Université Paris Nanterre

Licence MIASHS Deuxième Année

Rapport de projet informatique

Visualisation de tri

Projet réalisé du 15 novembre 2024 au 6 janvier 2025

https://github.com/kevkan75/VisualisationDeTri

Membres du groupe

BENDJEBBAR Adam 43006224
KAN Kevin 43005517

Remerciements

La réalisation de ce projet n'aurait pas été possible sans le soutien et les contributions précieuses de nombreuses personnes, que nous souhaitons ici remercier chaleureusement.

Tout d'abord, nous adressons notre gratitude à notre enseignant, M. BOUQUET Valentin, pour son accompagnement bienveillant, ses orientations éclairées et son soutien constant. Ses conseils avisés nous ont permis de surmonter les défis rencontrés et d'atteindre nos objectifs. Son aide précieuse a été déterminante pour mieux cerner les exigences et attentes du projet en nous guidant. Ainsi que M. DELBOT François, enseignant en cours magistral, pour la qualité des enseignements dispensés tout au long du semestre. Bien que son rôle ait été davantage centré sur l'aspect théorique, ses cours ont contribué à renforcer nos connaissances générales, utiles à l'approche de ce projet.

Nous exprimons également nos sincères remerciements à nos camarades de travail, dont l'énergie, l'engagement et l'esprit d'équipe ont été une source constante de motivation. Leur enthousiasme et leur collaboration ont rendu cette expérience enrichissante et agréable.

Enfin, nous remercions l'Université Paris Nanterre pour les ressources pédagogiques et matérielles mises à notre disposition. Ce cadre propice à l'apprentissage a grandement facilité le déroulement de notre travail

Ce projet est le fruit d'un effort collectif et d'un soutien indéfectible. À tous ceux qui ont contribué, directement ou indirectement, à cette aventure, nous adressons nos remerciements les plus sincères. Votre appui a été essentiel à notre réussite.

Table des matières

1 Introduction

Les algorithmes de tri occupent une place centrale dans l'informatique moderne, jouant un rôle clé dans l'organisation et la manipulation des données. Qu'il s'agisse de trier des listes, des tableaux ou des structures plus complexes, ils sont indispensables dans des domaines variés comme les bases de données, les moteurs de recherche ou encore les outils d'intelligence artificielle. Pourtant, malgré leur importance, leur compréhension reste complexe, en raison des concepts abstraits et des mécanismes spécifiques propres à chaque algorithme.

La théorie des algorithmes de tri est généralement enseignée à travers des descriptions textuelles ou du pseudo-code. Si ces méthodes permettent une compréhension théorique, elles montrent difficilement les manipulations réelles des éléments à chaque étape. De plus, appréhender les différences de performance entre les algorithmes (en termes de temps d'exécution ou de complexité spatiale) peut être ardu sans une observation pratique et interactive de leur fonctionnement.

Face à ces défis, ce projet propose une approche pédagogique novatrice : une application web interactive permettant de visualiser le déroulement des algorithmes de tri pas à pas, tout en offrant la possibilité d'analyser et d'exécuter des algorithmes personnalisés écrits en langage C. L'objectif est de rendre les processus abstraits plus concrets et accessibles, que ce soit pour les étudiants en informatique, les développeurs ou toute personne souhaitant approfondir ses connaissances.

Ce projet se distingue par sa double approche. D'une part, il offre une visualisation détaillée des étapes intermédiaires des tris préimplémentés, facilitant la comparaison entre des algorithmes comme le tri à bulles, le tri rapide ou le tri par insertion, sur des jeux de données variés. D'autre part, il propose une plateforme permettant de tester et de déboguer des algorithmes personnalisés. Les utilisateurs peuvent observer les étapes de leur exécution ou recevoir des messages d'erreur clairs en cas de compilation infructueuse.

Grâce à cette interface intuitive et interactive, ce projet vise à réduire l'écart entre la théorie et la pratique, tout en encourageant une exploration libre et expérimentale des concepts fondamentaux liés aux algorithmes de tri.

2 Présentation du projet

2.1 Objectifs principaux

1. Visualisation étape par étape des tris : L'objectif principal est de fournir une visualisation claire et détaillée des algorithmes de tri. Chaque étape intermédiaire, qu'il s'agisse d'une comparaison, d'une permutation ou d'une mise à jour des données, est affichée, permettant aux utilisateurs de comprendre le fonctionnement interne des algorithmes. Cela facilite l'apprentissage, en particulier pour les étudiants et les développeurs débutants.

2. Analyse de code utilisateur : Outre les algorithmes intégrés, l'application offre aux utilisateurs la possibilité de tester leurs propres implémentations en C. Le projet analyse, compile et exécute ces codes, fournissant un retour visuel étape par étape en cas de succès ou signalant les erreurs de compilation pour un débogage efficace.

2.2 Fonctionnalités détaillées

Le projet intègre une gamme de fonctionnalités interactives et éducatives, structurées autour de deux modules principaux.

Tout d'abord le Module de visualisation des tris. Les utilisateurs peuvent choisir un algorithme parmi une liste prédéfinie, incluant différents tris comme les tri à bulles, tri rapide, tri par insertion, tri par sélection, tri fusion, etc... Une fois l'algorithme sélectionné, les utilisateurs saisissent une liste de données (entiers) à trier. Puis l'application affiche ensuite, étape par étape, le déroulement du tri, mettant en évidence les comparaisons et permutations effectuées.

Ensite nous avons le Module d'analyse et d'exécution de code. Les utilisateurs peuvent rédiger ou coller leur propre code d'algorithme en C dans une interface dédiée. Le code est compilé via Clang. En cas d'échec de la compilation, des messages d'erreur détaillés sont retournés pour aider au débogage. Si le code est valide, chaque étape de l'exécution est visualisée, comme pour les algorithmes intégrés.

2.3 Technologies utilisées

Ce projet repose sur une combinaison de technologies complémentaires pour garantir une application performante et intuitive :

VSCode : Afin de coder dans différents langages de programmation.

ChatGPT : C'est un projet assisté par intelligence Artificielle. Il est utilisé pour nous guider, nous générer et corriger du code.

HTML et CSS : L'interface utilisateur, développée en HTML et stylisée avec CSS, privilégie une expérience simple et intuitive, accessible même aux utilisateurs moins familiers avec les technologies web.

Python et Flask : Python est utilisé pour gérer la logique des algorithmes et l'analyse du code utilisateur en C. Et Flask est un framework web qui gère les interactions utilisateur, les routes et la communication client-serveur.

Langage C : Les algorithmes intégrés sont implémentés en C pour des performances optimales. L'analyse et l'exécution de code utilisateur sont également réalisées grâce à l'intégration de Clang.

Clang : Clang est utilisé pour compiler et analyser le code en C soumis par les utilisateurs, offrant des messages d'erreur précis en cas d'échec.

GitHub : Le projet est hébergé sur GitHub, facilitant la collaboration, le suivi des versions et l'accès à la documentation et au code source.

2.4 Structure de l'application

Le projet adopte une architecture client-serveur où le client (navigateur web) permet aux utilisateurs de saisir des données, de choisir des algorithmes et de visualiser les résultats. Le serveur est basé sur Flask qui traite les demandes puis exécute les algorithmes et renvoie les résultats ou les étapes de tri.

Le programme est organisé en deux modules :
Un module de tri qui implémente les algorithmes intégrés et leur visualisation.
Et un module d'analyse qui gère l'exécution et la compilation du code utilisateur ainsi que l'affichage des étapes intermédiaires ou des messages d'erreur.

Notre interface est divisée en deux. La première correspond à la visualisation des tris. Elle permet aux utilisateurs de choisir un tri, de saisir une liste de données et d'observer les étapes de ce tri.
La seconde est cellle d'analyse de code. Elle fournit un espace pour entrer du code en C ; Si il y a des erreurs de compilation, elle les affiche. S'il n'y en a pas elle affiche les différents étapes intermédiaires du programme donner par l'utilisateur en C.

3 Environnement de travail

Pour réaliser ce projet, nous avons sélectionné des outils et des technologies adaptés, en veillant à leur complémentarité et à leur efficacité. Ces choix ont été guidés par les besoins spécifiques du projet, notamment la gestion d'algorithmes en Python, la compilation de code en C, et la création d'une interface web interactive.

Visual Studio Code a été notre principal environnement de développement intégré (IDE). Ce choix s'est imposé grâce à sa flexibilité et ses nombreuses extensions, qui ont grandement facilité le travail sur les différents langages du projet (Python, HTML, CSS). Cet outil a également été crucial pour le débogage et l'organisation du code.

Le framework Flask a été employé pour développer le backend de l'application. Sa légèreté et son efficacité ont permis de gérer les interactions entre les utilisateurs (via leur navigateur) et le serveur. De plus, Flask s'intègre parfaitement avec les bibliothèques Python nécessaires à l'exécution des algorithmes et au traitement des données.

Pour traiter les codes en C soumis par les utilisateurs, nous avons intégré l'outil Clang et son analyseur d'Arbre Syntaxique Abstrait (AST). Cet outil a été choisi pour sa précision et sa capacité à fournir des messages d'erreur détaillés, aidant ainsi les utilisateurs à comprendre et à corriger leur code. Clang assure également une exécution fluide des algorithmes validés.

Le projet est hébergé sur GitHub, une plateforme qui a permis un suivi collaboratif efficace et une gestion rigoureuse des versions. Le dépôt contient le code source, la documentation et toutes les ressources nécessaires à l'installation et à l'utilisation du projet.

L'interface utilisateur a été conçue avec HTML et CSS pour offrir une expérience fluide et intuitive. Nous avons veillé à ce que la navigation entre les différentes fonctionnalités soit simple, même pour les utilisateurs peu expérimentés. Les tests ont été réalisés sur plusieurs navigateurs modernes comme Safari, Chrome et Firefox afin de garantir une compatibilité optimale.

L'organisation du projet suit une structure claire et bien définie :
- Le répertoire `/static` contient le fichier CSS et les ressources graphiques (logo).
- Le répertoire `/templates` regroupe les fichiers HTML pour l'interface utilisateur.
- Le répertoire `/sorting_algorithms` comprend les douze algorithmes de tri que l'utilisateur peut utiliser.
- Le fichier principal `app.py` orchestre l'ensemble de l'application via Flask.
- Le fichier `analyze_ast.py` permet l'utilisation de l'AST Clang.

Pour simplifier l'installation et la configuration, un fichier `requirements.txt` a été créé. Il liste toutes les bibliothèques nécessaires (Flask, Clang, etc.) et aide à l'éxécution du programme.

Enfin, les tests et le débogage ont été réalisés en mode développement local, en activant les logs de Flask pour détecter et corriger rapidement les éventuelles anomalies.

Ce choix d'outils et cette configuration d'environnement ont permis de mener à bien le projet de manière structurée et efficace, tout en garantissant une interface conviviale et des performances optimales.

4 Conception du projet

La conception du projet a été élaborée dans un souci de simplicité, d'efficacité et de modularité. L'objectif était de développer une application web interactive capable de gérer plusieurs fonctionnalités essentielles tout en restant évolutive et facile à maintenir. Cette section décrit les choix architecturaux, les mécanismes des principales fonctionnalités et le design de l'interface utilisateur.

4.1 Architecture logicielle

Le projet repose sur une architecture classique client-serveur. Le client, accessible via un navigateur web, propose une interface utilisateur intuitive permettant d'interagir avec l'application. Toutes les actions effectuées par l'utilisateur, comme la sélection d'un algorithme de tri ou la soumission d'un code en C, sont transmises au serveur. Le serveur, développé avec Flask, traite ces demandes, exécute les algorithmes ou compile les codes soumis, puis renvoie les résultats au client.

Cette architecture sépare clairement les responsabilités : le client gère l'affichage et l'interaction, tandis que le serveur prend en charge la logique métier et le traitement des données. Cette division facilite l'évolution du projet, permettant d'ajouter de nouvelles fonctionnalités sans altérer l'interface utilisateur.

4.2 Fonctionnement des principales fonctionnalités

L'application se divise en deux modules principaux, chacun répondant à des besoins spécifiques :

- Module des algorithmes de tri préimplémentés : Ce module permet aux utilisateurs de choisir un type de tri (tri à bulles, tri rapide, tri par insertion, etc.) et de saisir une liste d'entiers à trier. Une fois le processus lancé, le serveur exécute l'algorithme sélectionné et renvoie les étapes intermédiaires, qui sont affichées sur l'interface utilisateur. Cette visualisation pas à pas aide les utilisateurs à comprendre le fonctionnement de l'algorithme.
- Module d'analyse et d'exécution de code en C : Les utilisateurs peuvent insérer leur propre algorithme de tri en langage C dans une zone de texte dédiée. Le code est envoyé au serveur pour être compilé avec Clang. En cas d'erreurs, des messages explicites sont renvoyés à l'utilisateur pour l'aider à les corriger. Si le code est valide, il est exécuté et ses étapes intermédiaires sont visualisées comme pour les algorithmes intégrés. Ce module favorise l'apprentissage actif et l'expérimentation.

4.3 Design de l'interface utilisateur

Nous avons développé un algorithme robuste capable de prendre en compte les contraintes spécifiées par les utilisateurs, telles que les disponibilités des ressources et les préférences des professeurs et des élèves. Cet algorithme utilise des techniques d'optimisation pour générer des emplois du temps adaptés et optimisés en temps réel.

En résumé, la conception du projet repose sur une architecture robuste, des mécanismes bien définis et un design ergonomique, offrant une application fonctionnelle et prête à évoluer en fonction des besoins futurs.

5 Implémentation

L'implémentation de ce projet s'est concentrée sur la mise en œuvre optimale des algorithmes de tri, l'intégration des outils nécessaires à l'analyse de code, et la création d'une interface utilisateur intuitive. Chaque composant a été minutieusement développé et testé pour offrir une expérience utilisateur fluide et cohérente.

5.1 Développement des algorithmes de tri

Tous les algorithmes de tri intégrés dans l'application ont été implémentés en langage C, un choix motivé par la performance et l'adéquation de ce langage avec le module d'analyse de code. Les algorithmes sélectionnés couvrent diverses méthodes adaptées à différents cas d'usage, par exemple :

- Tri à bulles : Simple à comprendre, utilisé principalement à des fins pédagogiques.
- Tri rapide : Connu pour son efficacité dans une grande variété de scénarios.
- Tri par insertion : Particulièrement adapté aux petites listes de données.
- Tri par sélection et Tri fusion : Proposant des approches variées et complémentaires.

Chaque algorithme a été modifié pour enregistrer ses étapes intermédiaires dans un format lisible. Ces étapes sont transmises au serveur, qui les relaie au client pour affichage. Ce mécanisme garantit une visualisation détaillée et éducative du fonctionnement interne des algorithmes.

5.2 Analyse et exécution de code en C

Le projet intègre Clang pour fournir une fonctionnalité avancée d'analyse et d'exécution de code. Lorsqu'un utilisateur soumet son code, celui-ci est transmis au serveur où il est analysé. Clang identifie les erreurs de syntaxe ou de logique et retourne des messages explicites pour aider l'utilisateur à corriger son code.

En cas de compilation réussie, le code est exécuté avec les données d'entrée fournies. Les résultats intermédiaires sont ensuite enregistrés et transmis au client pour visualisation. Cette fonctionnalité encourage une exploration approfondie des algorithmes tout en aidant les utilisateurs à perfectionner leurs implémentations grâce à des retours détaillés.

5.3 Intégration de Flask et gestion des routes

Le framework Flask joue un rôle central dans l'orchestration de l'application en gérant les interactions entre le client et le serveur. Ses responsabilités incluent :

- La réception des données utilisateur (listes à trier ou code en C).
- L'exécution des algorithmes ou la compilation des codes soumis.

— La transmission des résultats (étapes des tris ou messages d'erreur).

Chaque fonctionnalité est associée à une route Flask spécifique. Par exemple, une route est dédiée à l'exécution des algorithmes intégrés, tandis qu'une autre gère l'analyse de code. Cette organisation claire des routes simplifie la gestion des fonctionnalités et rend l'application plus facile à maintenir.

5.4 Conception de l'interface utilisateur

L'interface utilisateur, développée en HTML et CSS, garantit une navigation intuitive et fluide. Les interactions client-serveur sont gérées de manière asynchrone, éliminant les rechargements de pages inutiles. L'interface se divise en deux pages principales :

— Une page permettant de sélectionner un algorithme, de saisir une liste d'entiers, et de visualiser les étapes du tri.

— Une page dédiée à l'écriture ou à l'importation de code en C, avec les résultats (erreurs ou étapes du tri) affichés directement sous la zone de saisie.

Des efforts ont été faits pour rendre l'interface ergonomique, avec des boutons bien placés, des zones de saisie claires, et une présentation lisible des résultats.

5.5 Tests et validations

Pour assurer la fiabilité et la robustesse du projet, une phase de tests approfondis a été réalisée. Cette étape a permis d'évaluer les différentes fonctionnalités, de détecter et corriger les anomalies, et de garantir une expérience utilisateur fluide. Les tests ont porté sur les algorithmes, l'analyse de code, et l'interface utilisateur.

Les algorithmes intégrés ont été testés avec des ensembles de données variés comme des listes courtes et longues, déjà triées, en ordre décroissant et des listes contenant des valeurs répétées...

Ces tests ont permis de vérifier l'exactitude des tris, ainsi que la cohérence des étapes intermédiaires affichées à l'utilisateur. Par exemple, pour une liste de 5 éléments triée avec le tri à bulles, chaque permutation a été comparée aux attentes théoriques pour s'assurer que l'algorithme fonctionne correctement.

6 Difficultés rencontrées

Comme tout projet informatique, la réalisation de cette application a été marquée par divers défis techniques et organisationnels. Ces obstacles, bien qu'exigeants, ont offert de précieuses opportunités d'apprentissage et ont permis d'améliorer le projet. Voici un aperçu des principales difficultés rencontrées et des solutions mises en œuvre.

Notre solution offre une approche flexible, intuitive et efficace pour la création d'emplois du temps personnalisés, répondant ainsi aux besoins croissants en matière de planification dans les établissements éducatifs et les organisations professionnelles.

Cependant, malgré ses nombreux avantages, notre système présente quelques limites et points d'amélioration potentiels. Par exemple, la gestion des conflits de planification pourrait être améliorée pour fournir des solutions plus optimales dans certains cas. De plus, une plus grande personnalisation des emplois du temps en fonction des préférences individuelles des utilisateurs pourrait améliorer l'expérience utilisateur.

6.1 Apprentissage de nouvelles technologies

Le projet reposait sur des outils relativement nouveaux pour l'équipe, comme Flask et Clang, malgré l'utilisation de flask en première année, où nous n'arrivions plus à faire fonctionner notre programme, nous avions oublié l'utilisation. Leur maîtrise a nécessité une phase initiale d'apprentissage approfondi. Par exemple, l'intégration de Clang pour compiler et analyser des codes en C a été particulièrement complexe, notamment pour gérer les erreurs de compilation tout en assurant la compatibilité avec le serveur Flask.

En solution, l'équipe a utilisé des ressources en ligne, comme la documentation officielle et des tutoriels spécialisés. Des tests répétés sur des exemples simples ont également permis une prise en main progressive de ces outils.

6.2 Synchronisation entre le frontend et le backend

Assurer une communication fluide entre l'interface utilisateur (frontend) et le serveur Flask (backend) a présenté des défis. Il s'agissait notamment de transmettre les données saisies par l'utilisateur et d'afficher les résultats de manière cohérente. La gestion des entrées, des erreurs et des mises à jour dynamiques constituait une difficulté récurrente.

Solution : Une gestion rigoureuse des routes Flask et l'utilisation de techniques AJAX ont permis d'implémenter des interactions asynchrones. Ces améliorations ont rendu l'application plus réactive et fluide.

6.3 Gestion des erreurs de compilation

L'analyse des codes soumis en C a posé des défis spécifiques. Il fallait détecter les erreurs de compilation et les formater pour les rendre claires et compréhensibles, même

pour des utilisateurs débutants. Pouvoir afficher le bon résultat par rapport avec son code.

Solution : Clang a été configuré pour retourner des messages d'erreur détaillés. Ces messages ont été formatés et testés pour garantir une interprétation claire des erreurs courantes. Plusieurs cycles de test et l'aide de ChatGPT en donnant des lignes de code ont permis de peaufiner cette fonctionnalité.

Malgré ces défis, chaque problème a été une occasion d'apprendre et de perfectionner l'application. Ces difficultés ont non seulement contribué à la maturité du projet, mais elles ont également fourni des enseignements précieux pour les futurs développements.

7 Conclusion

La réalisation de ce projet a permis de concevoir une application interactive et pédagogique dédiée à la visualisation des algorithmes de tri et à l'analyse de code utilisateur en langage C. L'objectif principal, qui était de simplifier la compréhension des mécanismes internes des algorithmes tout en offrant un environnement pratique d'expérimentation, a été atteint. Ce projet constitue une avancée significative dans l'enseignement et l'apprentissage des concepts fondamentaux de l'informatique.

Grâce à cette application, les utilisateurs peuvent :

— Visualiser étape par étape des algorithmes intégrés, tels que le tri à bulles, le tri rapide, le tri dichotomique, ect...

— Soumettre leurs propres implémentations en C, analyser leur comportement et corriger leurs erreurs grâce à des retours détaillés.

L'utilisation de technologies comme Flask, Clang, HTML, et CSS a permis de concevoir une application modulaire, performante et évolutive. Les tests approfondis ont confirmé sa robustesse, et les retours des utilisateurs ont permis d'optimiser l'expérience utilisateur.

Ce projet a également constitué une expérience formatrice pour l'équipe, offrant l'occasion d'apprendre de nouvelles technologies et d'appliquer les concepts enseignés en cours. Les défis rencontrés, bien que parfois complexes, ont renforcé les compétences techniques et collaboratives des membres du groupe.

Cependant, malgré nos efforts, il est important de reconnaître que notre projet est loin d'être complet.

8 Webographie

Références

[CAT] `savoircoder.fr/cat`
[ChatGPT] `chatgpt.com`
[Github] `github.com`
[Visual Studio Code] `code.visualstudio.com`
[MacTex] `tug.org/mactex/`
[Python] `python.org`
[Flask] `flask.palletsprojects.com`
[Jinja2] `jinja.palletsprojects.com`
[Documentation Flask] `https://flask.palletsprojects.com`
[Documentation Clang] `https://clang.llvm.org`
[Tutoriels Python et Flask sur OpenClassrooms :] `https://openclassrooms.com`

9 Annexes

Annexe .1 : Manuel Utilisateur

Manuel Utilisateur

1. **Installation des dépendances :**
   - Assurez-vous d'avoir Python installé sur votre machine.
   - Clonez le dépôt GitHub du projet :
     git clone https://github.com/kevkan75/VisualisationDeTri.git
   - Accédez au dossier du projet :
     cd VisualisationDeTri
   - Installez les dépendances à l'aide du fichier requirements.txt

2. **Exécution de l'application :**
   - Lancez le serveur Flask en mode développement :
     flask run / pyhton app.py

     http://127.0.0.1:5000

3. **Utilisation de l'interface :**
   - Page de tri : Entrez une liste d'entiers séparés par des virgules, choisissez un algorithme, puis cliquez sur "Trier". Les étapes du tri seront affichées.

   - Page d'analyse de code : Collez un code C dans la zone dédiée, puis soumettez-le pour compilation et exécution. Les résultats ou erreurs seront affichés sous la zone de saisie.

FIGURE 1 – Manuel d'utilisateur

Annexe .2 : Page de tri

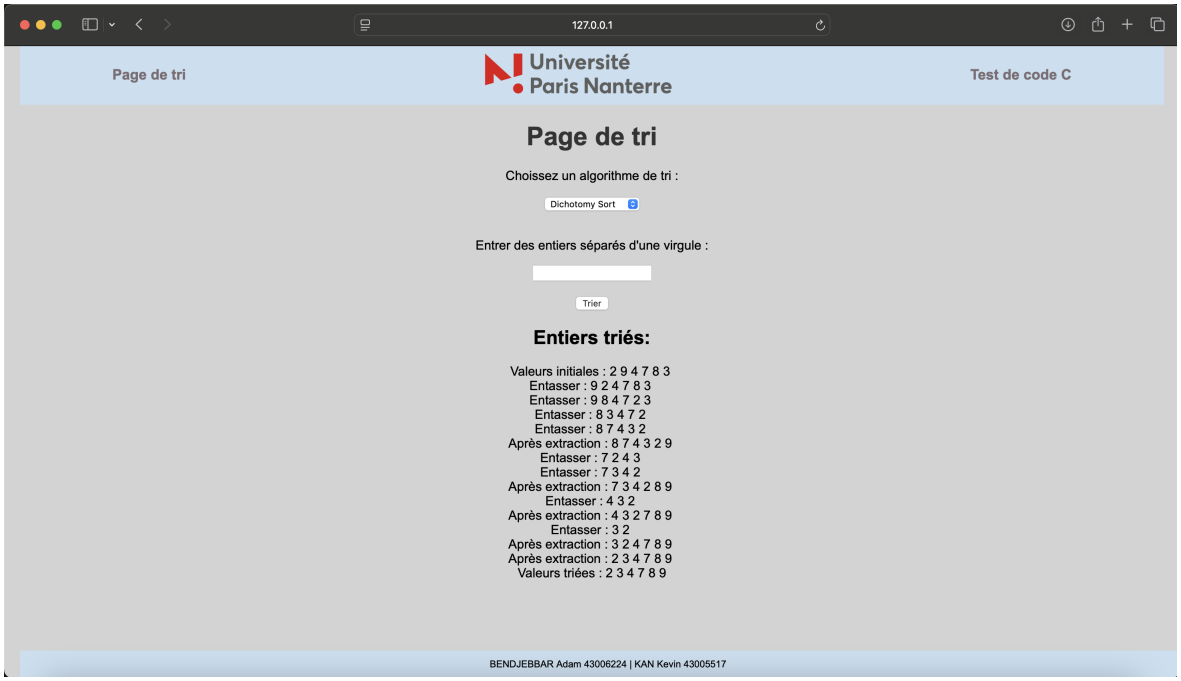

FIGURE 2 – Test de tri dichotomique

Annexe .3 : Page d'analyse de code

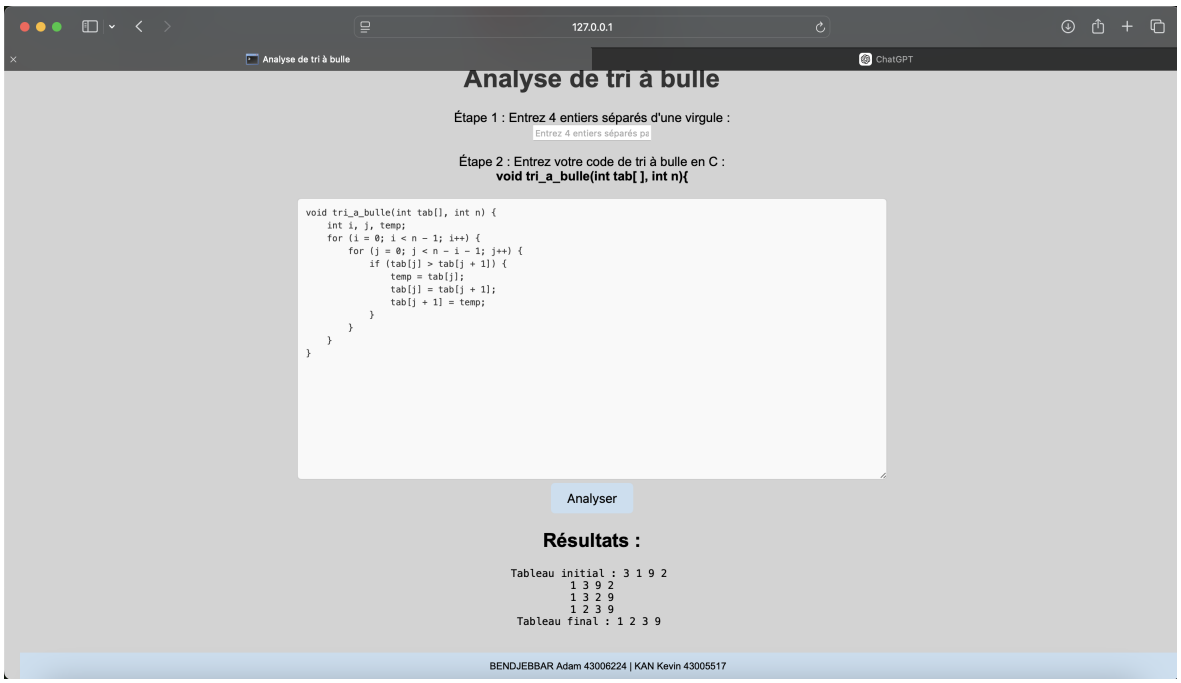

FIGURE 3 – Test de tri à bulle croissant

FIGURE 4 – Test de tri à bulle décroissant

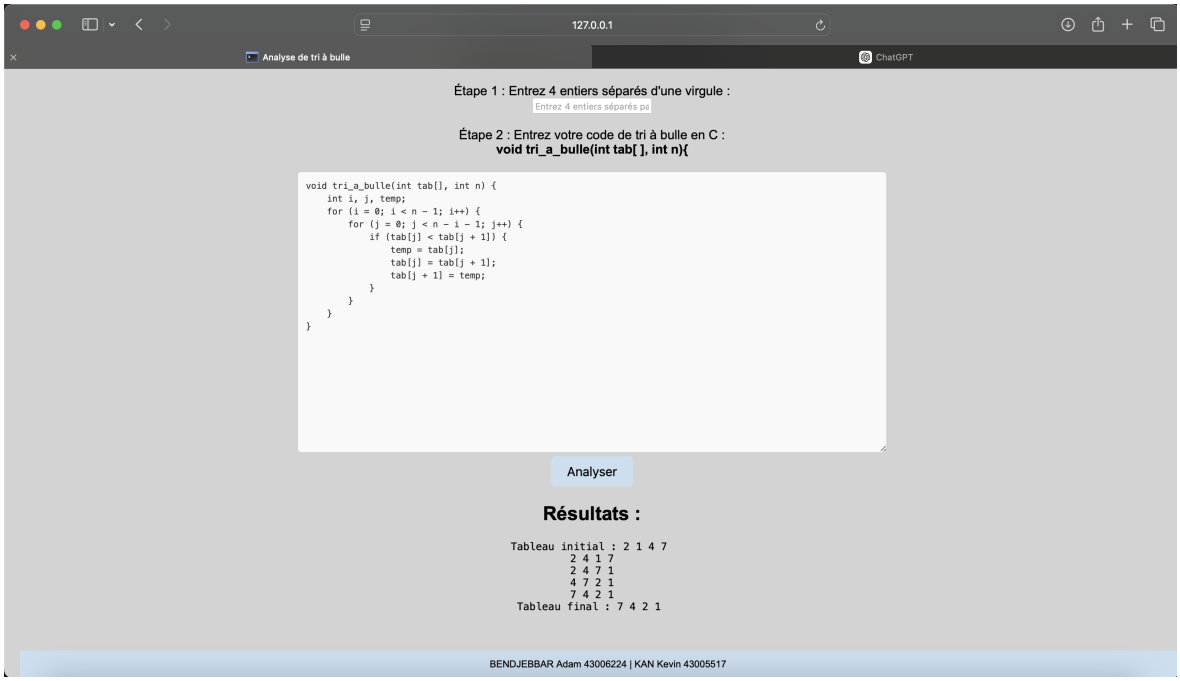

FIGURE 5 – Test de tri à bulle décroissant