |
| **373338** | **5** | Warning [L1] lifeform.cc : 94-95,191-192 | Excellent code ! Continuez ainsi | **10** |
| **373355** | **3** | [L1]project.cc:9-13,simulation.cc:128,140,157;[L2] lifeform.c:69-70,75-76,lifeform.h:30,58 | Code excellent, faites attention aux lignes trop longues avec les constructeurs. | **8** |
| **373398** | **3** | [L1]simulation.h61,64,66,simluation.cc77-82,shape.cc13-14 [P2]simulation.cc12-83,268-310 | Bonnes décisions d'architecture. Attention aux erreurs d'indentation et de style qui rendent le code plus difficile à lire. | **8** |
| **373412** | **3** | [L1]simulation.h61,64,66,simluation.cc77-82,shape.cc13-14 [P2]simulation.cc12-83,268-310 | Bonnes décisions d'architecture. Attention aux erreurs d'indentation et de style qui rendent le code plus difficile à lire. | **8** |
| **373418** | **5** | ok | Très bon travail et très beau code ; style parfait. Bravo, continuez comme ça ! | **10** |
| **373427** | **5** | OK | Code excellent et bonne modularisation. | **10** |
| **373443** | **4** | [L1] public/private indentés dans les interfaces, warning: if à une ligne possède une seule indetation selon la règle choisie, l.196 et l.140 ne respectent l'indentation choisie dans tous le code. | C'est un très bon travail, bien reflechi et cohérent. Votre code est clair et très lisible. Attention toutefois aux indentations qui vous font perdre des points. | **9** |
| **373447** | **5** | Excellent | Excellent code ! Continuez ainsi | **9** |
| **373465** | **5** | OK | The code is more or less readable and understandable, good job! But:<br>- Simulation should be a class to better control the data, the simulation and to be the single point of access to all vectors<br>- Choose a naming scheme (such as snake_case or camelCase for functions) and do not mix French and English<br>- Make all the getters as const, so you can avoid possible mistakes<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- In shape.cc, do not use a switch where you should have clearly used an if | **10** |

| | | | | |
|---|---|---|---|---|
| **373482** | **3** | [L1] Ne pas indenter les mots clef "public" et "private" dans tes classes, simulation.cc : 115,148 ; [P2] simulation.cc : 46-171, 253-307 (Méthodes qui dépassent les 40 et 80 lignes autorisées). Warning : les noms des classes ne respectent pas la convention du cours | Bon travail et code bien lisible. Bien penser à régler les soucis de prototypes de fonctions au mauvais endroit. Vous pouvez également passer des méthodes en private pour une meilleure encapsulation (classe simulation par exemple) | **6** |
| **373549** | **4** | [L1] public/private indentés dans les interfaces et lifeform.cc: constructeur Lifeform | Très bonne pratiques, code clair, lisible qui témoigne de bons reflexes. Faites attention aux public/private qui ne doivent pas être indentés et à vérfier à la fin que toutes vos indentations sont consistentes. De plus, la classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **9** |
| **373768** | **5** | ok | Très bon travail et très beau code ; style parfait. Bravo, continuez comme ça ! | **10** |
| **373799** | **5** | ok | code clair avec un bon usage des enums, et bonne décomposition en fonctions | **9** |
| **373831** | **4** | [L1] dans tous le code mauvaise indentation: 2 espaces aulieu de 4 [L2]warning:l.217 | Votre rendu témoigne d'un réel effort de bien faire et vos méthodes sont claires et bien écrites, mais faites attention à la cohérence et à la répétition de vos actions: je vous conseillerais pour le prochain rendu de bien appuyer la phase de conception papier crayon qui devrait éviter ces problèmes. | **6** |
| **373863** | **5** | [L1] shape.cc:11,22,37,47 [WARNING] [L2] constantes.h:4 [WARNING] | Bon code et bon style (hors quelques lignes où le style d'indentation ne colle pas avec le reste du code), mais faites attention à l'externalisation des méthodes: il ne faut pas définir de méthodes dans les fichiers .h! | **7** |
| **373877** | **4** | [L1] public/private indentés dans les interfaces | Bonne modularisation, pour que votre code soit davantage lisible pour les prochains rendus: diviser vos méthode en leur donnant une fonction spécifique, mettez les méthodes dont a pas besoin de l'accès en private et transformer vos structures en classe dans lifeform. Attention à l'ordre des élements (p.ex public/private)! | **8** |
| **373990** | **3** | [L1] Ne pas indenter les mots clef "public" et "private" dans tes classes, simulation.cc : 115,148 ; [P2] simulation.cc : 46-171, 253-307 (Méthodes qui dépassent les 40 et 80 lignes autorisées). Warning : les noms des classes ne respectent pas la convention du cours | Bon travail et code bien lisible. Bien penser à régler les soucis de prototypes de fonctions au mauvais endroit. Vous pouvez également passer des méthodes en private pour une meilleure encapsulation (classe simulation par exemple) | **6** |
| **373994** | **4** | [L1]projet.cc:10-13,cor.cc:16-17, | code globalement clair et très modulaire (bien que pas toujours nécessaire). La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **8** |

| | | | | |
|---|---|---|---|---|
| **374000** | **4** | [L1] lifeform.cc:17-20,25,32-35,... | Bon travail et beau code, mais faites attention à l'indentation et à l'architecture, et surtout attention à l'externalisation des méthodes: "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." | **6** |
| **374009** | **4** | [L1]simulation.cc 75;lifeform.cc 17-20,52,115; shape.cc 19, | Si un module doit déjà être inclu dans le .h, pas besoin de l'inclure à nouveau dans le .cc. N'hésitez pas à utiliser le .h pour déclarer la classe et à consacrer le .cc à son implémentation (simulation.cc) Les getters et les constructeur de moins d'une ligne peuvent figurer dans le .h Sinon le programme semble bien, bon travail. | **8** |
| **374108** | **4** | [L1]lifeform.h13-29, shape.h14-35, projet.cc9, shape.cc31 lifeform.h, shape.h : pas d'indentation pour les mots clés public et private, projet.cc et shape.cc : mauvais style pour le if (voir les conventions pour les styles acceptés) | Le code est globalement bon, mais l'encapsulation des classes gérant les entités pourrait être améliorée. En effet, la vérification des paramètres étant confiée au module simulation, cela rend vulnérable le module lifeform. Sinon, le code est clair, bon travail ! | **8** |
| **374239** | **4** | [L1]lifeform.h13-29, shape.h14-35, projet.cc9, shape.cc31 lifeform.h, shape.h : pas d'indentation pour les mots clés public et private, projet.cc et shape.cc : mauvais style pour le if (voir les conventions pour les styles acceptés) | Le code est globalement bon, mais l'encapsulation des classes gérant les entités pourrait être améliorée. En effet, la vérification des paramètres étant confiée au module simulation, cela rend vulnérable le module lifeform. Sinon, le code est clair, bon travail ! | **8** |
| **374440** | **4** | [L1] dans tous le code mauvaise indentation: 2 espaces aulieu de 4 [L2]warning:l.217 | Votre rendu témoigne d'un réel effort de bien faire et vos méthodes sont claires et bien écrites, mais faites attention à la cohérence et à la répétition de vos actions: je vous conseillerais pour le prochain rendu de bien appuyer la phase de conception papier crayon qui devrait éviter ces problèmes. | **6** |
| **374441** | **5** | | Super! Le code est très propre. Rien à redire. | **10** |
| **374657** | **5** | ok | Très beau travail, rendu très propre, bravo et continuez comme ça ! | **10** |
| **374686** | **4** | [L2] simulation.cc: l.61 [L1] public/private indentés dans les interfaces | Méthodes claires, bon choix de noms de variables et noms de méthode. Essayez de revoir les dépendances entre modules et les conventions sur les interfaces. La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **6** |
| **374686** | **4** | [L2] simulation.cc: l.61 [L1] public/private indentés dans les interfaces, simulation.cc fonction lecture: double indentation. | Méthodes claires, bon choix de noms de variables et noms de méthode. Essayez de revoir les dépendances entre modules et les conventions sur les interfaces. La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **6** |

| | | | | |
|---|---|---|---|---|
| **374966** | **4** | warning : plusieurs styles d'accolades dans la fonction lecture, [L2] projet.cpp:132,133,141-144 | Très bon code, c'est dommage il manque toute l'architecture, pensez à séparer projet.cpp en plusieurs module comme dit dans l'énoncé pour le rendu 2 et revoyez l'encapsulation pour les classes lifeforms | **4** |
| **375085** | **4** | [L1]shape.h 15-21,24-36,42-43, | Bon code, commentaires pertinent, bonne utilisation des classes. Bravo | **9** |
| **375108** | **4** | [L1] lifeform.cc : 31,34,37,132, shape.cc : 90. Warning [L2] lifeform.cc : 128, shape.cc : 83,90. Attention à tes if à une seule instruction : si tu décides de mettre l'instruction entre accolade respecte les conventions (tout sur la même ligne ou saut de ligne pour l'instruction et le "}") | Très bon code, continuez ainsi ! | **8** |
| **375121** | **3** | [L1] lifeform.cc 14,19,47, shape.cc 10 [P2] simulation.cc decodage_ligne | Si un module doit déjà être inclu dans le .h, pas besoin de l'inclure à nouveau dans le .cc. Attention, les lignes de simulation.cc 62,80 et de lifeform.cc 105 sont trop longues. Dans l'ensemble le code est bon. Bon travail | **8** |
| **375202** | **5** | [L1] shape.cc 23 | Attention pour la suite, argv[1] est accédé sans vérifier argc. Autrement le code est très clair, bien structuré et modularisé, excellent travail! | **10** |
| **375206** | **3** | [L1] simulation.cc:49-54, lifeform.cc:187 [L2] simulation.cc:60,80 simulation.h:19-20, lifeform.h:26,72 | Bon travail et bon rendu, en dehors de quelques erreurs mineures de style et d'architecture. C'est un peu étranger d'appeler votre classe simulation "Decodeline", peut-être serait-il mieux de changer le nom pour Simulation d'ici le prochain rendu ? Bizarre aussi de déclarer "static constexpr double epsil_zero = 0.5;" au milieu du fichier lifeform.h (ligne 46) | **7** |
| **375214** | **4** | [L1]lifeform.cc 135,143,145,163, | Pour le rendu 2 il faudra convertir lifeform en hierarchie de classe. Attention au nom de paramètre à une lettre. Bon code dans l'ensemble, bon travail | **9** |
| **375236** | **4** | [L1] : shape.cc l.21,22 lifeform.cc l.35,71,125,136 warning [L2] : simulation.cc l.128 | Bon travail, faites attention à votre utilisation des attribut static. | **8** |

| | | | | |
|---|---|---|---|---|
| **375353** | **5** | OK | The code is more or less readable and understandable, good job! But:<br>- Simulation should be a class to better control the data, the simulation and to be the single point of access to all vectors<br>- Choose a naming scheme (such as snake_case or camelCase for functions) and do not mix French and English<br>- Make all the getters as const, so you can avoid possible mistakes<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- In shape.cc, do not use a switch where you should have clearly used an if | **10** |
| **375361** | **4** | [L1]lifeform.cc 120,126,127,200,210 | N'oubliez pas de réécrire lifeform à l'aide de l'héritage pour le rendu 2. Plutôt que d'écrire un if sans corps et utiliser le else, utilisez directement un if not() (simulation.cc, 55). Cela rend la condition plus lisible. Code très agréable à lire, bien aéré et compréhensible. Attention à utiliser des symbole plutôt que des indice pour rendre le code plus lisible (orientation dans shape.cc)L'overload des opérateur est une bonne idée. | **9** |
| **375362** | **5** | OK | The code is very readeable and understandable, good job! Good modularity. For the future:<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Pay attention to some narrowing conversion as they can be the source of bugs<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- You define all your vectors as static global vector, but it would be better to be an attribute of your class Simulation so you can easily access them and pass them around<br>- In project.cc you call the method lecture through an instance of simulation, but since it is a static method, there is no need istantiate one (or change it to a non static method)<br>- You could use pointer/unique_ptr to store your instances in the vectors for performance (since you avoid copying them) | **10** |
| **375364** | **5** | Excellente lisibilité du code, vous pourriez cependant renomme la classe Lecture qui aura d'autres fonctions que la lecture dès les prochains rendus | Très bon travail, vous pourriez utiliser le type S2d crée dans shape pour stocker la position des entités | **9** |

| | | | | |
|---|---|---|---|---|
| **375390** | **4** | [L1] simulation.cc 26-27, lifeform.h (général) double indentation, lifeform.h 121, lifeform.cc 125-126,286-287. simulation.cc plusieurs styles d'accolades incohérents | Inclusion de constantes.h dans shape à corriger, et faites attention à l'externalisation des définitions de constructeurs. Sinon, le code est bien lisible, aéré et bien structuré. | **7** |
| **375399** | **5** | OK | Code excellent et bonne modularisation. | **10** |
| **375427** | **4** | [P2] simulation.cc:decodage_ligne() | Bon travail et bon code qui respect les conventions et l'architecture ; attention simplement à la fonction decodage_ligne qui est absolument colossale par rapport au nombre de lignes max autorisé pour les fonctions du projet. Mettez en place les principes d'abstraction et d'encapsulation afin de subdiviser les tâches pour finir avec une fonction plus courte. | **9** |
| **375454** | **4** | [P5] Variables avec majuscule pour tous les attributs de classes, les variables doivent etre en minuscules ([E12] dans les Conventions de programmations). Warning [L1] attention au changement de style d'indentation dans votre code, par exemple dna shape.cc l.79 vs l.88 | Très beau code, très propre. Surement le reflet d'un bon niveau de programmation. Bravo continuez comme ca. | **9** |
| **375632** | **5** | [L1] shape.cc 23 | Attention pour la suite, argv[1] est accédé sans vérifier argc. Autrement le code est très clair, bien structuré et modularisé, excellent travail! | **10** |
| **376115** | **3** | [L1]project.cc:9-13,simulation.cc:128,140,157;[L2] lifeform.c:69-70,75-76,lifeform.h:30,58 | Code excellent, faites attention aux lignes trop longues avec les constructeurs. | **8** |
| **376265** | **4** | [P5] Variables avec majuscule pour tous les attributs de classes, les variables doivent etre en minuscules ([E12] dans les Conventions de programmations). Warning [L1] attention au changement de style d'indentation dans votre code, par exemple dna shape.cc l.79 vs l.88 | Très beau code, très propre. Surement le reflet d'un bon niveau de programmation. Bravo continuez comme ca. | **9** |

| | | | | |
|---|---|---|---|---|
| **376501** | **4** | [L1] Simulation.h 11, lifeforms.cc 88, 169-172, 208-209,212, etc. Général: le style d'indentation d'accolades est à corriger. Warning [L2] lifeforms.cc 140. Warning [P5] Simulation::compteur, Simulation::total et Simulation::i très peu explicites pour des attributs static de classe. | Le style d'accolades est à corriger (utiliser une des variantes proposées dans les conventions). Cela rend votre code assez difficile à lire pour une personne externe, surtout lorsqu'il y a plusieurs niveaux d'indentation. Vos attributs static de Simulation peuvent probablement être transférés dans le .cc ou même dans une fonction; cela les rendra plus compréhensibles. Sinon, la structure générale est bien respectée et le code est bien modularisé. | **6** |
| **376698** | **4** | [P2] simulation.cc:decodage_ligne() | Bon travail et bon code qui respect les conventions et l'architecture ; attention simplement à la fonction decodage_ligne qui est absolument colossale par rapport au nombre de lignes max autorisé pour les fonctions du projet. Mettez en place les principes d'abstraction et d'encapsulation afin de subdiviser les tâches pour finir avec une fonction plus courte. | **9** |
| **376735** | **3** | [L1] simulation.cc:49-54, lifeform.cc:187 [L2] simulation.cc:60,80 simulation.h:19-20, lifeform.h:26,72 | Bon travail et bon rendu, en dehors de quelques erreurs mineures de style et d'architecture. C'est un peu étranger d'appeler votre classe simulation "Decodeline", peut-être serait-il mieux de changer le nom pour Simulation d'ici le prochain rendu ? Bizarre aussi de déclarer "static constexpr double epsil_zero = 0.5;" au milieu du fichier lifeform.h (ligne 46) | **7** |
| **376800** | **5** | Very clean code and style | Very solid base projects, well done, keep up the good work | **10** |
| **376866** | **5** | Very clean code and style | You guys are wizards, many c++ features not showned in class, unexpected methods. I saw a lot of code style used by advanced programmers. But the code is good so keep up the good work! | **10** |
| **377004** | **4** | [L1] lifeform.cc: 35, 201, 238-239, 266; shape.cc: 17-22, 25-30, 30 | Bien joué! Le code est très lisible. C'est dommage pour les petites erreurs d'indentations. | **8** |
| **377096** | **5** | OK | The code is very readeable and understandable, good job! Good modularity. For the future:<br>- It makes more sense for nextLine to be in simulation.cc and not lifeform.cc<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing<br>- Try to name your variables and functions with better names (do not use abbrevations) | **10** |
| **377272** | **5** | Ok | Excellent ! Très bon respect des convention. Votre code est très lisible. Bonne architecture et modularisation. Continuez ainsi | **10** |
| **377774** | **5** | OK | Code excellent et bonne modularisation. Le fichier entities.cc n'est pas utile (il y a seulement include entities.h). | **9** |

| | | | | |
|---|---|---|---|---|
| **378145** | **5** | Very clean code and style | You guys are wizards, many c++ features not showned in class, unexpected methods. I saw a lot of code style used by advanced programmers. But the code is good so keep up the good work! | **10** |
| **378335** | **5** | OK | The code is very readeable and understandable, good job! Good modularity. For the future:<br>- It makes more sense for nextLine to be in simulation.cc and not lifeform.cc<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing<br>- Try to name your variables and functions with better names (do not use abbrevations) | **10** |
| **378336** | **5** | Ok | Bravo! le code est limpide. Un plaisir à corriger :) | **10** |
| **378408** | **4** | [L1]lifeform.cc18, 32-34, 51-52; lifeform.h47; simulation.cc122-123, 132, 147 | Bon travail dans l'ensemble, votre code n'est cependant pas le plus clair, et vous pourriez faire preuve d'un peu plus d'abstraction et de séparation des responsabilités afin de rendre les choses plus lisibles. Je suis également convaincu que ça vous aiderait à rendre les erreurs d'indentation moins fréquentes | **8** |
| **378490** | **2** | [L1] lifeform.cc 80,84-105, 117, 120, ....<br>[L2] lifeform.cc 67-68, 94, 103, 142, ...<br>[P2] simulation.cc testCorail, project.cc lire_fichier | Code illisible. Les conventions ne sont pas du tout respectées, le code n'est pas indenté. On retrouve régulièrement des lignes de 100-130 caractères... | **6** |
| **378517** | **3** | [P2] decodage_ligne (144 lines), [L1] lifeform.h34-39, 62-67 | The code is convoluted and very hard to read. It looks like that it was rushed through... try to simplify it and organize it:<br>- When you don't need access to the current index of a loop, use modern range-based for-loops<br>- Use a formatting tool such as that provided in VS Code in order to fix the overall inconsistent spacing and indentation<br>- Some functions (such decodage_ligne) are massive, hard to read and convoluted try to break them in smaller functions, especially since it is above the maximal threshold<br>- Stick to a naming scheme (such as snake_case or camelCase for functions) and do not mix between them or between French and English<br>- Even if it is allowed, do not use multiple if-else without brackets, because it lowers the lisibility<br>- Try to name your variables and functions with better names<br>- You can only define getters and constructors in the interface and they must be in only one line | **5** |
| **378523** | **5** | Ok | Excellent ! Très bon respect des convention. Votre code est très lisible. Bonne architecture et modularisation. Continuez ainsi | **10** |
| **378533** | **4** | [L1] simulation.h 19-20,30,32,34,36,45, shape.h 19, shape.cc 89, etc. | Attention pour la suite, argv[1] est accédé sans vérifier argc. Faites attention à l'alignement des instructions sur plusieurs lignes. Sinon, le code est très clair et bien structuré, bravo ! | **9** |

| | | | | |
|---|---|---|---|---|
| **378614** | **3** | [L1] lifeform.cc 14,19,47, shape.cc 10<br>[P2] simulation.cc decodage_ligne | Si un module doit déjà être inclu dans le .h, pas besoin de l'inclure à nouveau dans le .cc.<br>Attention, les lignes de simulation.cc 62,80 et de lifeform.cc 105 sont trop longues. Dans l'ensemble le code est bon. Bon travail | **8** |
| **378660** | **4** | [L1] : simulation.h l.11,15; simulation.cc l.127,140; shape.cc l.69<br>warning [L2] : simulation.cc l.74 | Très bon travail, code très lisible et bonne modularisation.<br>Rq : t'as pas besoin de mettre un ";" à la fin de tes méthodes | **8** |
| **378744** | **5** | warning : ligne trop longue shape.h:37 | Code excellent et bonne modularisation. Le fichier lifeform.cc est assez long, peut-être le découper en plusieurs modules?? | **10** |
| **378782** | **5** | Ok | Un très bon travail dans l'ensemble, attenation cependant à vos statics vectors qui contiennent vos entités. Un très bon style sinon. | **10** |
| **378979** | **4** | [L2]ignorance totale de la taille max des lignes à 87 caractères | bonne structuration ; code très lisible mais la contrainte de taille max de ligne est totalement ignorée. | **8** |
| **379052** | **5** | OK | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. | **10** |
| **379114** | **4** | [L1] public/private indentés dans les interfaces et lifeform.cc: constructeur Lifeform | Très bonne pratiques, code clair, lisible qui témoigne de bons reflexes. Faites attention aux public/private qui ne doivent pas être indentés et à vérfier à la fin que toutes vos indentations sont consistentes. De plus, la classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **9** |
| **379158** | **5** | OK | Excellent! Votre code est clair, cohérent et homogène. Vous avez de bons reflexes de programmation et vous faites bien attention aux conventions. Continuez ainsi! | **10** |
| **379194** | **4** | [L1] public/private indentés dans les interfaces et lifeform.cc: l.50, l.71 [L2] lifeform.h: l21,l.23,l.41,l.10 | Très bonne modularisation et très bonne architecture, attention toutefois à la clarté du code: essayez d'aerer votre code (en sautant des lignes), réferez vous aux conventions pour l'ordre des élements. | **9** |
| **379270** | **4** | [L1] Ne pas indenter les mots clef "public" et private". Concernant les constantes, fais un fichier constantes.h comme demandé dans la donnée du projet plutôt que de les mettre seulement dans les fonctions où tu en as besoin. | Bon code dans l'ensemble. Module lifeform très léger. Il faudrait déléguer certaines parties du code simulation qui sont trop spécifiques au module lifeform. Pour le prochain rendu pensez bien à externaliser vos méthodes et constructeurs. | **6** |
| **379329** | **5** | Les "subfunctions" dans shape.cc peuvent être placées dans un unnamed namespace pour s'assurer que ces fonctions restent statiques au fichier. | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. | **10** |
| **379343** | **5** | Warning simulation.cc l.21 ? | Très bon projet, bien structuré avec un bon style, continuez comme ca! | **9** |
| **379344** | **5** | Excellent | Excellent code ! Continuez ainsi | **9** |

| | | | | |
|---|---|---|---|---|
| **379375** | **5** | [L1] simulation.h 8,24, lifeform.h 32,35,... | Attention, la ligne 94 de shape.cc est trop longue. Les mots-clés public,private et protected ne doivent pas être indentés. Attention aux nom de variables à une seule lettre. Code élégant et de bonne facture, bravo. | **8** |
| **379382** | **1** | [L1] projet.cc:7-8 simulation;cc:33-45,... lifeform.cc:36-37,... etc<br>[L2] simulation.cc:33,58,60 shape.cc:16,48,... shape.h:31 etc<br>[P2] simulation.cc:Simulation::lecture()<br>[P5] shape.cc:82,102,105,108,111 | Du travail a été fourni et cela est apprécié, cela-dit le style du code de votre rendu est assez désastreux. L'indentation est particulièrement bancale à peu près partout, essayez de revoir ça d'ici le prochain rendu SVP. Il y a également des fonctions trop longues et des lignes trop longues, ainsi que des magic numbers qui rendent le code d'autant plus difficile à lire (utilisez des enum ou des constexpr). Attention également aux attributs publics qui sont nombreux dans lifeform.h | **3** |
| **379386** | **4** | [L1] public/private indentés dans les interfaces et lifeform.cc: l.50, l.71 [L2] lifeform.h: l21,l.23,l.41,l.10 | Très bonne modularisation et très bonne architecture, attention toutefois à la clarté du code: essayez d'aerer votre code (en sautant des lignes), réferez vous aux conventions pour l'ordre des élements. | **9** |
| **379509** | **5** | OK | Bel effort sur les commentaire, peut-être un peu trop nombreux. (bravo pour la mention du forum où un morceau code à été repris, très bonne pratique). Belle décomposition. Toute petite erreur d'indentation dans simulation.cc ligne 190, sinon très bon travail. | **10** |
| **379550** | **4** | [P2]fonction lineDecoding in simulation.cc<br>Warning : wrapping line in lifeform.h56 | Très bon travail, le code est plutôt clair dans l'ensemble, vous pourriez juste user d'un peu plus d'abstraction pour réduire la taille de la fonction trop longue. À part ça, le travail est bien fait. | **8** |
| **379654** | **5** | Ok | Bien joué! Code est lisible et clair. Faites attention à bien externaliser vos méthodes. Dans votre décodage faites tout de même attention à trouver des noms plus parlants que caseCorail, switchEtat, etc, ... Ces fonctions n'ont pas un but claire à part éviter de passez au-dessus des 80 lignes :) Dernier point, vous avez quelques if statements à plusieurs endroit de votre code qui ne font rien, il me semble: ex: les if au début de chaque case dans decodage ou les if dans les fonctions caseScaven, etc, ... | **6** |
| **379659** | **4** | [L1] lifeform.cc: 35, 201, 238-239, 266; shape.cc: 17-22, 25-30, 30 | Bien joué! Le code est très lisible. C'est dommage pour les petites erreurs d'indentations. | **8** |
| **379660** | **5** | [L1] simulation.h 8,24, lifeform.h 32,35,... | Attention, la ligne 94 de shape.cc est trop longue. Les mots-clés public,private et protected ne doivent pas être indentés. Attention aux nom de variables à une seule lettre. Code élégant et de bonne facture, bravo. | **8** |
| **379664** | **5** | Ok | Travail de très grande qualité. Très beau style de code. Excellente modularisation et architecture.<br>Continuez ainsi | **9** |

| | | | | |
|---|---|---|---|---|
| **379714** | **4** | [L1] lifeform.cc:17-20,25,32-35,... | Bon travail et beau code, mais faites attention à l'indentation et à l'architecture, et surtout attention à l'externalisation des méthodes: "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." | **6** |
| **379723** | **3** | [L1] lifeform.cc: 217-220,225-229,234-239; shape.cc: 13-18<br>[P5] lifeform.cc 59,87,94,96,109... | Bien joué! Globalement le code est bien écrit, mais votre approche des vector pour les constructeurs d'entités est assez dur à lire surtout si ce que représente les différents indices des vecteurs n'est pas explicité (un enum améliorerait la lisibilité). Attention au fautes de double indentations. | **8** |
| **379734** | **3** | [L1] lifeform.cc: 217-220,225-229,234-239; shape.cc: 13-18<br>[P5] lifeform.cc 59,87,94,96,109... | Bien joué! Globalement le code est bien écrit, mais votre approche des vector pour les constructeurs d'entités est assez dur à lire surtout si ce que représente les différents indices des vecteurs n'est pas explicité (un enum améliorerait la lisibilité). Attention au fautes de double indentations. | **8** |
| **379816** | **4** | warning : plusieurs styles d'accolades dans la fonction lecture, [L2] projet.cpp:132,133,141-144 | Très bon code, c'est dommage il manque toute l'architecture, pensez à séparer projet.cpp en plusieurs module comme dit dans l'énoncé pour le rendu 2 et revoyez l'encapsulation pour les classes lifeforms | **4** |
| **379827** | **4** | [L1] simulation.cc 18-19,21, 214-215,232,244-245, etc. Général: double indentation des classes | Faites attention aux inclusions inutiles qui ne respectent pas l'architecture alors que vous ne les utilisez pas. Autrement, le code est plutôt clair et bien structuré. | **9** |
| **379847** | **5** | Warning [L2] lifeform.cc l.275, 421 | Well structured code, but again maybe try to displace some functions in other moduiles as lifeform has already 500+ lines of code and you are only at rendu1 | **10** |
| **379874** | **4** | [P2]fonction lineDecoding in simulation.cc<br>Warning : wrapping line in lifeform.h56 | Très bon travail, le code est plutôt clair dans l'ensemble, vous pourriez juste user d'un peu plus d'abstraction pour réduire la taille de la fonction trop longue. À part ça, le travail est bien fait. | **8** |
| **379882** | **5** | OK | Attention pour la suite, argv[1] est accédé sans vérifier argc. Préférez <cmath> à <math.h>, et essayez d'enlever les #include inutiles dans les .h. Pour la suite, faites attention à l'externalisation des définitions des constructeurs. Autrement, le code est très clair et lisible, rien à redire sur le style! | **8** |
| **379889** | **3** | [L1]projet.cc 11-16, simulation.cc 28,62,119, lifeform.h 33,50,...<br>[L2]lifeform.h 32,33,49,50; lifeform.cc 50,60; simulation.cc 39 | Attention, pour le rendu 2 une classe simulation sera nécessaire. Bon usage des enum. Attention aux constante. (epsil de shape.cc dans segmentSuperposition meriterait d'être une constexpr au début de shape.cc) Attention aux quelques erreurs d'inattention, sinon bonne base pour la suite. | **8** |
| **379895** | **4** | [L1] lifeform.cc méthode get_id, simulation.cc: l.39,l.47,l.97 double indentation, l.18 deux espaces aulieu de 4. | Rendu très bien organisé, respect des principes d'abstraction et très bonne modularisation. Attention à vos indentations. | **9** |

| | | | | |
|---|---|---|---|---|
| **379911** | **4** | [L1] Warning lifeform.cc l.36-54 3 indentations au lieu de 1 [L2] Simulation.h l.22-24 et Simulation.cc l.73,128,134265 | Excellent code, tres propre et tres bien structuré. Ca semble refeleter un très bon niveau de programmation. Attention cependant au nb max de character par ligne. | **9** |
| **379915** | **4** | [L1] : shape.cc l.179-203; prog.cc l.21;io.cc l.65-121, l.202-205, l.215, l.248-269, l.338, l.378-388 warning [L2]: lifeform.h l.50, io.cc l.233,239,385 | Bon travail, votre style de code est globalement très bon mais les conventiuons sont parfois complètement oubliées. Vous devez repenser votre architecture concernant les modules projet et simulation. La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **8** |
| **379988** | **5** | OK | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. | **10** |
| **379991** | **5** | Les "subfunctions" dans shape.cc peuvent être placées dans un unnamed namespace pour s'assurer que ces fonctions restent statiques au fichier. | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. | **10** |
| **380052** | **5** | Ok | Un très bon travail, le rendu est propre, bravo ! | **10** |
| **380080** | **5** | ok | Très beau travail, rendu très propre, bravo et continuez comme ça ! | **10** |
| **380082** | **5** | Ok | Excellent travail dans l'ensemble, le code est très bon, voici quelque points à améliorer que j'ai noté :<br>Le module simulation pourrait être largement dissequé en sous-tâche, ce qui éviterait l'unique et massive fonction de lecture.<br>Votre manière d'utiliser/ne pas utiliser espsil_0 n'est pas optimal car le nom de la variable est ambigu.<br>Vous devriez utiliser une variable définie par un constexpr par exemple pour votre epsil_rot. | **10** |
| **380090** | **3** | [L1]simulation.h21-48,52-151,... [L2]simulation.cc97,107,109,127,155... | La définition dans l'interface de la classe n'est accepté seulement si cette définition peux tenir sur la même ligne que le prototype de la fonction. L'indentation est mauvaise un peu partout, faites attention car le code devient illisible. | **6** |
| **380097** | **5** | Ok | Bravo! le code est limpide. Un plaisir à corriger :) | **10** |
| **380105** | **5** | Warning [L1] trop de styles d'indentation differentes dans lifeform.cc | Très bon travail, code très bien structuré, continuez comme ca! | **10** |
| **380137** | **3** | [L1]simulation.h21-48,52-151,... [L2]simulation.cc97,107,109,127,155... | La définition dans l'interface de la classe n'est accepté seulement si cette définition peux tenir sur la même ligne que le prototype de la fonction. L'indentation est mauvaise un peu partout, faites attention car le code devient illisible. | **6** |
| **380154** | **5** | Warning indentation shape.cc l.21,25 | Excellent code très bien structuré avec un beau style, continuez ainsi! Attention cependant au variables globales | **9** |

| | | | | |
|---|---|---|---|---|
| **380200** | **4** | [L1] simulation.h:13-20 shape.cc:61,73,... | Bon travail, bon respect général du style et de l'architecture. Attention, l'indentation est parfois hasardeuse (tantôt 2 espaces, tantôt 4, ...) et il ne faut pas indenter les mots-clés public et private. Attention aussi à ne pas laisser d'attributs publics quels qu'ils soient | **8** |
| **380211** | **4** | [L2]Algue.cc:18,Corail.cc:4,21,22,Corail.h :24 | Code globalement clair et très structuré, faites attention à la taille des lignes pour les constructeurs et les prints | **7** |
| **380226** | **4** | [L1] toutes vos classes ont une indentation de trop. Les mots de clé private/public ne doit pas avoir d'indentation. | Code excellent et très bonne modularisation. Faites attention à l'indentation dans les classes. | **9** |
| **380234** | **4** | [L1] public/private indentés dans les interfaces | Bonne modularisation, pour que votre code soit davantage lisible pour les prochains rendus: diviser vos méthode en leur donnant une fonction spécifique, mettez les méthodes dont a pas besoin de l'accès en private et transformer vos structures en classe dans lifeform. Attention à l'ordre des élements (p.ex public/private)! | **8** |
| **380279** | **4** | [L1] : shape.cc l.179-203; prog.cc l.21;io.cc l.65-121, l.202-205, l.215, l.248-269, l.338, l.378-388 warning [L2]: lifeform.h l.50, io.cc l.233,239,385 | Bon travail, votre style de code est globalement très bon mais les conventiuons sont parfois complètement oubliées. Vous devez repenser votre architecture concernant les modules projet et simulation. La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **8** |
| **380349** | **4** | [L1]simulation.h, shape.h, lifeform.h : surindentation des classes, simulation.cc15-91, shape.cc32-51, lifeform.cc11-19, 32-36, 54-61 | Le travail est bien réalisé, mais attention à garder votre code propre et lisible, ce qui en facilitera l'amélioration et le debug. | **8** |
| **380446** | | | | |
| **380459** | **5** | Warning [L1] trop de styles d'indentation differentes dans lifeform.cc | Très bon travail, code très bien structuré, continuez comme ca! | **10** |
| **380517** | **4** | [L1]simulation.cc 75;lifeform.cc 17-20,52,115; shape.cc 19, | Si un module doit déjà être inclu dans le .h, pas besoin de l'inclure à nouveau dans le .cc. N'hésitez pas à utiliser le .h pour déclarer la classe et à consacrer le .cc à son implémentation (simulation.cc) Les getters et les constructeur de moins d'une ligne peuvent figurer dans le .h Sinon le programme semble bien, bon travail. | **8** |
| **380538** | **4** | [L1]lifeform.h33-61,65-87,simulation.h10-17,shape.h13-22 | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. Attention double indentation dans la définition de 4 classes. | **9** |
| **380606** | **5** | Warning [L1] simulation.cc 274,276, lifeform.cc 58,178 | L'inclusion de fichiers du Modèle dans le module shape est à corriger. Style: les noms de classe devraient commancer par une majuscule, et faites attention à l'alignement de paramètres sur plusieurs lignes. Sinon, le code est propre et bien lisible. | **8** |
| **380610** | **4** | [L1] simulation.cc : 191-194, lifeform.cc : 78-82 | Très bon style de code, pour le prochain rendu pensez à bien externaliser vos constructeurs et méthodes | **6** |

| | | | | |
|---|---|---|---|---|
| **380642** | **5** | Excellente lisibilité du code, vous pourriez cependant renomme la classe Lecture qui aura d'autres fonctions que la lecture dès les prochains rendus | Très bon travail, vous pourriez utiliser le type S2d crée dans shape pour stocker la position des entités | **9** |
| **380661** | **5** | OK | Code excellent et bonne modularisation. | **10** |
| **380664** | **4** | [P2] simulation.cc : 159-207 | Excellent code ! Continuez ainsi. PS : pas besoin de mettre de " ; " après une " } " pour tes if, for, ... | **9** |
| **380673** | **5** | OK | Faites juste attention à l'externalisation des méthodes, mais sinon il s'agit d'un excellent travail, bravo ! | **8** |
| **380682** | **5** | Warning [L1] lifeform.cc : 94-95,191-192 | Excellent code ! Continuez ainsi | **10** |
| **380756** | **5** | OK | Bon code, belle organisation, bravo | **10** |
| **380773** | **4** | [L1] lifeform.h: 25-29, 33-36, 40-52, 57-61, ...<br>Ne pas indenter public et private<br>[L1] simulation.cc: 77-82,104-195,121-126; simulation.h:4-86<br>[L2] warning: shape.cc 68 | Bon travail! Le code est clair et lisible. Dommage pour les quelques fautes d'indentations. Bien joué! | **9** |
| **380788** | **4** | [L1] lifeform.h, il faut indenter les attributs et les methodes dans tout le code comme vous l'avez fait dans simulation.h ([L17]Conventions de programmation) | Bonne base de projet | **9** |
| **380883** | **4** | [L1]lifeform.h:19,51,63,77,84,107,simulation.h:13,26 | Code globalement clair et très structuré, pensez à ne pas indenter public et private dans les classes, je n'ai pas compris à quoi servait #define _USE_MATH_DEFINES, est-ce util? | **8** |
| **380931** | **5** | Ok | Bien joué! Code est lisible et clair. Faites attention à bien externaliser vos méthodes. Dans votre décodage faites tout de même attention à trouver des noms plus parlants que caseCorail, switchEtat, etc, ... Ces fonctions n'ont pas un but claire à part éviter de passez au-dessus des 80 lignes :) Dernier point, vous avez quelques if statements à plusieurs endroit de votre code qui ne font rien, il me semble: ex: les if au début de chaque case dans decodage ou les if dans les fonctions caseScaven, etc, ... | **6** |
| **380963** | **3** | [P2]lecture>100lignes;<br>[L1]simulation.cc166-188,lifeform.cc42, | respectez l'architecture complète du projet ; ajoutez -std=c++17 pour la compilation ; pi pourrait être plus précis en double precision ; pas clair comment epsil_zero sera géré dans le cas général ; aérez plus la définition de vos fonctions/méthodes dans lifeform et externalisez celle de simulation ; mettre en oeuvre le principe d'abstraction pour lecture() | **6** |

| | | | | |
|---|---|---|---|---|
| **380988** | **4** | [L1] projet.cc 22.30, shape.cc 20,47, lifeform.h 21. Remarque: essayez d'éviter trop de niveaux d'indentation dans vos méthodes isValid | Certains #include ne sont pas nécessaires dans vos .h, essayez de les minimiser. Bien structuré et présentation claire et lisible, bravo ! | **8** |
| **381001** | **5** | Warning simulation.cc  l.21 ? | Très bon projet, bien structuré avec un bon style, continuez comme ca! | **9** |
| **381013** | **5** | OK | Faites juste attention à l'externalisation des méthodes, mais sinon il s'agit d'un excellent travail, bravo ! | **8** |
| **381098** | **4** | [L1]lifeform.cc18, 32-34, 51-52; lifeform.h47; simulation.cc122-123, 132, 147 | Bon travail dans l'ensemble, votre code n'est cependant pas le plus clair, et vous pourriez faire preuve d'un peu plus d'abstraction et de séparation des responsabilités afin de rendre les choses plus lisibles. Je suis également convaincu que ça vous aiderait à rendre les erreurs d'indentation moins fréquentes | **8** |
| **381128** | **5** | Warning: [L1] lifeform.cc: 116-121 | Bien joué! Je vois que vous êtes déjà prêt pour le prochain rendu. Pour ce qui est des noms de fonctions/variables essayez de rester sur 1 seul langue. Aussi pour les prochains rendus, je ne suis pas sûr qu'un vector contenant toutes les entités vous simplifiera la vie. A part ces deux points, super! | **10** |
| **381136** | **5** | Warning [L2] lifeform.cc l.275, 421 | Well structured code, but again maybe try to displace some functions in other moduiles as lifeform has already 500+ lines of code and you are only at rendu1 | **10** |
| **381157** | **5** | Attention : shape.cc ligne 15 : double indentation; lifeform.cc ligne 73 | C'est un bon travail, attention cependant à vos tests et à votre style de code qui n'est pas toujours constant | **10** |
| **381162** | **5** | warning : une indentation de trop dans le fichier shape.cc ligne 8 à 10 | Code excellent et bien modulariser, pensez à revoir la classe Simulation cela pourrait vous aider pour le rendu 2 | **9** |
| **381171** | **5** | OK | Bon code, belle organisation, bravo | **10** |
| **381175** | **4** | [L1] lifeform.cc : 118-128, simulation.cc : 109-111, 160-192 ; Warning [L2] dans lifeform.cc (l.113,114,131) | Très bon code, très lisible et compréhensible. Attention à ne pas changer de style d'indentation au cours du code. Pensez à faire un fichier "constantes.h" pour les constantes du projet (pour ne pas les avoir dans lifeform.h) | **7** |
| **381229** | **5** | OK | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. | **10** |
| **381243** | **4** | [L2]Algue.cc:18,Corail.cc:4,21,22,Corail.h :24 | Code globalement clair et très structuré, faites attention à la taille des lignes pour les constructeurs et les prints | **7** |
| **381357** | **5** | Ok | Excellent travail dans l'ensemble, le code est très bon, voici quelque points à améliorer que j'ai noté :<br>Le module simulation pourrait être largement dissequé en sous-tâche, ce qui éviterait l'unique et massive fonction de lecture.<br>Votre manière d'utiliser/ne pas utiliser espsil_0 n'est pas optimal car le nom de la variable est ambigu.<br>Vous devriez utiliser une variable définie par un constexpr par exemple pour votre epsil_rot. | **10** |

| | | | | |
|---|---|---|---|---|
| **381391** | **4** | [L1]lifeform.h33-61,65-87,simulation.h10-17,shape.h13-22 | Code très clair et modulaire, très bonnes décisions d'architecture et très bon style. Attention double indentation dans la définition de 4 classes. | **9** |
| **381393** | **4** | [L1] simulation.cc 18-19,21, 214-215,232,244-245, etc. Général: double indentation des classes | Faites attention aux inclusions inutiles qui ne respectent pas l'architecture alors que vous ne les utilisez pas. Autrement, le code est plutôt clair et bien structuré. | **9** |
| **381402** | **3** | [L1] projet.cc : 10, simulation.cc : 50,65,67,72,76,78,...,199-203,222-227,..., lifeform.cc : 91,104,113,131,..., shape.cc : 19,54,58,107,108. [P2] simulation.cc : decodage. WARNING : Ne changez pas de style d'accolade au cours du code ! Beaucoup de lignes trop ou pas assez indentée. | Bon code. Attention à bien garder le même style de code tout le programme, à mettre vos accolades au bon endroit, etc (voir conventions de programmation du cours). Pour vos fonctions/méthodes/constructeurs avec beaucoup de paramètres,je vous recommande de suivre la règle [L22] et de mettre le plus de paramètres sur une seule ligne pour plus de lisibilité. | **8** |
| **381646** | **4** | [L1]projet.cc:10-13,cor.cc:16-17, | code globalement clair et très modulaire (bien que pas toujours nécessaire). La classe simulation était exigée dès le rendu1. il y aura pénalité dès le rendu2 en cas d'absence de cette classe. | **8** |
| **381765** | **4** | [L1] lifeform.cc:139-143 | Beau travail et beau style, faites néanmoins attention à l'architecture car shape doit être indépendant du modèle. | **8** |
| **381793** | **4** | [L1] : le fichier lifeform.h est mal configuré -> indentation de 3 espaces | Très bon travail, bon style de code, bonne architecture et bonne modularisation, continuez ainsi | **9** |
| **381905** | **4** | [P2] simulation.cc : 159-207 | Excellent code ! Continuez ainsi. PS : pas besoin de mettre de " ; " après une " } " pour tes if, for, ... | **9** |
| **381925** | **5** | Ok | Un très bon travail dans l'ensemble, attenation cependant à vos statics vectors qui contiennent vos entités. Un très bon style sinon. | **10** |
| **381964** | **4** | [L1] public/private indentés dans les interfaces, warning: if à une ligne possède une seule indetation selon la règle choisie, l.196 et l.140 ne respectent l'indentation choisie dans tous le code. | C'est un très bon travail, bien reflechi et cohérent. Votre code est clair et très lisible. Attention toutefois aux indentations qui vous font perdre des points. | **9** |
| **381993** | **1** | [L1] projet.cc:7-8 simulation;cc:33-45,... lifeform.cc:36-37,... etc<br>[L2] simulation.cc:33,58,60 shape.cc:16,48,... shape.h:31 etc<br>[P2] simulation.cc:Simulation::lecture()<br>[P5] shape.cc:82,102,105,108,111 | Du travail a été fourni et cela est apprécié, cela-dit le style du code de votre rendu est assez désastreux. L'indentation est particulièrement bancale à peu près partout, essayez de revoir ça d'ici le prochain rendu SVP. Il y a également des fonctions trop longues et des lignes trop longues, ainsi que des magic numbers qui rendent le code d'autant plus difficile à lire (utilisez des enum ou des constexpr). Attention également aux attributs publics qui sont nombreux dans lifeform.h | **3** |

| | | | | |
|---|---|---|---|---|
| **382029** | **4** | [L1] lifeform.cc méthode get_id, simulation.cc: l.39,l.47,l.97 double indentation, l.18 deux espaces aulieu de 4. | Rendu très bien organisé, respect des principes d'abstraction et très bonne modularisation. Attention à vos indentations. | **9** |