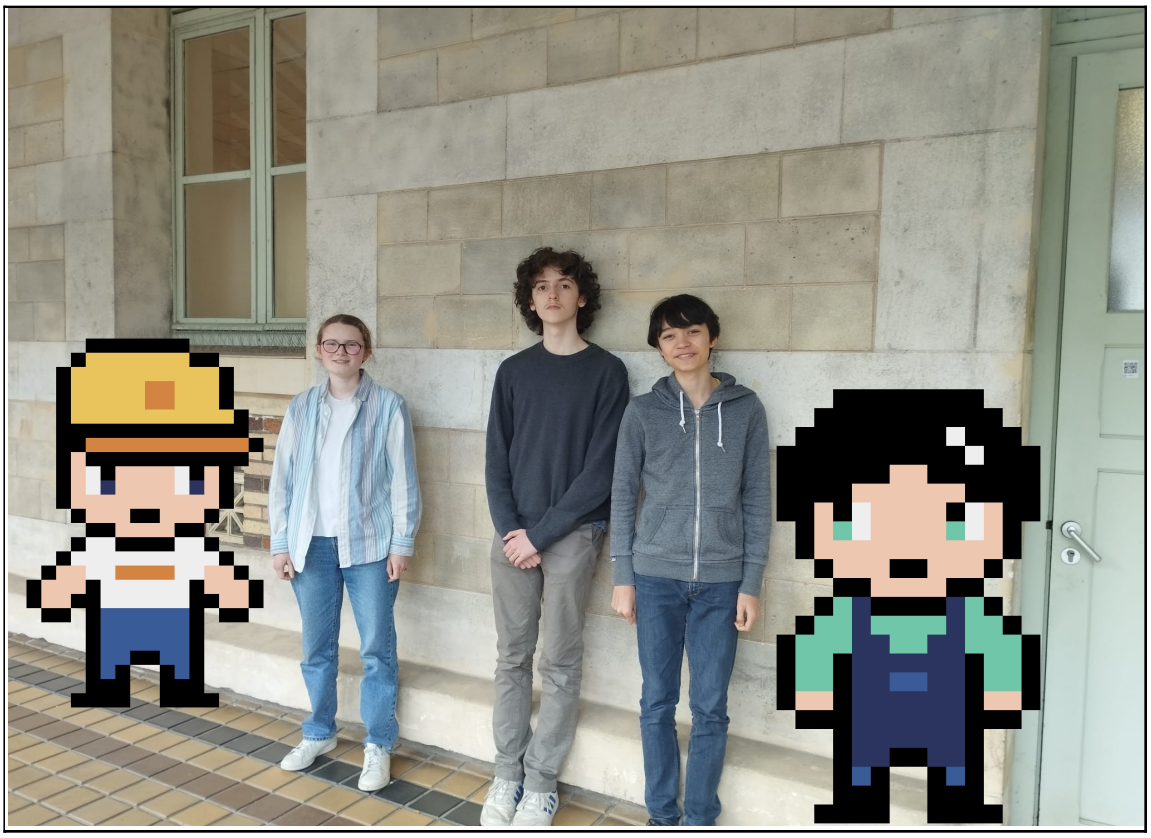
LES TROPHÉES NSI

Édition 2024

DOSSIER DE CANDIDATURE

PRÉSENTATION DU PROJET



nom de votre projet :	Amaze’d
membres de l’équipe :	Aristide Even-Enquin
membres de l’équipe :	Kim Podeur
membres de l’équipe :	Ukyo Nguyen
membres de l’équipe :	Prune Rousseau
membres de l’équipe :	Olivier Daoud
niveau d’étude :	Première générale
établissement scolaire :	Lycée Louis-le-Grand
enseignante/enseignant de NSI :	Loïc Josse

> PRÉSENTATION GÉNÉRALE :

Pouvez-vous présenter en quelques mots votre projet ?
Comment est né ce projet ? Quelle était la problématique de départ ?
Quels sont les objectifs ? À quels besoins répondez-vous ?

Notre projet Amaze'd est un projet consistant à donner à l'utilisateur une interface de création de labyrinthe. L'utilisateur peut créer un labyrinthe, le tester, voir si celui-ci est résoluble, le partager, l'enregistrer… Pour rendre les labyrinthes plus ludiques et plus complexes, nous y avons rajouté une mécanique de portails, permettant de se téléporter d'un endroit du labyrinthe à un autre. On a également essayé de rendre l'expérience utilisateur la meilleure possible, en implémentant des graphismes, des animations et des musiques faites de telles sorte que le labyrinthe ressemble à un jeu d'aventure, il sera ainsi plus agréable aux utilisateurs de résoudre leurs labyrinthes ou ceux de leurs amis.

Quand nous avons décidé de participer au projet NSI, nous nous sommes dit que nous voulions coder un projet interactif et amusant, afin de pouvoir en profiter plus tard. Nous voulions aussi pouvoir travailler plusieurs domaines en même temps (l'algorithmique, l'affichage, l'interaction homme-machine…), et c'est là que l'idée nous est venue. Un jeu, interactif, ludique, qui possède un côté algorithmique : Le labyrinthe nous est tout de suite venu à l'esprit. On a au fil du projet affiné les fonctionnalités (portails, partage, sauvegarde…), mais c'est comme ça que le projet nous est venu.

On veut donc coder un projet durable, qui pourra toujours être complexifié et amélioré, amusant pour l'utilisateur, et polyvalent dans le code. Notre projet reste cependant un jeu, on veut donc avant tout faire plaisir à l'utilisateur. En tant que jeu, notre projet répond bien sûr à des problématiques liés à l'ennui, au fait de joindre le ludique à l'instructif, car réfléchir pour créer et résoudre un labyrinthe demande quand même une certaine réflexion.

> ORGANISATION DU TRAVAIL :

Pouvez-vous présenter chaque membre de l'équipe et préciser son rôle dans ce projet ?
Comment avez-vous réparti les tâches et pourquoi ?
Combien de temps avez-vous passé sur le projet ? Avez-vous travaillé en dehors de l'établissement scolaire ?
Quels sont les outils et/ou les logiciels utilisés pour la communication et le partage du code ?
Vous veillerez au bon équilibre des différentes tâches dans le groupe. Chaque membre de l'équipe doit impérativement réaliser un aspect technique du projet (hors design, gestion de projet).

Olivier est le codeur de l'éditeur de labyrinthe, et du mode de jeu. Il a une expérience extrascolaire en codage importante, et un des créateurs et sûrement la personne que ce projet a motivé le plus tôt, c'est pourquoi il s'est occupé du prototype du projet et a donc codé l'interface de création et de jeu. Il a codé la création de labyrinthe, ainsi que le mode de jeu. Il a par la suite aidé les autres candidats dans leur tâche, surtout lors des tâches les plus complexes.

Ukyo a codé l'algorithme de résolution (et l'a implémenté), et a de plus implémenté les portails, avec l'aide d'Olivier. Cette décision nous a paru relativement logique étant donné qu'Ukyo est de loin le plus doué en algorithmie et en mathématiques au sein de notre équipe, et donc le plus apte à optimiser les problèmes combinatoires.

Prune a codé le reste de l'interface de jeu (menu, différents boutons permettant de switcher entre les différents modes) et s'est occupée avec Kim de la bibliothèque d'images. Prune est sans doute (avec Olivier) la personne ayant passé le plus de temps sur ce projet, elle a pris en charge une partie considérable du travail artistique, ainsi qu'un long bout du code, montrant son investissement au sein du projet. De plus, Prune est de loin la personne ayant trouvé le plus de bugs parmi nous 5.

Aristide s'est occupé du partage et de la sauvegarde, c'est-à-dire de la création et de la lecture de fichiers représentant des labyrinthes, ainsi que de l'implémentation des labyrinthes de base au jeu. De plus, étant assez polyvalent, il s'est également occupé de la gestion et de l'assemblage des différents codes effectués par ses camarades (nous travaillions la plupart du temps en même temps, il était donc difficile de faire autrement que de travailler chacun sur son ordinateur et puis d'assembler après-coup). Il a de plus contribué à la gestion de bugs, notamment dans le code d'Olivier quand ce dernier s'occupait de tâches plus complexes.

Kim s'est occupé de la banque de sons ainsi que de l'implémentation du son dans le jeu ainsi que de la gestion des animations (personnage dans le mode jeu quand il se déplace, arrivée...). Il a contribué avec Aristide à la gestion des bugs post-programme, car, comme il avait le bras cassé durant la création du code, il a surtout contribué après. Cependant, à la fin du code, alors que la période était telle qu'il était compliquée pour nous de trouver du temps, il a été sans conteste celui qui passait le plus de temps sur le projet, s'occupant aussi de l'amélioration de l'expérience utilisateur (simplification des commandes…)

En résumé, nous avons partagé les tâches de la manière la plus adaptée possible, en prenant en compte la situation et les différents points forts de chacun. Nous avons tous mis du nôtre dans ce projet, on a bien sûr tous dû travailler chez nous, en communiquant via un groupe WhatsApp. Nous nous réunissions aussi fréquemment au lycée pour faire le point sur les différentes tâches à produire, et nous avancions aussi tous ensembles là-bas. Nous avons passé en tout presque quatre mois sur ce projet, par intermittence en fonction du temps dont nous disposions. Je pense pouvoir dire qu'en additionnant les temps que chacun des membres de l'équipe a passé dessus, nous avons passé près de 100 heures

LES ÉTAPES DU PROJET :

Présenter les différentes étapes du projet (de l'idée jusqu'à la finalisation du projet)

Ce que nous voulions éviter à tout prix était d'avoir des erreurs d'origine inconnue dans notre code, car plus le code aurait été long, plus elles auraient été difficiles à localiser et à modifier. On a donc commencé, après avoir eu l'idée, à réfléchir au concept et à faire des brouillons (on a pris un certain temps avant de commencer à coder). Dans un premier temps, on a fait des croquis du jeu, et c'est là qu'on a eu l'idée d'implémenter les portails. En parallèle, Ukyo a vérifié qu'il était bien dans ses capacités de créer un algorithme trouvant le plus court chemin d'un labyrinthe avec portails (nous parlerons de l'aspect mathématique à un endroit à part). Puis Olivier a commencé à la fin du mois de décembre à coder un jeu 'prototype' avec un point comme personnage, un départ, une arrivée et des murs, à l'aide de la bibliothèque pyxel. L'idée derrière cela est que comme ce programme était fonctionnel et comme on avait prévu de l'implémenter petit à petit, on serait toujours en capacité de savoir d'où viennent les erreurs. Pour que le jeu soit « Jouable », l'étape suivante a été la création des graphismes, du personnage, des murs, du départ et de l'arrivée par Prune (à cette même période). Elle en a aussi profité pour coder divers autres items, dont certains n'ont même au final jamais été implémentés. La création de tous ces graphismes a été complexifié par l'utilisation de pyxel, qui n'accepte que 16 couleurs pour les images importées. La taille des objets importés dans le jeu etait de 8*8px pour une case (chiffre issu d'un compromis entre résolution et nombre de blocs visible), ce qui a aussi complexifié leur création. A ce moment, le projet était « Jouable », mais il n'y avait pas grand intérêt à y jouer, et l'outil était très compliqué à manier (les commandes étaient peu intuitives, il n'y avait pas de fonctionnalité permettant d'aider la création de labyrinthe…). Puis l'équipe a pu se retrouver. A ce moment-là, Prune a commencé à créer le menu, et en parallèle Aristide a commencé à s'occuper de la gestion de la sauvegarde, c'est-à-dire à créer des fichiers contenant du texte supposé représenter le labyrinthe. Olivier l'a aidé à réaliser cette partie. La sauvegarde a été plusieurs fois modifiée par la suite, notamment une fois lors de l'implémentation des portails, une fois pour vérifier que les fichiers étaient bien conformes au format demandé. Nous avons également essayé de remplacer les fichiers par les lignes cryptées, plus faciles à transmettre, mais il était compliqué de réduire leur taille suffisamment, et nous nous y sommes pris trop tard, de telle sorte que nous n'eûmes finalement pas le temps d'implémenter cette fonction. En même temps que la création de la sauvegarde, le menu fut achevé (du moins sa première version). Il nous restait donc principalement trois choses à faire : coder et implémenter l'algorithme de résolution, ajouter la mécanique des portails, et les finitions (musique, animation, correction des bugs). Ukyo s'est occupé de l'algorithme de résolution, et a choisi l'algorithme BFS (voir annexe pour le fonctionnement). Ensuite, Ukyo et Olivier ont dû implémenter les portails, car le reste de l'équipe devait modifier beaucoup de choses dans le code pour s'adapter à cette nouvelle mécanique. Les portails ont été longs à coder, notamment car nous n'avions pas prévu la possibilité de certaines interaction (portail sur un mur, portail sur un portail…), il a aussi été difficile d'exprimer clairement quel portail étaient reliés dans les fichiers sauvegarde. Après cet ajout, Kim Aristide et Prune ont corrigé les divers bugs qu'ils avaient engendré, tandis qu'Ukyo et Olivier ont implémenté au jeu l'algorithme de résolution. Après cela, on aurait pu penser le jeu fonctionnel, seulement on était tous d'accords pour se dire que l'expérience était un peu ennuyeuse, et qu'il manquait des détails. Kim a fini de créer sa musique (qu'il avait commencé en même temps que le commencement du menu et de la sauvegarde). Tout comme pour les items, la musique était compliquée à créer puisque personne dans notre groupe n'avait jamais eu l'occasion de faire de l'improvisation musicale ou de créer de mélodie (Kim a passé plusieurs heures à l'imaginer). Kim a ensuite contribué à la création des animations de départs et d'arrivées, et nous avons corrigé les quelques bugs qui restaient. Prune et Aristide ont également amélioré la partie Tkinter liée à la sauvegarde et à la résolution, pour que l'utilisation devienne plus fluide. En parallèle, nous avons bien sûr commencé à remplir les documents pour participer au trophée. En plus du travail fourni pour remplir cette documentation, le groupe a rencontré un problème lié à la vidéo. En effet, plusieurs membres du groupes ne souhaitaient pas y apparaître, et nous avons dû réfléchir à un moyen de

produire malgré tout une vidéo représentative de notre travail. Nous avons finalement opté pour une partie « Gameplay » dans laquelle on montrait à quoi ressemblait notre jeu, et une partie explicative faite avec les membres qui pouvaient se montrer.

> FONCTIONNEMENT ET OPÉRATIONNALITÉ :

Pouvez-vous présenter l'état d'avancement du projet au moment du dépôt ? (ce qui est terminé, en cours de réalisation, reste à faire)
Quelles approches avez-vous mis en œuvre pour vérifier l'absence de bugs et garantir une facilité d'utilisation de votre projet ?
Quelles sont les difficultés rencontrées et les solutions apportées ?

Au moment où j'écris ces lignes, il reste une semaine pour rendre le projet. Ce dernier n'est pourtant pas complètement fini : on n'a pas encore fait les animations de départ et surtout d'arrivée, et quelques bugs embêtants subsistent liés à la sauvegarde des portails.
Pour vérifier l'absence de bugs, notre méthode était assez simple : chaque fois que nous rajoutions un élément, nous essayions de faire tout ce qui était possible de faire avec cet élément, et nous vérifiions que cela marchait bien. C'est comme cela par exemple que nous nous sommes rendus compte qu'il était possible de mettre des portails dans des murs.
Pour garantir une facilité d'utilisation de notre projet, nous avons mis en œuvre plus de choses : nos commandes étaient les plus intuitives possibles, et nous avons un menu contenant un bouton d'aide pouvant rappeler à l'utilisateur les commandes à utiliser. Tous les boutons permettant de faire des choses hors-jeu sont regroupés dans le menu, et ce dernier est accessible en un clic.
Outre l'apparition et la correction de bugs, nous avons rencontré plusieurs problèmes majeurs :
- Le premier problème rencontré a été la vitesse d'avancement variée, qui nous a un peu perdu. On ne voyait pas vraiment combien de temps nécessiterait ce qu'on voulait faire, et combien de temps nécessiterait ce qu'on voulait ajouter, de telles sorte qu'on est parti avec beaoucp d'idées sans vraiment savoir lesquelles on allait implémenter. Pour remédier à ce problème, un mois avant le rendu, nous nous sommes imposés un cahier des charges précis, en nous disant : « On fera ça, pas plus et pas moins ! ». Un exemple d'une de ces idées qui n'a pas été ajoutée est la possibilité de rajouter des monstres poursuivant l'utilisateur dans le labyrinthe, ce qui aurait beaucoup complexifié notamment l'algorithme de résolution.
- Le second problème a été de pouvoir se répartir le travail de manière égale. Ainsi, Nous avons tous des niveaux différents en informatique, et au début, nous avons codé sans s'en soucier. On s'est retrouvé après la création du prototype avec une personne du groupe qui avait codé un quart du projet tout seul, ce qui a déjà porté atteinte à l'égalité des charges de travail. En nous définissant des rôles précis à l'avance, ce problème a lui aussi disparu, bien que les charges de travail finales soient de ce fait inégales par endroit.
- Le dernier problème a été la fermeture de l'ENT de notre lycée une semaine et demie avant la fin du rendu. Ainsi, nous avions quelques données stockées dans cette messagerie, que nous avons perdu de ce fait. Pour surmonter ce problème, nous avons tout simplement travaillé plus pour tout refaire.

> OUVERTURE :

Quelles sont les nouvelles fonctionnalités à moyen terme ? Avez-vous des idées d'amélioration de votre projet ?
Pourriez-vous apporter une analyse critique de votre projet ? Si c'était à refaire, que changeriez-vous dans votre organisation, les fonctionnalités du projet et les choix techniques ?
Quelles compétences/appétences/connaissances avez-vous développé grâce à ce concours ?
En quoi votre projet favorise-t-il l'inclusion ?

Nous avons pensé à plusieurs fonctionnalités à rajouter à moyen terme. En voici un florilège :
- L'ajout de « monstres » poursuivant l'utilisateur, rendant l'expérience meilleure et le l'algorithme de résolution plus complexe.
- Généraliser l'algorithme de résolution. Ainsi, il ne fonctionne pour l'instant seulement sur les labyrinthe d'une taille maximale de 200*200 blocs.
- Permettre à l'utilisateur de changer dans le menu les commandes de jeu. Cette fonctionnalité paraît simple à implémenter, mais nous y avons pensé après avoir fait notre cahier des charges, et avons préféré finir le reste.
- Changer le système de fichiers par des lignes cryptées, facilitant le transmission de labyrinthes.
-Créer un système d'enregistrement de temps de résolution de labyrinthe, pour permettre la compétition entre amis.
On a aussi pensé à beaucoup d'autres fonctionnalités, et un des points positifs de ce projet est qu'il est quasiment améliorable à l'infini, signifiant que le trophée NSI n'est pas une fin, mais un début pour lui. Qui sait, peut-être même que vous le retrouverez l'année prochaine !

Cependant, notre projet a aussi des inconvénients, le principal étant son abscence de point fort très prononcé. Ainsi, on voulait en faire un jeu, mais ce côté n'a pas été assez développé, au profit de la polyvalence (algorithme de résolution, place importante du système de sauvegarde…), de telle sorte que l'expérience utilisateur n'est pas assez bien pour qu'on puisse considérer notre pojet comme un « bon jeu ». De plus, certaines commandes (surtout liées aux portails) ne sont pas toujours pratiques à comprendre Si c'était à refaire, on pense donc qu'il faudrait plus accentuer un côté sur les autres, ici l'expérience utilisateur, pour rendre ce projet plus ludique et plus proche d'un véritable jeu. Par exemple, il aurait peut-être été plus judicieux de choisir pygame à pyxel comme bibliothèque de jeu, afin d'avoir accès à de meilleurs graphismes, et de, pour compenser cela, faire des concessions sur d'autres sujets.

Ce projet a surtout développé deux compétence chez nous : la discipline, et l'humilité. Ainsi, nous avons appris au fur et à mesure du projet à travailler de mieux en mieux chez nous (quand nous nous retrouvions, c'était toujours un peu compliqué d'être sérieux, mais cela était néanmoins bien mieux à la fin du projet qu'au début). De plus, on a eu beaucoup moins de problèmes liés à une surestimation de nos capacités et de notre vitesse aux dernière modifications qu'au première. C4est pour cela que je pense pouvoir dire que ce projet, notre premier vrai projet informatique, nous a permis de débloquer ce deux compétences. Plus techniquement, nous avons aussi appris à manier des fichiers, ainsi qu'à utiliser Tkinter (on avait déjà des connaissances théoriques avant le projet, mais la théorie et la pratique sont vraiment deux choses distinctes).

> ANNEXE MATHEMATIQUE :

Étant donné un graphe muni de ses noeuds/sommets, l'algorithme BFS permet de déterminer un des plus courts chemins reliant deux noeuds.
L'algorithme prend en entrée un graphe non pondéré et l'identifiant du sommet source s. Le graphe d'entrée peut être orienté ou non orienté (c'est le cas avec le labyrinthe), mais cela n'a pas d'importance pour l'algorithme.
L'algorithme peut être compris comme un feu se propageant sur le graphe : à l'étape zéro, seul le sommet source s est en feu. À chaque étape, le feu brûlant à chaque sommet se propage à tous ses voisins. Lors d'une itération de l'algorithme, ce "cercle de feu" est étendu en largeur d'un cran (d'où le nom de l'algorithme Breadth-first Search).
Plus concrètement, l'algorithme peut être formulé comme suit : on crée une file d'attente q qui contiendra les sommets à traiter et un tableau booléen utilisé[] qui indique pour chaque sommet s'il a été allumé (ou visité) ou non. Initialement, on ajoute le sommet source s à la file d'attente et on définit utilisé[s] = Vrai, et pour tous les autres sommets v, on dit que utilisé[v] = Faux. Ensuite, on boucle ceci jusqu'à ce que la file d'attente soit vide, et à chaque itération, on extrait un sommet de l'avant de la file d'attente. On parcourt toutes les arêtes sortant de ce sommet et si certaines de ces arêtes mènent à des sommets qui ne sont pas encore allumés, on les met en feu et on les place dans la file d'attente.

En conséquence, lorsque la file d'attente est vide, le "cercle de feu" contient tous les sommets accessibles à partir du sommet source s, chaque sommet atteint de la manière la plus courte possible. On peut également calculer les longueurs des plus courts chemins (ce qui nécessite simplement le maintien d'un tableau de longueurs de chemin distance[]) ainsi que sauvegarder des informations pour restaurer tous ces plus courts chemins (pour cela, il est nécessaire de maintenir un tableau parent[], qui stocke pour chaque sommet le sommet à partir duquel nous l'avions atteint).

Le pseudo-code :

```
Entrée :
Une liste d'adjacences adj
Le nombre de sommets n
La source/noeud de départ s
La destination f


File d'attente q
Un tableau de booléens utilisé
Deux tableau d'entiers, distance et parent (initialisé à -1)

Ajouter s à q
utilisé[s] ← Vrai
Parent[s] ← -1 (pour dire que s n'a pas de "parent")
Distance[s] ← 0

Tant que q n'est pas vide
        Considérer v le sommet de la file d'attente
        Retirer ce sommet de la file
        Pour chaque noeud u dans adj[v]
                Si utilisé[u] est Faux
```

utilisé[u] ← Vrai
Distance[u] ← Distance[v] + 1
Parent[u] ← v
Ajouter u dans la file d'attente

Si utilisé[f] est Faux
Retourner qu'il n'y a pas de chemin
Sinon
On retourne le chemin qui peut être reconstruit en remontant en arrière à l'aide du tableau parent

L'algorithme est en O(V + E), où V est le nombre de sommets et E le nombre d'arêtes.

Ce document est l'un des livrables à fournir obligatoirement lors du dépôt de votre projet : 4 pages maximum. Le non-respect du modèle fourni peut impacter la notation.

La documentation technique complète est à intégrer dans le dossier technique, dans un répertoire nommé doc.

Pour accéder à la liste complète des éléments à fournir, consultez la page **Comment participer ?**.

Vous avez des questions sur le concours ? Vous souhaitez des informations complémentaires pour déposer un projet ?

Contactez-nous à **info@trophees-nsi.fr** ou consulter la page **Foire aux questions**.