

Outil d'aide à la planification des examens

Othmane Aboussaad

Naoufal Berquecha

Hajar Bougataya

Kenza Mouharrar

Xavier Zak

Contents

1	Introduction	3
2	Méthodes Utilisées	4
2.1	Programmation Linéaire	4
2.1.1	Modèle Mathématique	4
2.1.2	Fonction Objectif	4
2.1.3	Contraintes	4
2.2	Algorithmes Génétiques	5
2.2.1	Principe	5
2.2.2	Représentation d'une Solution (Génome)	5
2.2.3	Fonction d'Aptitude (Fitness Function)	5
2.3	Algorithme Tabou	6
2.3.1	Principe	6
2.3.2	Composants Principaux	6
2.3.3	Processus de l'Algorithme Tabou	6
2.4	Réseaux de Neurones	6
2.4.1	Principe	7
2.4.2	Formulation Mathématique	7
3	Cahier des Charges	8
3.1	Objectifs de la Plateforme	8
3.2	Fonctionnalités Requises	8
3.3	Contraintes Techniques	9
3.4	Attentes des Utilisateurs	9
4	Choix de l'Algorithme Tabou	10
4.1	Choix du Langage de Programmation	10
4.1.1	Comparaison avec Python	11

5	UML de l'Application	12
5.1	Description des Classes	12
5.1.1	Classe Calendrier	12
5.1.2	Classe Examen	13
5.1.3	Classe Amphitheatre	14
5.1.4	Classe AdminDashboard	14
5.1.5	Classe LoginForm	15
5.1.6	Classe StudentDashboard	15
5.1.7	Associations entre Classes	15
6	Présentation de la Plateforme	16
6.1	Interface d'AJout d'Examen	16
6.2	Tableau de Bord Administrateur	17
6.3	Suppression d'Examen	17
6.4	Génération de Planning	18
6.5	Interface de Connexion Administrateur	18
6.6	Erreur de Connexion	19
6.7	Modification d'Examen	19
6.8	Erreur de Sélection pour Modification	20
6.9	Tableau de Bord Étudiant	20
6.10	Interface de Connexion Étudiant	20
7	Références Bibliographiques	21

Chapter 1

Introduction

La planification des examens est un problème complexe qui nécessite la prise en compte de diverses contraintes et objectifs. Plusieurs méthodes peuvent être utilisées pour résoudre ce problème, incluant la programmation linéaire, les algorithmes génétiques, l'algorithme tabou, et les réseaux de neurones. Chaque méthode a ses avantages et ses limitations.

Chapter 2

Méthodes Utilisées

2.1 Programmation Linéaire

La programmation linéaire permet de modéliser le problème de planification des examens en termes de fonction objectif et de contraintes.

Responsable : Kenza Mouharrar

2.1.1 Modèle Mathématique

- Ensembles et paramètres :
- Variables de décision :

2.1.2 Fonction Objectif

Minimiser le nombre total de salles de classe attribuées pendant toute la période d'examen :

$$\min \sum_{z \in M} \sum_{j \in J} \sum_{k \in K} \sum_{g \in G} \sum_{s \in S} \sum_{l \in L} Azjkg s$$

2.1.3 Contraintes

- Un seul examen planifié dans une session pour un niveau j dans un département z :

- Les examens de niveaux différents du même département ne se déroulent pas lors de la même session :
- Capacité des salles :
- Examens de différents départements attribués à des salles différentes :
- Nombre de salles et surveillants :

2.2 Algorithmes Génétiques

Les algorithmes génétiques sont des méthodes d'optimisation basées sur les principes de la génétique et de la sélection naturelle.

Responsable : Xavier Zak

2.2.1 Principe

- Étape 1 : Sélection des meilleurs individus.
- Étape 2 : Reproduction (crossover) entre les individus sélectionnés.
- Étape 3 : Mutations des individus obtenus.

2.2.2 Représentation d'une Solution (Génome)

Une solution peut être représentée par une liste de nombres où chaque élément représente un créneau d'examen.

2.2.3 Fonction d'Aptitude (Fitness Function)

Pour évaluer les solutions :

$$fE(s) = \frac{1}{1 + \sum_{i=1}^C n_i p_i}$$

où n_i est le nombre d'occurrences de la contrainte i et p_i est la pénalité associée.

2.3 Algorithme Tabou

L'algorithme tabou est une méthode d'optimisation heuristique efficace pour résoudre des problèmes combinatoires complexes.

Responsables : Hajar Bougataya et Naoufal Berquecha

2.3.1 Principe

L'algorithme tabou parcourt l'espace des solutions en évitant les cycles et en explorant de nouvelles régions grâce à une mémoire adaptative appelée "liste tabou".

2.3.2 Composants Principaux

- Solution initiale
- Liste tabou
- Fonction d'évaluation
- Mouvements

2.3.3 Processus de l'Algorithme Tabou

- Initialisation
- Boucle Principale
- Retourner la Meilleure Solution

2.4 Réseaux de Neurones

Les réseaux de neurones peuvent être utilisés pour modéliser et résoudre le problème de planification d'examens.

Responsable : Othmane Aboussaad

2.4.1 Principe

Un réseau de neurones artificiels est composé de neurones organisés en couches. Chaque neurone reçoit des entrées, applique une fonction d'activation et produit une sortie.

2.4.2 Formulation Mathématique

- Neurone :

$$y = f(w \cdot x + b)$$

- Propagation vers l'avant :

$$a^{(l+1)} = f(W^{(l)}a^{(l)} + b^{(l)})$$

- Fonction de coût :

$$J = \frac{1}{2m} \sum_{i=1}^m (h(x^{(i)}) - y^{(i)})^2$$

Chapter 3

Cahier des Charges

3.1 Objectifs de la Plateforme

- Automatiser la planification des examens en prenant en compte les contraintes des étudiants et des enseignants.
- Minimiser les conflits d'examens pour les étudiants.
- Utiliser de manière optimale les amphithéâtres disponibles.

3.2 Fonctionnalités Requises

- **Gestion des Examens :**
 - Ajouter, modifier et supprimer des examens.
 - Visualiser la liste des examens planifiés.
- **Génération Automatique de Planning :**
 - Générer un planning d'examens en fonction des contraintes définies.
- **Interface Utilisateur :**
 - Interface intuitive pour les administrateurs et les étudiants.
 - Notifications des conflits de planning et autres erreurs.

3.3 Contraintes Techniques

- **Sécurité** : Gestion sécurisée des données utilisateurs et des informations sur les examens.
- **Performance** : L'algorithme doit générer les plannings en un temps raisonnable, même pour un grand nombre d'examens et d'amphithéâtres.

3.4 Attentes des Utilisateurs

- **Facilité d'Utilisation** : Interface utilisateur simple et claire.
- **Fiabilité** : Le système doit fonctionner sans erreurs et gérer correctement les cas d'exception.
- **Flexibilité** : Capacité à ajuster les paramètres de planification et à modifier les données sans complexité excessive.

Chapter 4

Choix de l'Algorithme Tabou

Après une analyse approfondie des différentes méthodes, nous avons choisi d'implémenter l'application avec l'algorithme tabou. Cette décision repose sur plusieurs facteurs :

- **Efficacité** : L'algorithme tabou est particulièrement efficace pour les problèmes combinatoires complexes.
- **Flexibilité** : Il permet une exploration approfondie de l'espace de solutions.
- **Adaptabilité** : La liste tabou permet d'éviter les cycles et d'explorer de nouvelles solutions.

4.1 Choix du Langage de Programmation

Nous avons opté pour Java pour les raisons suivantes :

- **Avantages** :
 - Portabilité : Java est indépendant de la plateforme, ce qui permet une exécution sur diverses machines sans modification du code source.
 - Robustesse : Gestion automatique de la mémoire et un système de gestion des exceptions contribuent à la stabilité du programme.
 - Large Écosystème : Nombreuses bibliothèques et frameworks disponibles facilitent le développement rapide et la maintenance.

- **Inconvénients :**

- Verbose : Le code Java peut être plus verbeux comparé à d'autres langages, ce qui peut augmenter le temps de développement.
- Performances : Java peut être moins performant que des langages compilés comme C++ en termes de vitesse d'exécution.

4.1.1 Comparaison avec Python

Nous avons également envisagé Python pour son interactivité et sa simplicité. Voici une comparaison :

- **Python :**

- Avantages : Syntaxe concise et facile à lire, large écosystème de bibliothèques, idéal pour le prototypage rapide.
- Inconvénients : Moins performant que Java en termes de vitesse d'exécution, gestion de la mémoire moins robuste.

- **Java :**

- Avantages : Portabilité, robustesse, performance supérieure pour les applications à grande échelle.
- Inconvénients : Verbose, temps de compilation plus long.

Finalement, nous avons préféré Java pour sa robustesse et sa performance dans un environnement de production.

Chapter 5

UML de l'Application

Le diagramme UML de l'application illustre les différentes classes et leurs interactions. Voici une description complète des principales classes et des choix de conception :

5.1 Description des Classes

5.1.1 Classe Calendrier

- **Attributs :**
 - `examens` : Liste des examens.
 - `amphitheatres` : Liste des amphithéâtres.
- **Méthodes :**
 - `getInstance()` : Retourne l'instance unique de la classe (Singleton).
 - `addExamen(Examen examen)` : Ajoute un examen à la liste.
 - `modifierExamen(Examen examen, String nom, int duree, String date, String heure, Amphitheatre amphitheatre)` : Modifie un examen existant.
 - `supprimerExamen(String nom, String classe)` : Supprime un examen de la liste.

- `generatePlanning(LocalDate startDate, LocalDate endDate)`
: Génère le planning des examens.
- `getAllExamens()` : Retourne la liste de tous les examens.
- `getAmphitheatreByName(String name)` : Retourne l’amphithéâtre correspondant au nom donné.

5.1.2 Classe Examen

- **Attributs :**

- `classe` : La classe à laquelle l’examen est destiné.
- `nom` : Nom de l’examen.
- `duree` : Durée de l’examen.
- `date` : Date de l’examen.
- `heure` : Heure de l’examen.
- `amphitheatre` : Amphithéâtre où l’examen a lieu.

- **Méthodes :**

- `getClasse()` : Retourne la classe de l’examen.
- `getNom()` : Retourne le nom de l’examen.
- `getDuree()` : Retourne la durée de l’examen.
- `getDate()` : Retourne la date de l’examen.
- `getHeure()` : Retourne l’heure de l’examen.
- `getAmphitheatre()` : Retourne l’amphithéâtre de l’examen.
- `setClasse(String classe)` : Définit la classe de l’examen.
- `setNom(String nom)` : Définit le nom de l’examen.
- `setDuree(int duree)` : Définit la durée de l’examen.
- `setDate(String date)` : Définit la date de l’examen.
- `setHeure(String heure)` : Définit l’heure de l’examen.
- `setAmphitheatre(Amphitheatre amphitheatre)` : Définit l’amphithéâtre de l’examen.

5.1.3 Classe Amphitheatre

- **Attributs :**

- `nom` : Nom de l’amphithéâtre.
- `capacite` : Capacité de l’amphithéâtre.

- **Méthodes :**

- `getNom()` : Retourne le nom de l’amphithéâtre.
- `getCapacite()` : Retourne la capacité de l’amphithéâtre.
- `setNom(String nom)` : Définit le nom de l’amphithéâtre.
- `setCapacite(int capacite)` : Définit la capacité de l’amphithéâtre.

5.1.4 Classe AdminDashboard

- **Attributs :**

- `calendrier` : Référence à l’instance du calendrier.
- `tableView` : TableView pour afficher les examens.
- `examensList` : Liste observable des examens pour la table.

- **Méthodes :**

- `showAddExamenDialog()` : Affiche la boîte de dialogue pour ajouter un examen.
- `showModifyExamenDialog()` : Affiche la boîte de dialogue pour modifier un examen.
- `showDeleteExamenDialog()` : Affiche la boîte de dialogue pour supprimer un examen.
- `showGeneratePlanningDialog()` : Affiche la boîte de dialogue pour générer le planning.

5.1.5 Classe LoginForm

- **Attributs :**
 - `usernameField` : Champ de texte pour le nom d'utilisateur.
 - `passwordField` : Champ de texte pour le mot de passe.
- **Méthodes :**
 - `start()` : Démarre l'application de connexion.
 - `login()` : Valide les informations de connexion.

5.1.6 Classe StudentDashboard

- **Attributs :**
 - `studentId` : Identifiant de l'étudiant.
 - `tableView` : TableView pour afficher les examens.
 - `examensList` : Liste observable des examens pour la table.
- **Méthodes :**
 - `initialize()` : Initialise le tableau de bord de l'étudiant avec ses examens.
 - `logout()` : Déconnecte l'étudiant et revient à l'écran de connexion.

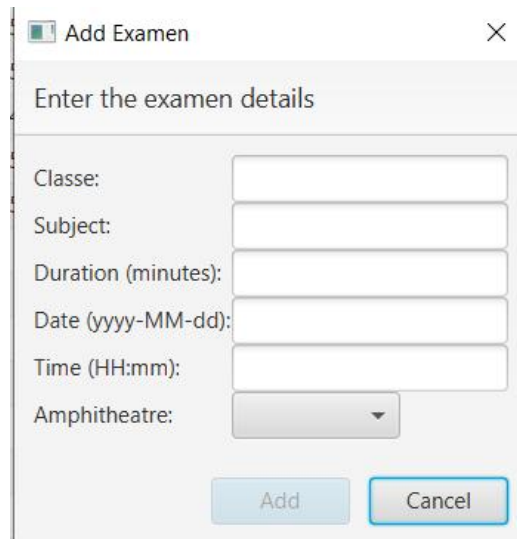
5.1.7 Associations entre Classes

- **Calendrier — 1..* — Examen** : Un calendrier peut contenir plusieurs examens.
- **Examen — 1 — Amphitheatre** : Un examen est associé à un amphithéâtre.
- **AdminDashboard — 1 — Calendrier** : Le tableau de bord admin interagit avec le calendrier pour gérer les examens.
- **StudentDashboard — 1 — Calendrier** : Le tableau de bord étudiant affiche les examens du calendrier.

Chapter 6

Présentation de la Plateforme

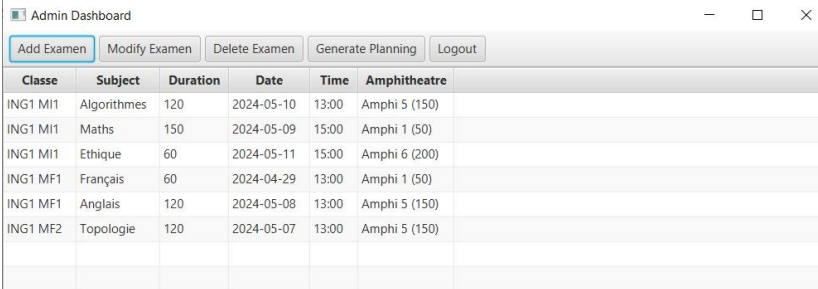
6.1 Interface d'Ajout d'Examen



The image shows a software dialog box titled "Add Examen" with a close button (X) in the top right corner. Below the title bar is a header area with the text "Enter the examen details". The main area contains six input fields, each with a label to its left: "Classe:", "Subject:", "Duration (minutes):", "Date (yyyy-MM-dd):", "Time (HH:mm):", and "Amphitheatre:". The first five fields are text boxes, and the last one is a dropdown menu. At the bottom right of the dialog are two buttons: "Add" and "Cancel".

Figure 6.1: Interface d'ajout d'examen

6.2 Tableau de Bord Administrateur



The Admin Dashboard window features a title bar with standard window controls. Below the title bar is a toolbar with five buttons: 'Add Examen' (highlighted in blue), 'Modify Examen', 'Delete Examen', 'Generate Planning', and 'Logout'. The main area contains a table with the following data:

Classe	Subject	Duration	Date	Time	Amphitheatre
ING1 MI1	Algorithmes	120	2024-05-10	13:00	Amphi 5 (150)
ING1 MI1	Maths	150	2024-05-09	15:00	Amphi 1 (50)
ING1 MI1	Ethique	60	2024-05-11	15:00	Amphi 6 (200)
ING1 MF1	Français	60	2024-04-29	13:00	Amphi 1 (50)
ING1 MF1	Anglais	120	2024-05-08	13:00	Amphi 5 (150)
ING1 MF2	Topologie	120	2024-05-07	13:00	Amphi 5 (150)

Figure 6.2: Tableau de bord administrateur

6.3 Suppression d'Examen

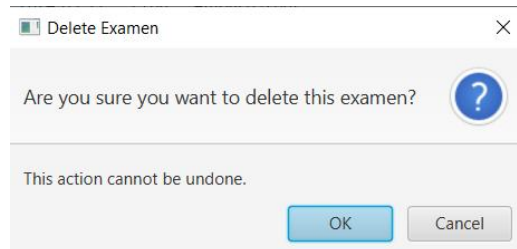
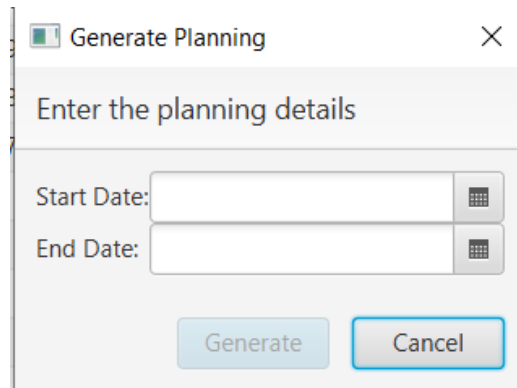


Figure 6.3: Confirmation de suppression d'examen

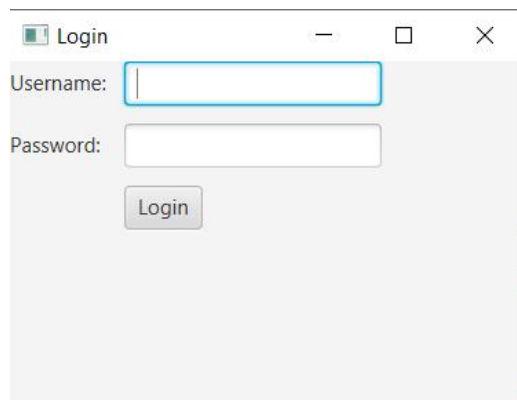
6.4 Génération de Planning



A dialog box titled "Generate Planning" with a close button (X) in the top right corner. Below the title bar is a header area with the text "Enter the planning details". The main area contains two input fields: "Start Date:" and "End Date:", each followed by a small calendar icon. At the bottom, there are two buttons: "Generate" and "Cancel".

Figure 6.4: Interface de génération de planning

6.5 Interface de Connexion Administrateur



A dialog box titled "Login" with standard window controls (minimize, maximize, close) in the top right corner. The main area contains two input fields: "Username:" and "Password:". Below the "Password:" field is a "Login" button.

Figure 6.5: Interface de connexion administrateur

6.6 Erreur de Connexion

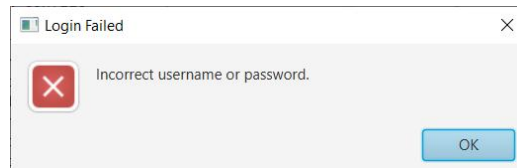


Figure 6.6: Erreur de connexion

6.7 Modification d'Examen

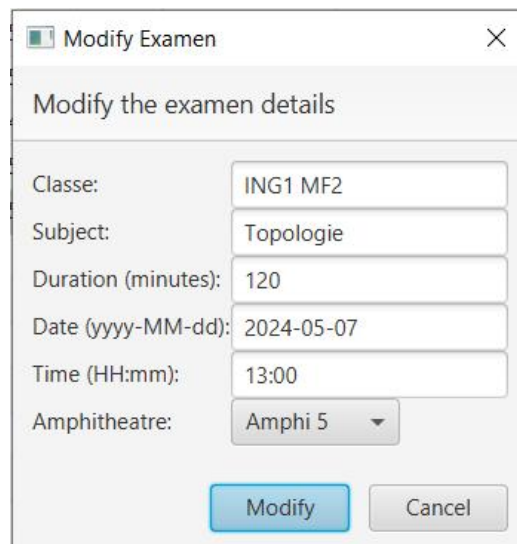
A dialog box titled "Modify Examen" with a close button (X) in the top right corner. Below the title bar is a header "Modify the examen details". The main area contains several input fields: "Classe:" with a text box containing "ING1 MF2", "Subject:" with a text box containing "Topologie", "Duration (minutes):" with a text box containing "120", "Date (yyyy-MM-dd):" with a text box containing "2024-05-07", "Time (HH:mm):" with a text box containing "13:00", and "Amphitheatre:" with a dropdown menu showing "Amphi 5". At the bottom, there are two buttons: a blue "Modify" button and a grey "Cancel" button.

Figure 6.7: Interface de modification d'examen

6.8 Erreur de Sélection pour Modification

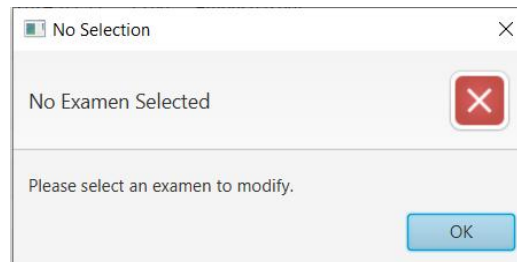


Figure 6.8: Erreur de sélection d'examen à modifier

6.9 Tableau de Bord Étudiant

A screenshot of a window titled "Student Dashboard - ING1 M11". It has a "Logout" button and a table with the following data:

Subject	Duration	Date	Time	Amphitheatre
Algorithmes	120	2024-04-28	15:00	Amphi 4
Maths	150	2024-04-28	11:00	Amphi 4
Ethique	60	2024-04-28	10:00	Amphi 4

Figure 6.9: Tableau de bord étudiant

6.10 Interface de Connexion Étudiant

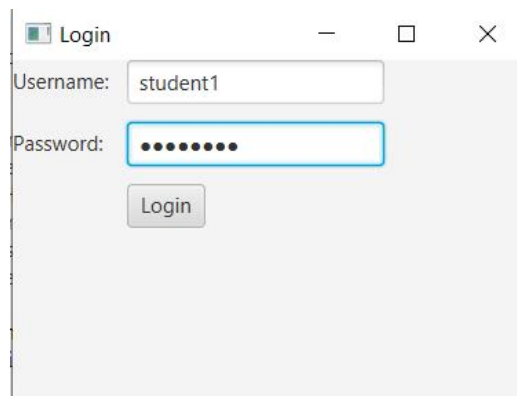


Figure 6.10: Interface de connexion étudiant

Chapter 7

Références Bibliographiques

- 1 Moalic, L. (2018). *Coloration de graphe - méthodes et applications*. Académie de Strasbourg. https://pedagogie.ac-strasbourg.fr/fileadmin/pedagogie/isn/ISN/Conferences/cours_isn_Coloration_graphe.pdf
- 2 Corne, D., Fang, H. L., Mellish, C. (1994). *Solving the modular exam scheduling problem with genetic algorithms*. https://www.researchgate.net/publication/239665421_Solving_the_Modular_Exam_Scheduling_Problem_with_Genetic_Algorithms
- 3 Alvarez-Valdés, R., Crespo, E., Tamarit, J. M. (1997). *A tabu search algorithm to schedule university examinations*. *Qüestiió*, 21(1-2), 201-215. <https://www.idescat.cat/sort/questiio/questiiopdf/21.1.8.AlvarezValdes.pdf>
- 4 Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation*. https://cdn.preterhuman.net/texts/science_and_technology/artificial_intelligence/Neural%20Networks%20-%20A%20Comprehensive%20Foundation%20-%20Simon%20Haykin.pdf