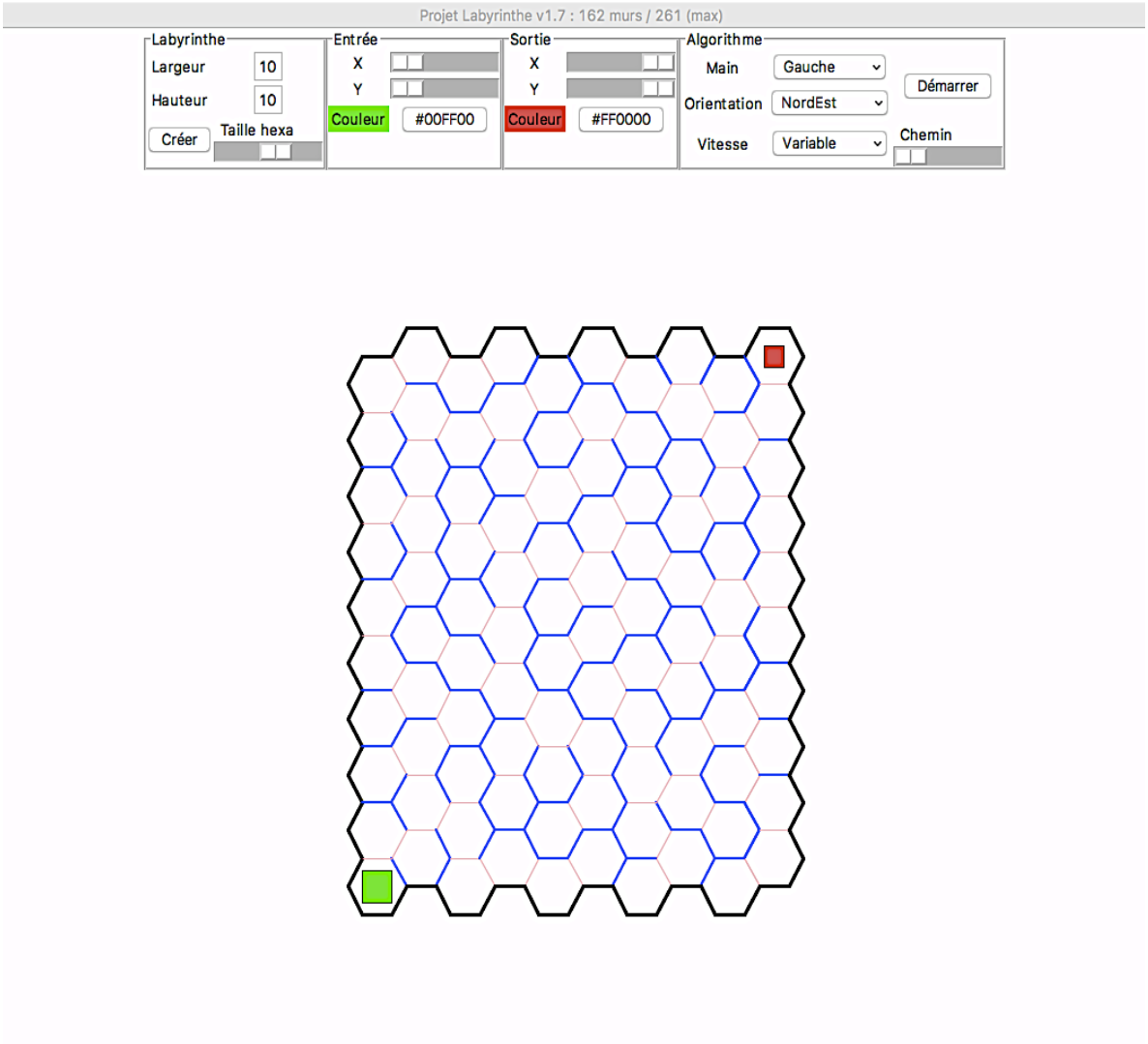
TERMINALE S ISN
Lucas Schneider
Maxence Decourtis
Maé de Montalier

PROJET ISN : LABYROBOT



I – PRESENTATION DU PROJET

Notre équipe LabyRobot s'est lancée dans la réalisation d'un projet en apparence basique mais qui en réalité demande de nombreuses connaissances. Nous sommes fiers de vous présenter notre projet final : un générateur automatique de labyrinthe parfait composé d'îlots avec un robot capable de trouver à chaque fois la sortie.

Pourquoi ce projet ?

Tout d'abord, nous avons réfléchi ensemble à propos de notre futur projet. Assez rapidement, l'idée de la programmation d'un labyrinthe généré aléatoirement et automatiquement avec un robot capable de la résoudre nous est apparue comme la meilleure idée. Il nous a fallu alors peu de temps pour déterminer ce projet et ainsi, nous lancer dans sa réalisation.

Nous avons choisi ce projet pour de multiples raisons. Premièrement, étant tous les trois des étudiants en Terminale Scientifique, les algorithmes sont des suites finis d'instructions avec lesquels nous avons déjà travaillé que ce soit en mathématique ou hors cadre scolaire. Il se trouve que la programmation du projet met en jeu plusieurs algorithmes. Connaissant alors déjà un peu le fonctionnement de ces derniers, nous avons désiré en savoir plus pour pouvoir les mettre en pratique dans notre projet. La curiosité et l'excitation d'apprendre de nouvelles choses nous ont alors poussé à nous lancer dans ce travail.

Dans un second temps, nous avions auparavant lu sur internet des articles mentionnant ce type de projet. L'idée de base nous semblait intéressante mais nous avons vraiment choisi de créer ce programme car nous voulions apporter quelque chose de plus. Nous avons alors réalisé un générateur de labyrinthe qui comporte des hexagones et non des simples cases carrées. Cependant, lors des premières recherches sur le code nécessaire à la création de ce labyrinthe, nous nous sommes rendus compte que le programme était plus dur que dans un labyrinthe classique. Mais attiré par la difficulté et l'envie de se surpasser, nous nous sommes acharnés pour que notre idée puisse devenir réalité.

Dans un dernier temps, ce projet nous parut très intéressant à réaliser car il allait nous octroyer de nombreuses connaissances de programmation qui pourront être utilisées à nouveau dans notre future carrière professionnelle. La quantité de travail que nécessitait ce code ne nous a pas effrayé car nous étions très déterminés et le résultat final en valait le coup. C'était alors pour nous un vrai apport autant personnel que professionnel car nous savions que le projet n'allait pas être simple et qu'ainsi, la cohésion au sein de notre groupe ne pouvait être que renforcé. L'aspect répartition des tâches et travail de groupe nous a beaucoup plu et à travers ce travail, nous savions qu'ils seraient essentiels.

Ainsi, les enjeux du projet étaient multiples :

- Trouver l'algorithme de résolution le plus simple et le plus rapide pour le robot.
- Comprendre et utiliser de nouvelles fonctions de programmation.
- Obtenir un "jeu" facile à prendre en main par l'utilisateur au travers d'une interface clair.
- Générer un labyrinthe aléatoirement à chaque fois
- Se répartir équitablement les tâches et travailler ensemble

1. Positionnement du projet par rapport à des solutions existentielles

Notre projet étant un labyrinthe parfait hexagonal automatiquement généré avec un avec un algorithme de résolution, nous avons recherché si des projets tels que le nôtre avait déjà été réalisé auparavant. Après quelques recherches nous avons surtout trouvé des labyrinthes carrés parfaits générés automatiquement, en effet il existe de nombreux algorithmes permettant la génération de labyrinthes parfaits, et de nombreux algorithmes de résolution. Cependant nous n'avons pas trouvé nous n'avons pas trouvé de tels algorithmes pour des labyrinthes à bases hexagonales.

2. Cahier des charges de l'équipe

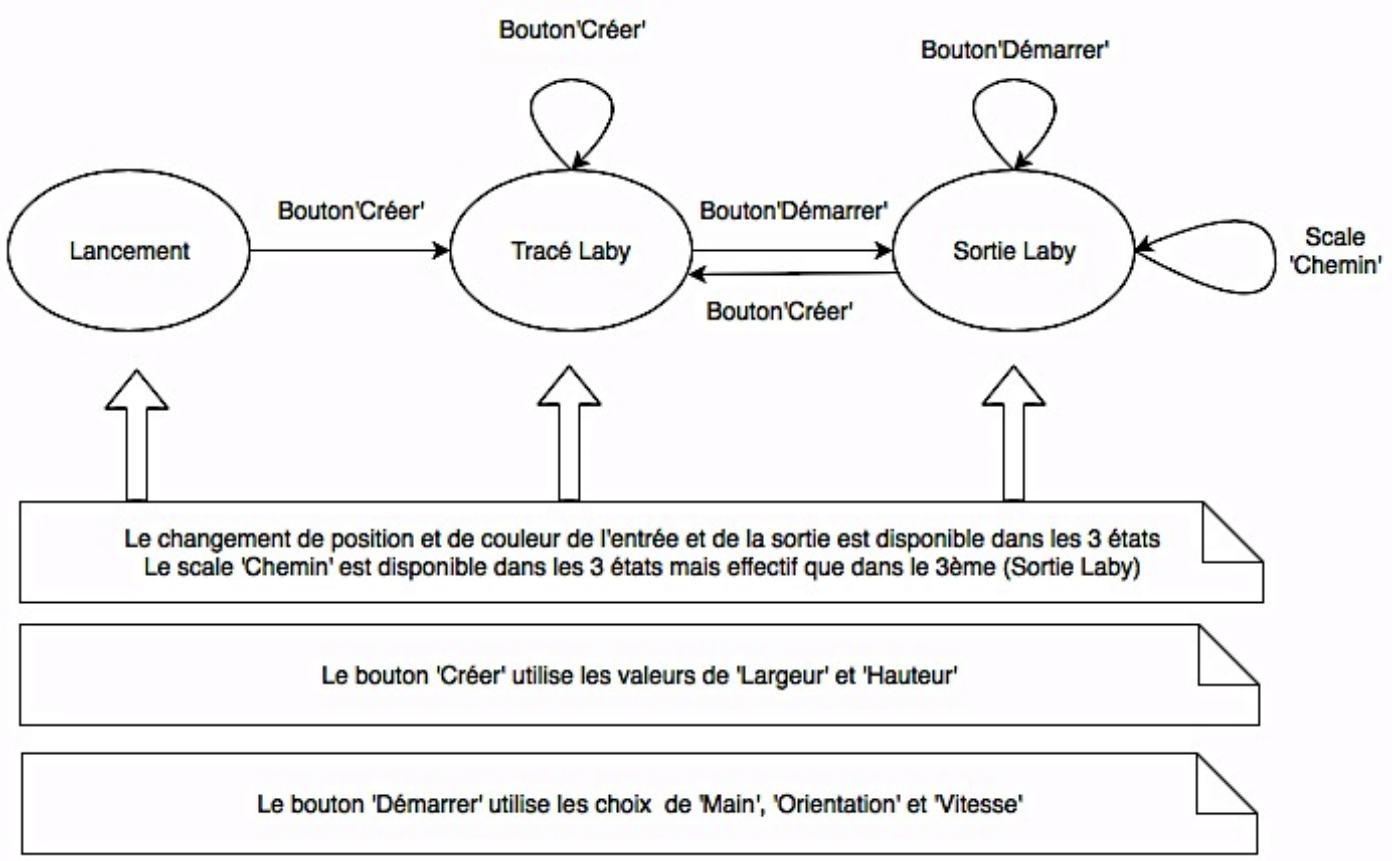
- génération automatique de labyrinthe / parfait
- cases hexagonales
- taille configurable
- définition de la position entrée et sortie
- interface utilisateur conviviale (graphique)
- pouvoir rejouer le tracé
- visualisation du déplacement (tracé pas pratique ca marche avec les numéros)
- algorithme de suivi de mur (deux options main gauche et main droite)

3. Moyens mis en œuvre (langage / matériel)

Notre programme nécessite l'utilisation de Turtle. En effet, ce module graphique permet de, comme son nom l'indique, déplacer une tortue sur un écran. Ce dernier fonctionne avec le langage de programmation Python. On a alors pu utiliser certaines de nos connaissances sur la langage pour les appliquer dans le code. De plus, nous avons eu recours à TKinter pour la création de l'interface du labyrinthe. Encore une fois, c'est le langage Python qui est utilisé car c'est un module de base intégré dans Python. Ainsi, la totalité de notre code repose sur le langage Python.

4. Structure globale du projet

Il y a 3 états dans lesquels l'utilisateur peut réaliser des actions différentes : l'état 'Lancement' obtenu au lancement du programme, l'état 'Tracé Laby' et l'état 'Sortie Laby' :



5. Tableau présentant les tâches et les avancées de notre équipe

	AVANCÉES et TÂCHES		
	LUCAS	MAXENCE	MAE
12/01/18	Choix sujet : labyrinthe parfait généré automatiquement avec un robot le qui le résout par lui même. Faire des recherches sur labyrinthes car sujet complexe et ambitieux.		
19/01/18	Idée d'un labyrinthe avec cellules en diagonales pour originalités. Trouver un programme pour l'interface graphique.		

<table>
<tr><td></td><td colspan="3"></td></tr>
<tr><td>26/01/18</td><td colspan="3">Il faut s'occuper de :
- L'interface graphique
- Comment générer un labyrinthe ?
- Quels outils informatiques utiliser ?
- Modélisation du robot (affichage son chemin)
- Quelle case d'entrée ? Quelle case de sortie ?
- Algorithme du robot → aléatoire ou plusieurs mode de sortie

Test pour algorithme du robot : peut-il sortir en se collant au coté gauche ou au côté droit ? → vrai à gauche et vrai à droite</td></tr>
<tr><td rowspan="2">02/02/18</td><td colspan="3">Test des labyrinthes parfait et imparfait.
Mise en commun des premières lignes de programmation et des résultats des tâches → Utilisation du logiciel Turtle pour la génération du labyrinthe, et de TK inter pour l'interface graphique.
Répartition globale des tâches :</td></tr>
<tr><td>Trouver comment générer le labyrinthe avec Turtle.</td><td>Trouver les ressources nécessaires pour la réalisation du projet.</td><td>Comprendre les techniques de TK inter pour l'interface.</td></tr>
<tr><td>09/02/18</td><td colspan="3">Cours annulé, Skype pour parler des avancements et des tâches à faire.</td></tr>
<tr><td>16/02/18</td><td>Algorithme du labyrinthe terminé par Lucas.
Et mise en forme de programme.</td><td>Avancer sur le déplacement du robot.
Savoir ce qu'il y manque dans le programme.</td><td>Début de l'interface graphique fait.</td></tr>
<tr><td rowspan="2">Pendant les vacances de février</td><td>Codage de la partie génération du labyrinthe en hexagone avec l'initialisation de Turtle</td><td colspan="2">Trouver les idées et faire des recherches pour le code de la génération du labyrinthe → idées ensemble, Lucas lui qui codait le plus + petite aide de son père, ingénieur en informatique.</td></tr>
<tr><td colspan="3">On s'est vus mardi, vendredi et samedi de la première semaine, Maé en Skype. Traçage de la grille d'hexagone centrée sur la fenêtre Turtle.
Réglages de quelques problèmes lorsque le labyrinthe contenait un nombre pair ou impair d'hexagones en largeur au niveau des murs extérieurs.
Aide avec fonction help.turtle, → beaucoup de nouvelles idées.</td></tr>
<tr><td>09/03/18</td><td>Commencer l'algorithme du robot avec l'algorithme de Pledge : sortir du labyrinthe en tournant tout le temps à gauche ou tout le temps à droite.</td><td>Faire les finitions du codage du programme du labyrinthe.</td><td>Continuer de coder l'interface graphique avec Lucas.</td></tr>
</table>

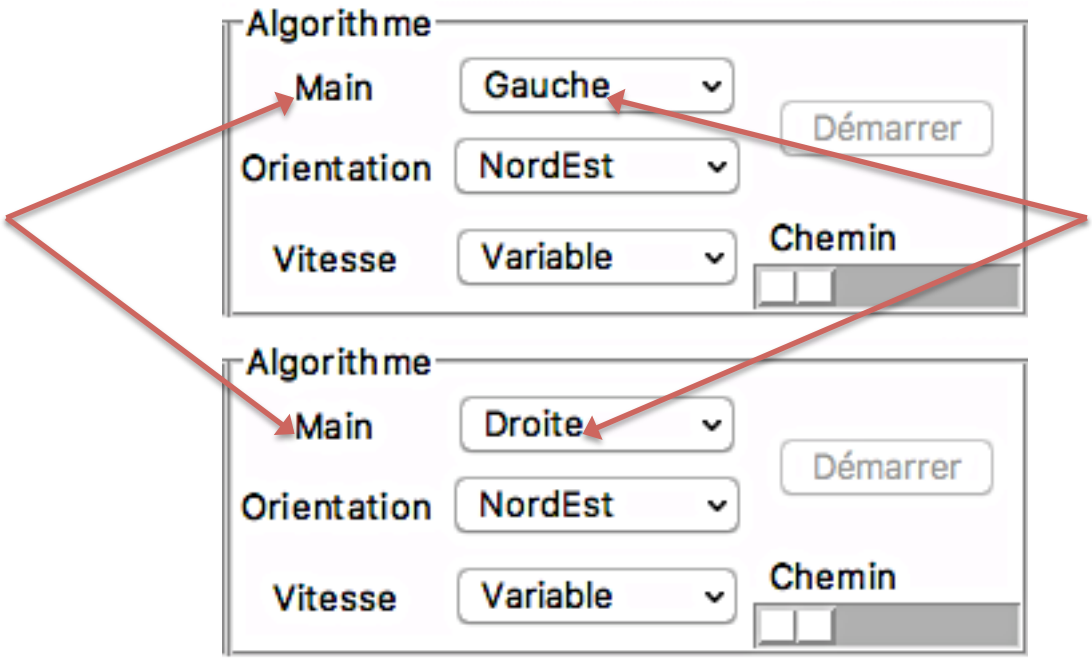
<table>
<tr><td>16/03/18</td><td colspan="2">Début programmation de l'algorithme de Pledge.
Continuer programmation génération automatique du labyrinthe.
Continuer à avancer maintenant tous ensemble sur l'interface graphique.</td></tr>
<tr><td rowspan="2">23/03/18</td><td colspan="2">Pas cours d'ISN, réunion chez Maxence.
Abandon de l'algorithme de Pledge trop compliqué, autre algorithme car un seul type de labyrinthe : le labyrinthe parfait.</td></tr>
<tr><td>Utilisation et programmation d'un nouvel algorithme « suivre le mur », le robot va seulement suivre soit le mur de gauche ou soit le mur de droite plus simple à coder.
Nouvelle idée → slider : suivre résolution laby</td><td>Avancée au niveau de l'interface graphique (la doc de TK inter sur internet très pratique)</td></tr>
<tr><td>30-31/03/18</td><td colspan="2">Pas cours d'ISN, réunion chez Lucas pendant le week-end pour réalisation de l'interface graphique. Petite aide du père de Lucas l'organiser et comprendre les fonctionnalités de chaque fonction.</td></tr>
<tr><td rowspan="2">06/04/18</td><td colspan="2">Mise au point avec Mr Jules, → il faut avancer vite.
Idée d'un fichier audio.</td></tr>
<tr><td>Réglage d'un défaut.
Avancer codage algorithme résolution.
Interface graphique.</td><td>Grande avancée dans l'interface graphique de la page du labyrinthe, nouvelles idées et nouvelles mises en formes.</td></tr>
<tr><td>13/04/18</td><td colspan="2">Mise au point des avancées de la semaine.
Objectif : terminer code pour rentrée pour finaliser petits détails.</td></tr>
<tr><td rowspan="2">Pendant les vacances de Pâques</td><td colspan="2">Finir l'interface et l'algorithme de programmation du robot.</td></tr>
<tr><td>Programmer l'algorithme de gauche et droite.
Test vérification de l'algorithme « suivi du mur.
Faille : robot bloqué en main gauche.</td><td>Programmer slider pour chemin robot (aide de Lucas) + option pour l'utilisateur : dessiner parois labyrinthe avec couleur.
Programmer boutons slider et boutons main droite et main gauche.</td></tr>
<tr><td>04/05/18</td><td colspan="2">Retour des vacances → quasiment terminé juste détails. Accords sur modifs à faire pour finaliser. Répartition tâches pour rédaction dossier.
Changer nom certaines fonctions car pas explicites. Ajout d'une fonction pour changer vitesse tracé chemin du robot.</td></tr>
</table>

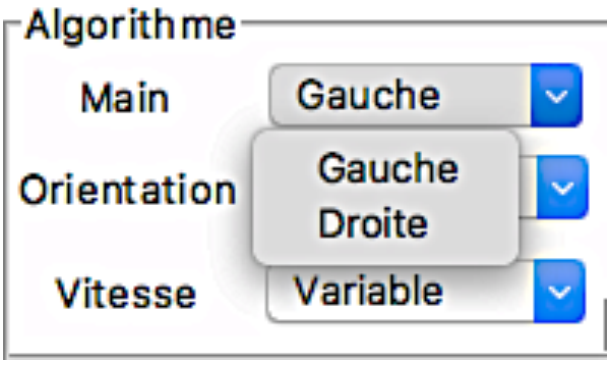
II – REALISATION PERSONELLE

Pour ce projet j'ai surtout travaillé sur l'interface graphique. Il a d'abord fallu que je comprenne comment coder une interface graphique, que je m'approprie les différentes fonctions de TK inter. Cependant c'était assez compliqué de tout comprendre, il m'a fallu du temps et heureusement que Lucas et Maxence ont aussi travaillé dessus, ça nous a permit de comprendre des fonctions qu'un de nous comprenait et pouvait nous expliquer. Pour pouvoir comprendre les différentes options que nous présentait TK inter nous nous sommes aidés d'internet où l'on y trouvait une grande documentation dessus, ce qui fut très utile.

Voici une petite partie de l'interface que j'ai codé :

```
# Main
Label(algoFrame, text='Main').grid(row=0, column=0)
varMain = StringVar(Wroot)
varMain.set(MAIN[0])
OptionMenu(algoFrame, varMain, *MAIN, command=mainCmd).grid(row=0, column=1)
```





Cette partie de l'interface graphique associe le choix de la main de l'algorithme du robot à la fonction qui permet le changement de la main et donc à l'algorithme de résolution.

« Label » → permet d'insérer un texte dans une fenêtre et ici dans la fenêtre algoFrame.

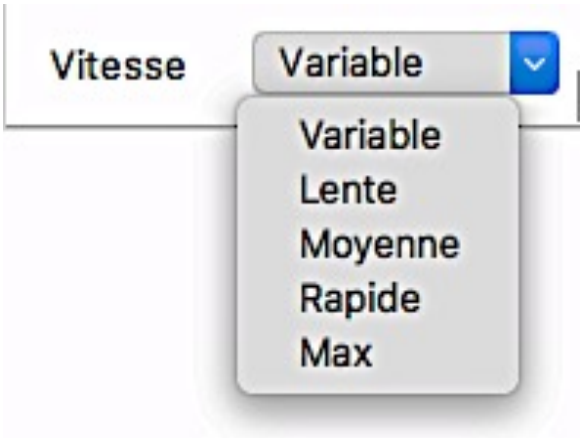
« .grid() » → va permettre de positionner, placer ce texte en fonction des colonnes et lignes de cette fenêtre.

« StringVar(Wroot) »→ est une variable de contrôle qui va mémoriser une chaîne de caractère et sa valeur par défaut va être « Wroot », étant la fenêtre principale de notre projet, ainsi « varMain » est une variable de valeur Wroot. Elle est utilisée avec la fonction « .set() » qui va permettre de modifier la valeur courante de notre variable et si les options d'un ou plusieurs widgets sont reliées à cette variable, ces widgets seront automatiquement mis à jour quand la boucle principale sera à nouveau en attente. « MAIN[0] » est ainsi la valeur par défaut de la variable.

De plus cette variable est associée à un bouton menu : « OptionMenu() » qui va ainsi permettre en cliquant sur le bouton de choisir entre le main gauche et la main droite. Ainsi l'utilisation d'une fonction StringVar() et sa méthode .set() c'est pour être capable de modifier le texte affiché sur un bouton de menu.
On place donc le bouton menu dans algoFrame, on y associe la variable varMain, le tableau MAIN et enfin la commande « mainCmd » qui est la fonction permettant le changement de la main. Puis le bouton menu est placé avec la fonction « .grid ».

La difficulté qui s'est beaucoup présentée à nous lors du codage de notre interface fut d'associer les fonctions aux boutons de commande. C'est-à-dire qu'il était compliqué de faire en sorte que lorsqu'on clique sur un bouton ou que l'on choisisse une option, cela actionne bien correctement la commande demandée. Ça nous a notamment posé un petit problème pour le choix de la vitesse. En effet avec Lucas nous avons travaillé pendant tout un cours sur la vitesse du tracé du chemin de résolution et le bouton menu pour le choix de la vitesse. Nous avons eu beaucoup de mal à faire en sorte que la vitesse change en fonction de l'option choisi mais nous avons finalement réussi et Lucas l'a même amélioré chez lui.

```
# Vitesse
Label(algoFrame, text='Vitesse').grid(row=2, column=0)
VarVitesse = StringVar(Wroot)
VarVitesse.set(VITESSE_TXT[VITESSE_DEF])
OptionMenu(algoFrame, VarVitesse, *VITESSE_TXT, command=vitesseCmd).grid(row=2, column=1)
```

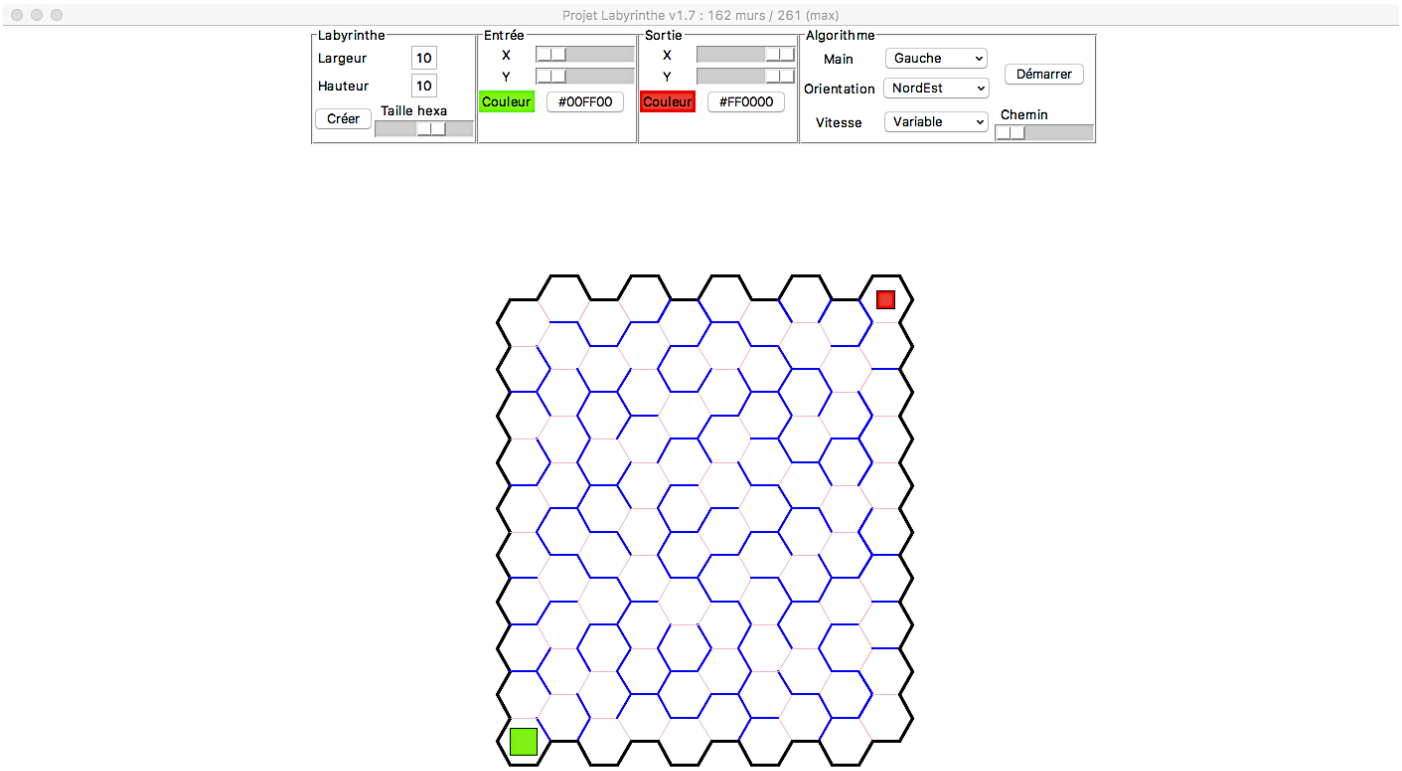


On peut cependant observer que le bouton menu vitesse est codé de la même manière que le bouton menu main. En effet ce qui fut compliqué était de se servir de la fonction de turtle, turtle.speed(), d'une certain manière pour que l'interface et cette fonction puissent correspondre.

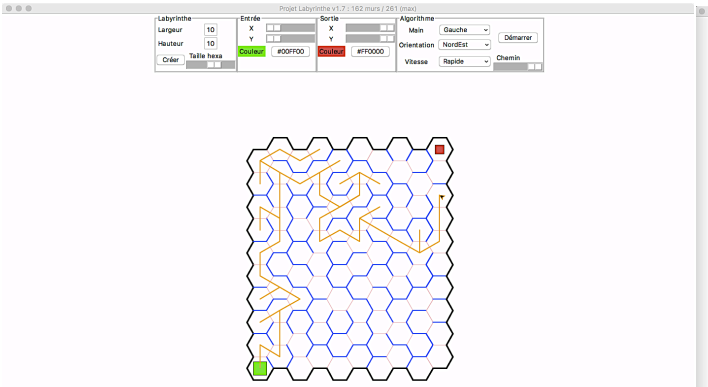
Ainsi il m'a fallu fournir beaucoup de travail pour comprendre comment coder l'interface qui nous a pris du temps et nous avons tout de même obtenu ce que nous voulions. L'interface est lisible, précise, claire pour tout le monde et offre beaucoup d'options aux utilisateurs.

III – Intégration et Validation

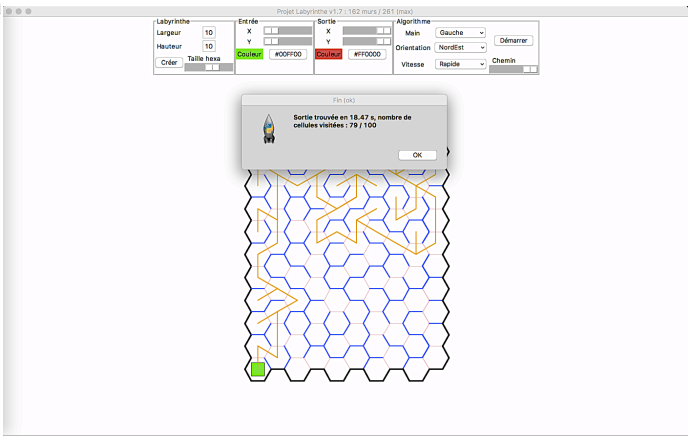
Voici donc notre réalisation finale :



1 – Génération du labyrinthe



2 – Résolution du labyrinthe



3 – Fin de la résolution

4 – Affichage de la durée de la résolution et du nombre de cellules visitées.

On peut voir ici toute l'interface réalisée avec toutes les options disponibles, la génération du labyrinthe ainsi que la résolution de ce labyrinthe. On peut y voir les différents boutons menus que j'ai réalisés et d'autres.

On remarque notamment que notre projet correspond entièrement au cahier de charges, en effet on retrouve le labyrinthe généré automatiquement à base hexagonale. On peut modifier la taille du labyrinthe ou même la taille des hexagones, on peut aussi définir l'entrée et la sortie et on a notre algorithme de résolution qui fonctionne. Ainsi on peut visualiser la résolution par un tracé et rejouer ce même tracé. Nous avons aussi réussi à obtenir, configurer une interface lisible et compréhensible ainsi qu'explique.

De plus nous avons fait plus de choses que nous avions mis dans le cahier de charges, en effet certaines options n'étaient pas prévues telles que la différence de vitesse pour le tracé ou même le choix de l'orientation. Toutes ces idées ont été permises grâce à un travail de groupe collaboratif et efficace.

Pour parvenir à la réalisation de notre projet nous avons en effet du réaliser plusieurs test et nous avons eu plusieurs versions du code.
Nous avons d'abord réaliser plusieurs test à l'écrit sans utiliser d'ordinateur pour voir les possibilités de résolution du labyrinthe ce qui nous a permis de choisir l'algorithme de résolution « suivre le mur de gauche/droite », mais aussi un test de labyrinthe parfait et imparfaits en base hexagonale qui nous a décidé à ne pencher notre projet seulement sur des labyrinthes parfaits.
De plus grâce à la plateforme collaborative github, nous avons pu mettre en commun à chacun d'entre nous chaque version différentes du code ce qui nous permettait d'avoir chacun un œil, une perspective différente et ainsi de relever les différents problèmes.

IV – Bilan et Perspectives

Pour améliorer ce projet, je pense que l'interface graphique pourrait être plus jolie même si elle est très nette et précise de plus je pense que l'on pourrait faire une page de présentation expliquant ce projet et comment utiliser.

Enfin, ce projet m'a beaucoup apporté, en effet j'ai appris beaucoup de choses sur le codage et surtout le codage d'une interface graphique cependant même si je n'ai pas beaucoup participé dans l'algorithme de génération du labyrinthe et dans l'algorithme de résolution, j'ai tout de même tout bien compris et cela m'a aussi permis d'apprendre dans ce domaine.

De plus le travail de groupe fut très bénéfique, en effet il a permis de faire avancer vite notre projet grâce à un travail sérieux et une super collaboration, en effet même si j'avais plus de mal à comprendre le codage j'essayais de comprendre et je fournissais des idées très utiles et je pouvais même finalement résoudre, régler des problèmes que d'autres n'arrivait pas à faire. En effet avoir des perspectives différentes sur un même projet a permis au projet de grandir et de se développer par un flux d'idées différentes et beaucoup de travail fourni.

Maé de Montalier