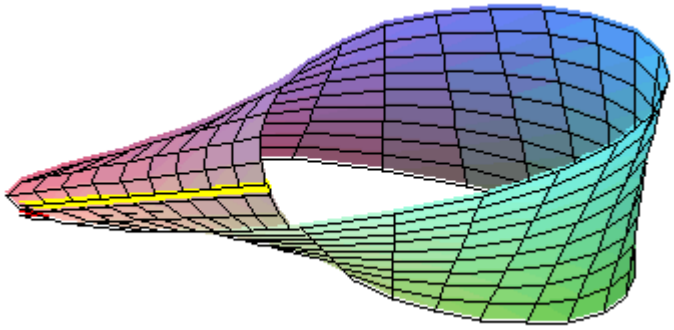
PROJET SDA

Dédale sur un ruban de Möbius à pavage octogonal



Sébastien **CUVELLIER**, Groupe 106
Emma **DEVOS**, Groupe 106

Table des matières:

Présentation du projet

Dans le cadre de notre projet de SDA, nous avons réalisé une application en C++ qui permet à un “Dragon ailé” de se déplacer dans un labyrinthe, sous forme de ruban de Möbius à pavage octogonal. Celle-ci était à faire en 5 différentes étapes qui étaient donc les 5 sprints de notre projet. À chaque sprint, il fallait créer de nouvelles fonctions, de nouveaux fichiers .h ou .cpp, ou tout simplement écrire des lignes de code pour modifier le programme.

Le premier sprint que nous avions à réaliser consistait à démarrer le projet en construisant notre labyrinthe. À la fin de ce sprint nous devions être capables d’afficher le labyrinthe initial de notre programme.

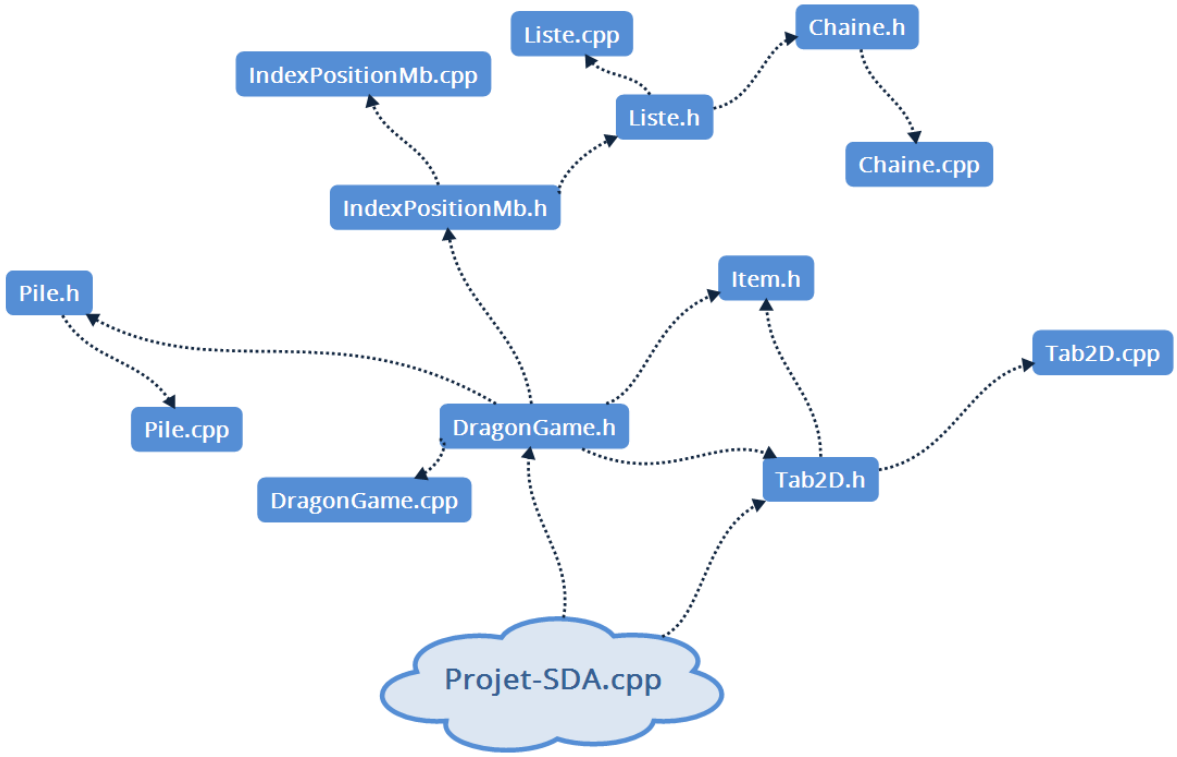
À la suite de l’affichage du labyrinthe, le sprint 2, lui, nous permet d’afficher les 10 premières cases que le dragon a visitée en fonction des données qui sont en entrée du programme. Les positions seront affichées grâce à 3 coordonnées, tout d’abord x qui représente comme dans un repaire l'abscisse, la ligne ou le dragon se trouve, ensuite nous avons y qui représente les ordonnées, donc la colonne, et enfin le dernier chiffre donné nous indique sur quelle face du labyrinthe le dragon se balade.

Le sprint 3 quant à lui, nous a permis d’afficher, en marquant la case de la lettre “V”, toutes les cases du labyrinthe ayant été visitées par le dragon, sur la première mais aussi la deuxième face de notre noeud de Moebius, nécessitant donc le codage de ce changement de face.

Ensuite vient le sprint 4, qui permet d’afficher le chemin connexe du dragon, une fois qu’il a visité plus de 20 cases. C'est-à-dire d’afficher le chemin du dragon à partir de la première case du labyrinthe, qui lui permettra ensuite de revenir au départ après avoir trouvé les Plans du monde. Ce chemin est affiché grâce à la lettre “C” marquée dans chacune de ses cases.

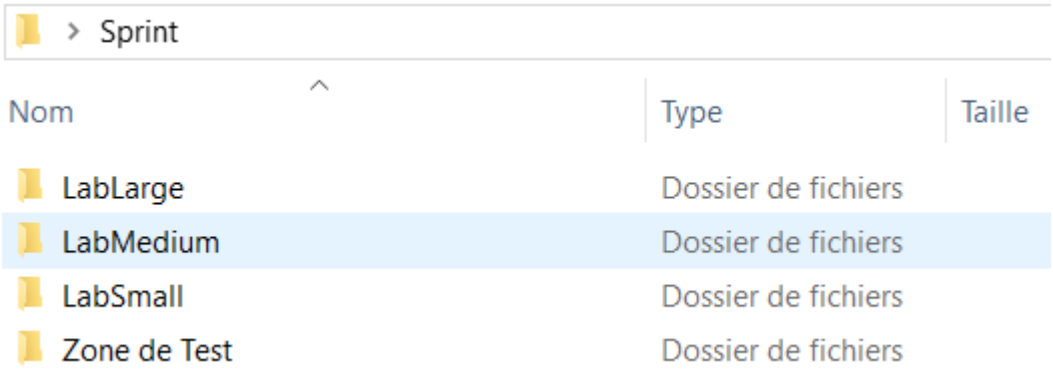
Enfin, nous avons le sprint 5, pour le réaliser, il faut vérifier s'il existe un chemin connexe permettant de relier la position des Plan du monde avec l'entrée du labyrinthe, et s'il y en a un, il faut pouvoir l’afficher. Au contraire, si ce chemin n’existe pas, il faut afficher le mécontentement du Dragon, où toutes les cases du labyrinthe, sauf le “P” du Plan des mondes, affichent un “#”.

Graphique de dépendance fonctionnelles

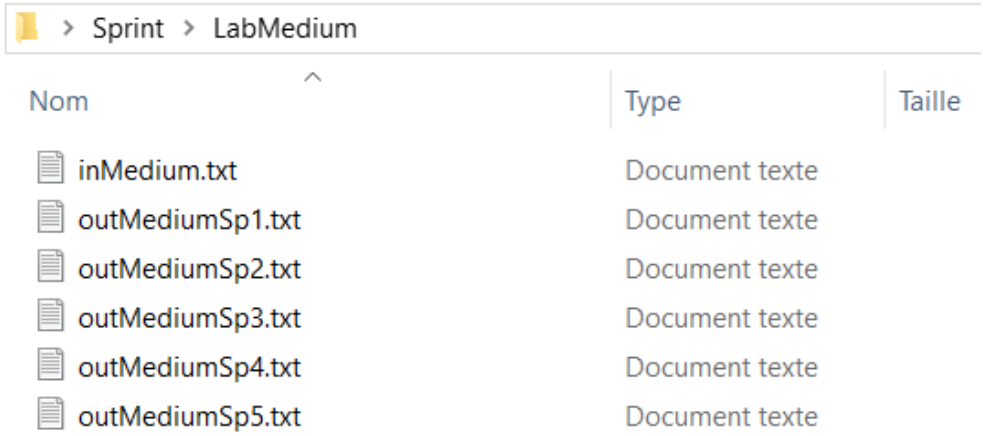


Organisation des tests:

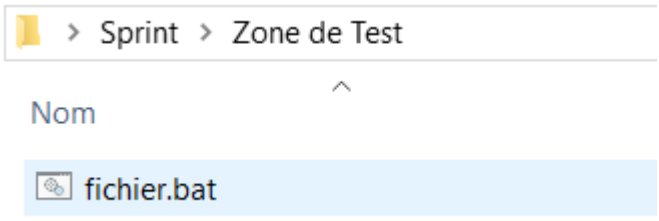
Le projet que nous avons mené a été réalisé sous une forme de développement par sprints. Chaque sprint étant défini par des objectifs précis sous forme de spécification, accompagné d'un fichier avec un jeu de données test (in**Size**Sp.txt) et un fichier avec le résultat attendu après le programme exécuté (out**Size**Sp.txt) .



Dans un premier temps, on réunit tous les fichiers suivants dans le même dossier:



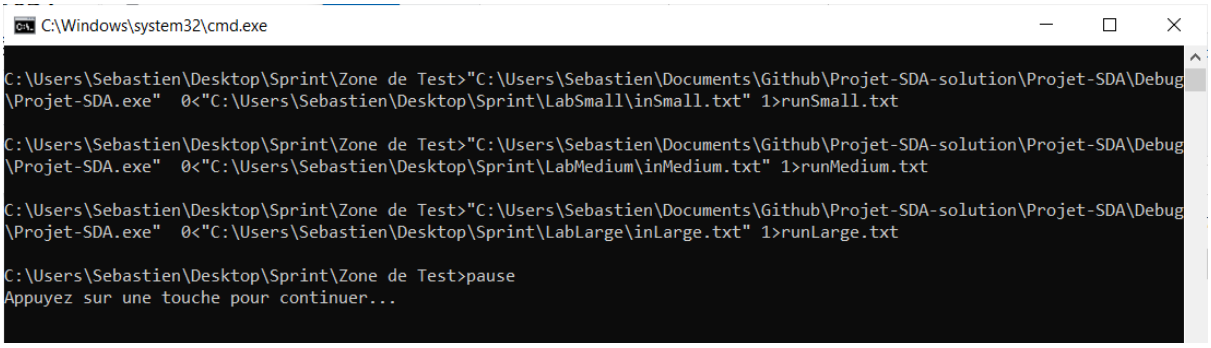
On crée un script batch qui nous permet de générer les différents et on exécute le programme:



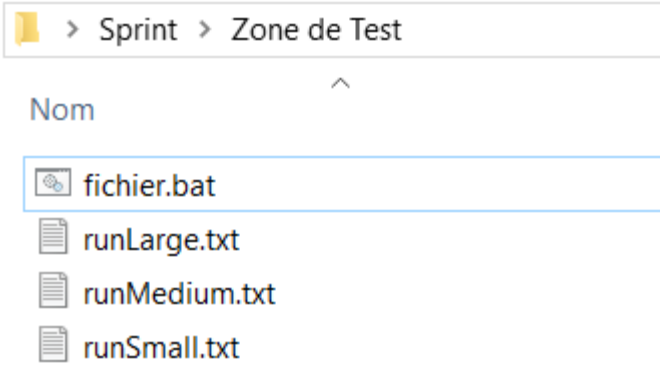
Voici le code de notre script batch:

```
Fichier  Edition  Format  Affichage  Aide
"C:\Users\Sebastien\Documents\Github\Projet-SDA-solution\Projet-SDA\Debug\Projet-SDA.exe" <"C:\Users\Sebastien\Desktop\Sprint\LabSmall\inSmall.txt"> runSmall.txt
"C:\Users\Sebastien\Documents\Github\Projet-SDA-solution\Projet-SDA\Debug\Projet-SDA.exe" <"C:\Users\Sebastien\Desktop\Sprint\LabMedium\inMedium.txt"> runMedium.txt
"C:\Users\Sebastien\Documents\Github\Projet-SDA-solution\Projet-SDA\Debug\Projet-SDA.exe" <"C:\Users\Sebastien\Desktop\Sprint\LabLarge\inLarge.txt"> runLarge.txt
pause
```

On exécute ensuite notre script:



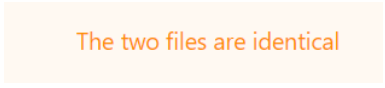
On a alors l'apparition d'un fichier run**Size**.txt qui contient les sorties du programme:



On cherche ensuite à vérifier si notre fichier run.txt correspond au fichier outS4.txt, on utilise alors l'outil en ligne https://www.diffchecker.com/ qui nous a permis de vérifier la correspondance de nos fichiers.

Nous avons eu un cas où il y avait un problème dans la correspondance, avec cet exemple d'un problème de saut de ligne que nous avons oublié:

Une fois le problème corrigé on renvoie les deux fichiers sur le site et si tout se passe comme prévu, on obtient ce message et on passe au sprint suivant.



Validation des tests:

Sprint 1:

Le sprint 1 a été la première phase de notre projet et il n'aura pas été très compliqué, en effet nous avions bien cerné les tableaux à deux dimensions et l'allocation dynamique de mémoire ce qui nous a permis de gagner beaucoup de temps car il nous suffisait d'appliquer nos acquis.

Sprint 2:

Le sprint 2 a été pour nous assez facile car avoir codé le sprint 1 du mieux que nous pouvions nous a permis de simplement avoir à intégrer une pile comme nous avions vu en tp et td, qui nous a ensuite permis de coder facilement l'algorithme de recherche qui ne cachait pas de piège. Nous avons ensuite eu un problèmes avec l'apparition d'un ":" puis nous avons compris que c'était la suite de la table ASCII après le numéro 9, nous avons donc changé l'arrêt de notre boucle et le problème a été résolu.

Sprint 3:

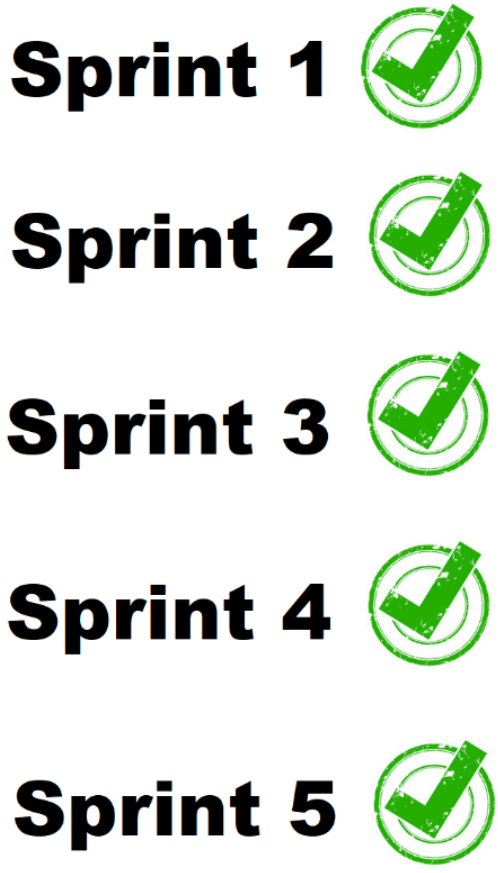
Le sprint 3 a été pour nous l'un des plus compliqués, en effet nous avons eu ici de gros problèmes à comprendre comment fonctionnait le passage d'une face à l'autre ce qui nous a fait perdre beaucoup de temps. Au final nous avons représenté physiquement le ruban de moebius ce qui nous a enfin permis de comprendre les mécaniques de changement de face.

Sprint 4:

Le sprint 4 a aussi été pour nous un des sprints les plus compliqués, en effet le chemin connexe nous a causé de nombreux soucis et le codage des fonctions de la liste n'a pas été très simple malgré le TP et TD. Nous sommes alors retournés sur du papier pour chercher les potentielles erreurs et comprendre que nous n'inscrivons pas les éléments de la liste à la case 0 de la liste, ce qui produisait donc des erreurs et une impossibilité de fonctionnement du sprint. Finalement nous aurons fini par régler les problèmes et valider ce sprint via nos différents test.

Sprint 5:

Le sprint 5 ne nous a pas causé de problèmes particuliers, de plus nous commencions à bien connaître les différentes erreurs, et dès notre sprint 4 nous avions intégré la possibilité de changement de face, nous avons donc agrandit la taille du chemin connexe, et affiché le chemin comme demandé, simplement grâce à une boucle qui ressortait les valeurs de la liste. Le mécontentement du dragon n'a pas été facile mais après réflexion, nous avons trouvé un moyen de l'afficher au bon moment pour ensuite quitter le programme après l'affichage.



Bilan sur le projet:

Le projet nous a permis dans un premier temps de se confronter à un vrai projet, ce qui change complètement des exercices habituels de TP/TD avec la nécessité de collaborer avec son collègue et d'installer un environnement de travail.

Au début, nous avions un simple fichier que nous nous envoyons via un drive après chaque modification mais on a vite compris que ce n'était pas la bonne solution. En effet il nous était impossible de voir les modifications de l'autre et de voir réellement ce qui causait problème si c'était le cas, on y a perdu beaucoup de temps pour au final découvrir GitHub qui nous a permis de gagner du temps les dernières semaines car nous avions accumulé un certain retard. GitHub nous a permis de travailler plus facilement à deux, de se fixer des objectifs et d'avoir une communication claire dans nos modifications, et le système de branche nous aura été très utile.

Nous avons également perdu de précieuses heures à se perdre sur la documentation de Microsoft sur visual studio et le C++ afin de tenter de lever les erreurs. Erreurs qui globalement avaient été causées par des erreurs qui nous paraissent maintenant "stupides".

Nous avons également eu beaucoup de mal à démarrer suite aux problèmes précédents car nous avions rarement la possibilité de se retrouver physiquement autour du projet, à cause des grèves et en vue de nos horaires et obligations bien différentes. Nous passions donc du temps à essayer de se comprendre au téléphone ce qui nous ralentissait clairement. Nous avons également perdu du temps sur le premier sprint car nous avions peur de rater un détail et de finalement se retrouver bloqués plus tard, nous avons donc pris le temps de vérifier chaque petit détail, essayer les éventuels pièges que nous redoutions, par exemple des caractères non prévus afin d'éviter les mauvaises surprises lors de la validation des sprints.

Nous avons donc beaucoup appris de ce projet collaboratif, en effet il nous a permis de découvrir et d'approfondir notre connaissance du langage C++, ses possibilités et également ses aspects difficiles à appréhender. Le projet nous a également permis d'améliorer notre capacité de travail en groupe dans l'informatique et nous a poussé à trouver comment mieux collaborer dans le développement du projet aussi bien physiquement mais également à distance via de nouveaux outils informatiques. Il nous a permis d'apprendre à suivre des directives de génie logiciel, à appliquer ce qu'on attend de nous tout en cherchant à tirer le maximum de potentiel de notre environnement de développement pour maximiser l'optimisation de notre travail.

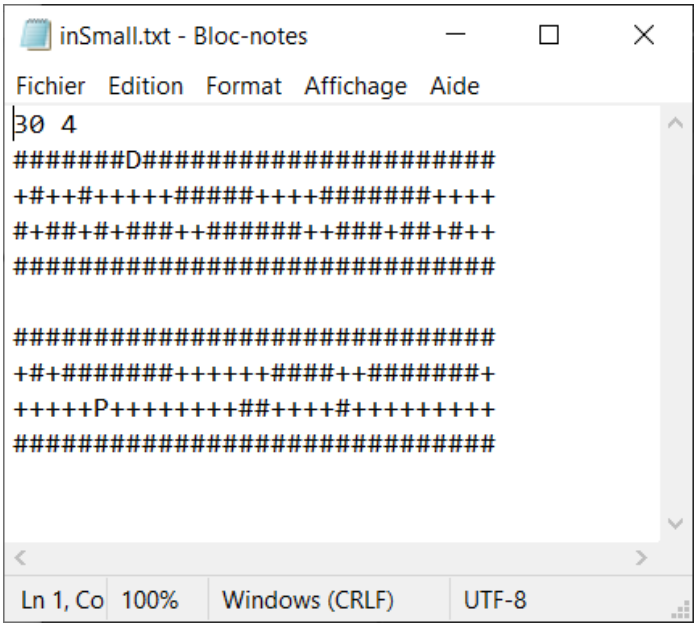
Durant le projet nous avons été confrontés à de nombreux problèmes énumérés en partie précédemment mais nous avons persévéré du mieux que nous pouvions et nous avons finalement réussi à atteindre les 5 sprints. Nous avons également dû faire face aux imprévus, et au temps qui est finalement passé assez vite.

Pour finir, l'aspect collaboratif du projet aura été très enrichissant, en effet nous avons beaucoup échangé et beaucoup appris l'un de l'autre. La possibilité d'avoir un oeil extérieur sur notre code, grâce au groupe nous a permis d'apprendre à clarifier ce qu'on écrivait car tout cela devait être compréhensible par le binôme mais également par les personnes susceptible de modifier dans le futur notre code.

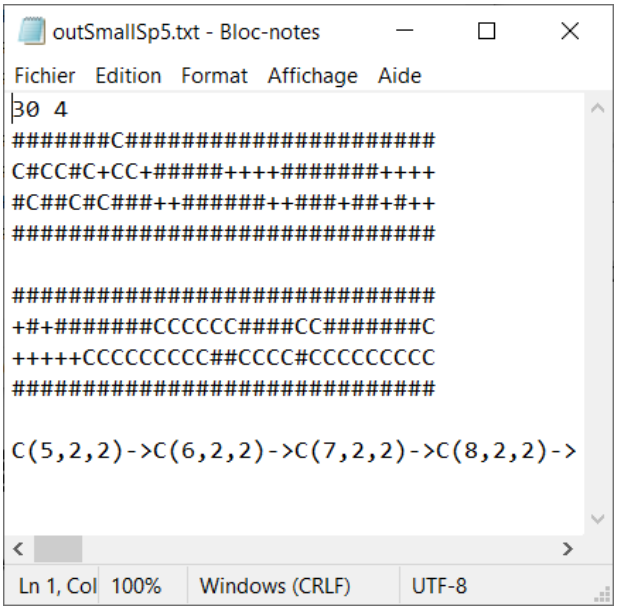
Annexe 1:

Jeux de test et sortie:

Jeux de test du sprint 5 (inSmallSp5.txt):

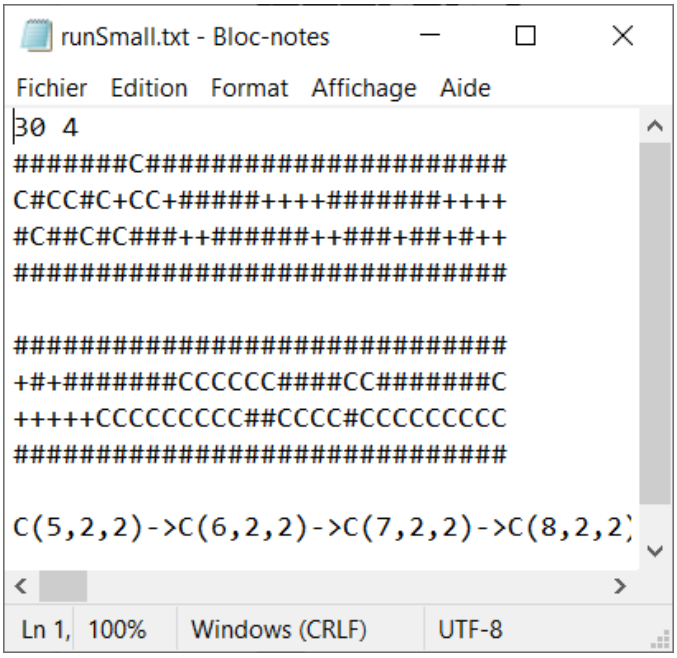


Fichier avec le résultat attendu pour le sprint 5 (outSmallSp5.txt):



Annexe 1:

Sortie de notre programme:



Annexe 2:

Listing des sources:

Ensemble d'informations ayant contribué à la réussite de notre projet

- Utilisation massive du cours en C++ vu en amphithéâtre.

- Utilisation de certaines mécaniques vues en TP et en TD, pour par exemple les piles et les listes.

- Utilisation de la documentation officielle de Visual Studio en C++ par Microsoft pour résoudre de nombreux codes erreur.

- Utilisation du site devdocs.io/cpp/ pour les informations sur les fonctions intégrées au langage C++

Annexe 3: