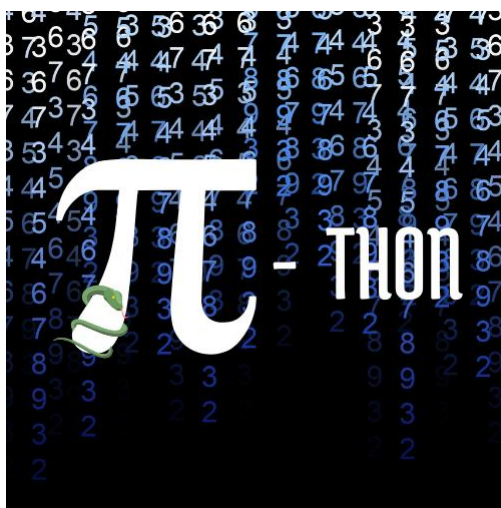




# Édition 2025

## PRÉSENTATION DU PROJET



Nom de votre projet	Π-THON
Membre de l'équipe n°1 (prénom/nom)	Arthur Jeaugey
Membre de l'équipe n°2 (prénom/nom)	Samuel Mopty
Membre de l'équipe N°3 (prénom/nom)	Paul Chevasson
Niveau d'étude	Première générale
Établissement scolaire	Lycée Notre-Dame, 21000 Dijon
Responsable du dépôt (professeur de NSI)	M. P.Moreau

# 1 / PRÉSENTATION GÉNÉRALE

Notre projet consiste à développer différentes méthodes d'estimation du nombre  $\pi$  grâce au langage informatique Python, en combinant approche mathématique et programmation graphique avec la bibliothèque Pygame.

Ce projet est né de notre intérêt commun pour les mathématiques et l'informatique, renforcé par l'invitation et l'initiation de notre professeur à participer au concours des Trophées NSI.

Notre problématique initiale était de savoir comment estimer précisément et efficacement la valeur de  $\pi$  en utilisant diverses méthodes mathématiques, tout en rendant celles-ci visuellement compréhensibles et interactives grâce à un affichage graphique unique et cohérent.

L'objectif principal est d'offrir une interface intuitive regroupant plusieurs méthodes classiques (Monte Carlo, Buffon, Archimède...) permettant à l'utilisateur de visualiser et de comparer facilement leur précision et leur performance. Nous répondons ainsi au besoin d'un outil pédagogique clair et interactif, utile tant pour l'enseignement que pour approfondir la compréhension du nombre  $\pi$  et des concepts algorithmiques associés, et faire de la vulgarisation scientifique.

De plus, la capacité de notre programme à calculer un grand nombre de décimales et à pouvoir générer des fichiers textes avec un nombre de décimales voulues (exemple ci-dessous) peut s'avérer utile pour des applications dans les domaines scientifiques et cryptographiques.

Enfin, le calcul d'un grand nombre de décimales notamment avec la méthode de Chudnovsky est souvent utilisé comme un moyen de benchmark pour évaluer la puissance de processeurs, notre application peut donc être un bon moyen de benchmark (en rajoutant certaines fonctionnalités), notamment en comparant la vitesse de calcul par rapport à l'estimation de temps initiale. Cela peut par exemple permettre de comparer les performances de plusieurs machines entre elles.

Méthode de Gauss-Legendre

Nombre de décimales : 500 mille

Temps de calcul : 00:00:00

L'estimation de  $\pi$  est correcte !

Estimation de  $\pi$  :

```
3.1415926535897932384626433832795028841971693993751058209749445923078164062862089986280348253421
886593615338182796823030195203530185296899577362259941389124972177528347913151557485724245415069
407726719478268482601476990902640136394437455305068203496252451749399651431429809190659250937221
920773467221825625996615014215030680384477345492026054146659252014974428507325186660021324340881
791045334885034611365768675324944166803962657978771855608455296541266540853061434443185867697514
940924323789690706977942236250822168895738379862300159377647165122893578601588161755782973523344
825202618798531887705842972591677813149699009019211697173727847684726860849003377024242916513005
387993620162951541337142489283072201269014754668476535761647737946752004907571555278196536213239
361031029161615288138437909904231747336394804575931493140529763475748119356709110137751721008031
218654364802296780705765615144632046927906821207388377814233562823608963208068222468012248261177
510716378529023452924407736594956305100742108714261349745956151384987137570471017879573104229690
36985891754426473998822846218449008776977631279572267265556259628254276531830013407092233436577
359210031541508936793008169980536520276007277496745840028362405346037263416554259027601834840306
068691709264625484232407485503660801360466895118400936686095463250021458529309500009071510582362
```

## 2 / ORGANISATION DU TRAVAIL




### Présentation des membres et leurs rôles :

- **Arthur** (Coordinateur du groupe et développeur principal) :  
Il a structuré le projet, intégré l'ensemble des méthodes en partie codées par Samuel, créé les fonctions utilitaires nécessaires, et veillé à la cohérence globale du code.
- **Samuel** (Mathématicien du groupe) :  
Il s'est occupé de rechercher et sélectionner les différentes méthodes mathématiques permettant d'estimer  $\pi$ , de comprendre leur fonctionnement détaillé, et de réfléchir à leur intégration optimale dans notre programme, tout en assurant leur bonne explication.
- **Paul** (Responsable de l'interface graphique et vidéaste) :  
Il a réalisé toute la partie graphique du projet, en particulier l'intégration visuelle et interactive des méthodes dans notre interface Pygame, ainsi que la création des éléments graphiques nécessaires au programme.

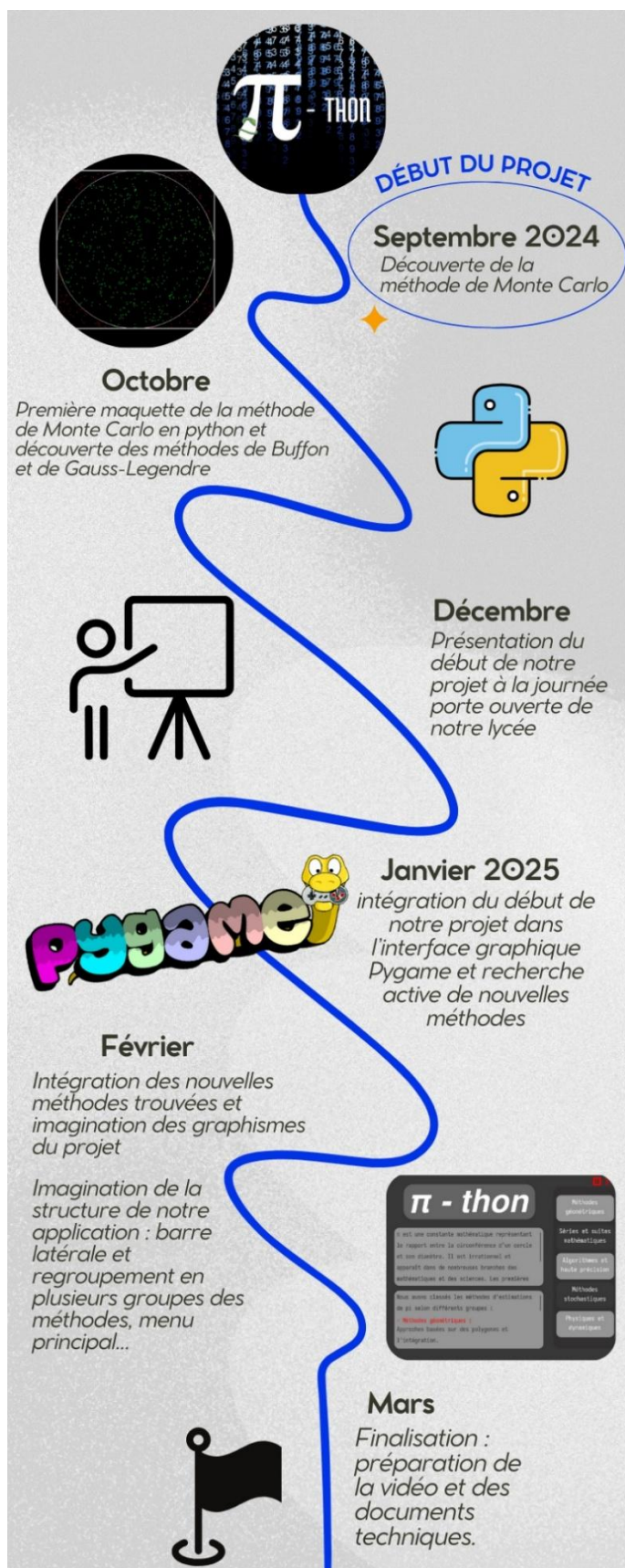
Étant tous les trois âgés de 16 à 17 ans et élèves du Lycée Notre Dame, nous formons un groupe d'amis passionnés par l'informatique, ayant l'habitude de collaborer ensemble. L'organisation du travail s'est naturellement orientée en fonction des centres d'intérêt et des points forts de chacun : Samuel apprécie particulièrement les mathématiques et possède d'excellentes compétences théoriques dans ce domaine, Arthur est passionné par la programmation et est celui qui a le plus d'expérience dans ce domaine, tandis que Paul est passionné par le design et les aspects graphiques, domaines dans lesquels il excelle. Cette répartition équilibrée a permis à chaque membre de participer efficacement au projet en fonction de ses compétences et préférences personnelles, assurant ainsi une cohésion optimale et une avancée efficace.

Nous avons passé environ 3 **mois en tout** sur ce projet (En réalité, nous travaillons dessus depuis le mois de septembre, mais nous nous sommes attelés activement au développement de l'application seulement depuis le mois de décembre).

### Outils et logiciels utilisés pour la communication et le partage du code :

-  **Discord** (communication quotidienne et réunions pour faire le point sur l'avancement)
-  **VSCode** (édition collaborative du code)
-  **Cloud OneDrive** (partage direct et synchronisé du dossier du projet pour une gestion efficace et continue du code)

### 3 / ÉTAPES DU PROJET



Nous avons découvert la méthode de Monte Carlo au début de l'année, avant même d'avoir une idée précise de projet. Cette méthode nous a immédiatement fascinés par sa simplicité et son aspect interactif. Avec l'aide de nos professeurs de mathématiques et de NSI, nous avons réalisé qu'il existait bien d'autres méthodes pour estimer  $\pi$ , chacune reposant sur des concepts mathématiques différents.

Après avoir présenté notre projet à la journée porte ouverte de notre lycée, nous avons décidé de regrouper ces différentes approches dans un même programme avec un affichage graphique sous pygame afin de les comparer visuellement et de mieux comprendre leur fonctionnement. Puis nous avons commencé une longue phase de recherche pour identifier les méthodes les plus intéressantes et adaptées à notre projet. Certaines ont en effet dû être écartées à cause de leur complexité (Par exemple la méthode de Bailey-Borwein-Plouffe).

Une fois cette base posée, nous avons progressivement développé et ajouté chaque méthode, en veillant à ce qu'elles s'intègrent correctement dans notre structure. Parallèlement, nous avons travaillé sur l'interface graphique avec Pygame pour que l'affichage soit à la fois intuitif et interactif.

Enfin, nous avons optimisé le code, corrigé les bugs et amélioré l'affichage pour assurer un rendu fluide et fonctionnel. Après trois mois de travail, nous avons obtenu un programme complet et interactif, regroupant plusieurs méthodes d'estimation de  $\pi$  dans une seule interface cohérente.

## 4 / FONCTIONNEMENT ET OPÉRATIONNALITÉ

### État d'avancement du projet au moment du dépôt

Le projet est opérationnel avec 13 méthodes d'estimation de  $\pi$  intégrées et affichées via une interface unique sous Pygame. La structure du code est bien définie avec une séparation claire entre les fonctions, améliorant ainsi la lisibilité et la maintenabilité. Cependant, certaines optimisations restent à faire, comme la centralisation de la barre de progression, mais aussi la division des longues fonctions des méthodes en plusieurs fonctions. Par exemple, pour les méthodes performantes, nous pourrions séparer la partie calcul de la partie graphique dans des fonctions différentes. De plus, nous avons rencontré une limite à la précision pour les méthodes les plus performantes et ne pouvons actuellement obtenir environ 300 millions de décimales au maximum avec ces méthodes. Nous avons pour objectif de trouver une solution pour contrer cette limite et atteindre un objectif que nous nous sommes fixés de calculer 1 milliard de décimales.

### Vérification des bugs et facilité d'utilisation

Nous avons procédé à des tests systématiques en isolant chaque méthode pour détecter d'éventuelles erreurs. Un débogage manuel a été effectué en repérant les comportements inattendus, complété par des logs de contrôle. Le fichier `utils.py` a permis d'éviter les répétitions de code et d'assurer une meilleure réutilisation. Pour garantir une facilité d'utilisation, notre application s'ouvre en exécutant le fichier `main.py`. L'estimation du temps de calcul permet également d'améliorer l'expérience utilisateur en l'informant du temps de calcul théorique.

### Difficultés rencontrées et solutions apportées

- **Redondance du code** : utilisation du fichier `utils.py` pour centraliser certaines fonctionnalités communes.
- **Temps de calcul long sur certaines méthodes** : utilisation de la bibliothèque `gmpy2` pour rendre le calcul beaucoup plus rapide, et affichage progressif des décimales envisagé pour éviter une attente trop longue.
- **Affichage non optimisé parfois pour le plein écran, parfois pour le fenêtré** : affichage dynamique des éléments graphiques en utilisant la taille de l'écran pour le placement des éléments et non un placement fixe.
- **Navigation** : auparavant, pour changer de méthodes nous devions relancer le programme, nous avons donc incorporé des moyens de navigations comme un bouton de retour au menu.



## 5 / OUVERTURE

### Idées d'amélioration et nouvelles fonctionnalités à moyen terme

- **Accessibilité** : ajout d'un mode audio et amélioration de la gestion des couleurs et de la luminosité pour les daltoniens.
- **Optimisation du code** : réduction des fichiers trop longs en divisant certaines fonctions, déplacement de la gestion du menu principal dans un fichier plus adapté, centralisation de certaines fonctionnalités redondantes (barre de progression, estimation de temps...)
- **Meilleure interactivité** : intégration d'un affichage progressif des résultats lors des calculs longs.
- **Plus de décimales** : nous recherchons activement un moyen de dépasser la barre des 300 millions de décimales maximum actuelle.

### Analyse critique du projet

Le projet est bien structuré avec une séparation claire des modules et un code bien commenté, facilitant sa compréhension et sa réutilisation. L'intégration de la bibliothèque gmpy2 permet une précision élevée et des calculs plus rapides. L'utilisation du programme est intuitive, avec des boutons permettant une navigation fluide, ou des informations sur le temps de calcul pour les méthodes performantes. De nombreuses méthodes d'estimations de pi sont intégrés rendant notre programme complet. Cependant, certaines fonctions sont trop longues et nécessiteraient d'être divisées pour un code plus lisible. De plus certaines fonctionnalités telles que l'affichage progressif des décimales en même temps que le calcul pourrait être ajoutées.

Si c'était à refaire, nous aurions mieux organisé la structure du projet dès le départ, et testé plus rapidement certaines méthodes pour éviter des réajustements tardifs.

### Compétences développées

Ce projet nous a permis de développer des compétences et connaissances dans divers domaines : pour commencer nous avons tous une connaissance plus précise de l'histoire des mathématiques et des méthodes d'estimations de pi mais aussi de connaissances directes telles que la découverte des intégrales et de certaines fonctions mathématiques. De plus, ce projet nous a permis une amélioration de nos compétences en programmation et en gestion d'un projet sous python, avec une compréhension accrue de la structure à adopter. Enfin, cela nous a aussi permis d'apprendre à utiliser une bibliothèque graphique : Pygame entre l'apprentissage de la gestion des événements, de l'affichage, ou aussi des fonctions internes comme la gestion du temps avec cette bibliothèque.

### Favoriser l'inclusion

Pour une meilleure inclusion, l'ajout d'options comme la lecture audio et un mode daltonien permettrait d'élargir l'accessibilité du projet.