



♦ CAHIER D'EXAMEN ♦

Génie Informatique et Logiciel

ÉPREUVE EST PRÉSENTÉ PAR:

EVIDENT

Introduction

Bienvenue à l'épreuve de Génie Informatique et Logiciel des Jeux de Génie 2026 🎉

Durant cet examen, vous serez plongés dans un univers où les rêves deviennent réalité grâce à vos compétences techniques et votre esprit d'équipe. Vous formez ensemble une équipe de consultation de 7 personnes travaillant pour la compagnie fictive **Nocturna Solutions**.

Votre mission : répondre aux mandats confiés par différents clients. Chaque client représente un domaine particulier de l'ingénierie logicielle et informatique – de l'architecture logicielle à la sécurité, en passant par les bases de données, l'UX, l'IA, les algorithmes et plus! Pour chacun de ces mandats, vos réponses devront démontrer autant votre compréhension théorique que votre capacité à appliquer vos connaissances.

L'examen est d'une durée totale de 3h30. Vous êtes invités à gérer votre temps stratégiquement afin de couvrir le plus de mandats possibles.

En plus de ce feuillet papier (qui contient les mises en contexte ainsi que les espaces pour répondre aux questions, vous avez accès à un [repo GitHub](#)). Ce repo contient toutes les questions qui se résolvent à travers du code. Les consignes pour la remise des travaux devraient avoir été précisées le jour de l'examen.

Bonne chance à tout le monde ! Et n'oubliez pas, ce n'est pas juste en rêvant qu'on fait avancer la technologie, c'est en mettant nos compétences à l'œuvre pour de vrai 😊

	Points possibles	Points obtenus
Mandat 1 – Architecture	13	
Mandat 2 – Database	23	
Mandat 3 – Algorithmes	32	
Mandat 4 – Sécurité	20	
Mandat 5 – Devops	12	
Mandat 6 – Backend	14	
Mandat 7 – Systèmes	11	
Mandat 8 – UI/UX	14	
Mandat 9 – Logique numérique	13	
Mandat 10 – AI	12	
Mandat 11 – Créativité Informatique	28	
Total	192	

Mandat 1 – Architecture - Rêva Construction (13 points)

Le client **Rêva Construction** souhaite bâtir des systèmes logiciels aussi solides que leurs infrastructures physiques. Ils vous consultent aujourd’hui afin d’obtenir une architecture claire, modulaire et capable de soutenir leurs projets à long terme.

Rêva Construction veut développer une application qui permet de concevoir et commander des lits et des matelas sur mesure pour ses clients. L’application est une plateforme web accessible sur ordinateur et mobile, qui guide les utilisateurs à travers un processus de configuration personnalisé. Les clients peuvent se créer un profil, remplir un formulaire détaillé sur leurs préférences de sommeil et explorer différentes options de lits, matériaux et accessoires.

L’application utilise un **algorithme complexe** pour calculer automatiquement le meilleur lit adapté à chaque utilisateur, en prenant en compte ses préférences et contraintes spécifiques. Elle génère ensuite une prévisualisation interactive du lit choisi et permet de passer la commande directement en ligne via une interface d’achat intégrée. Du côté de Rêva Construction, l’équipe doit pouvoir gérer les modèles proposés, suivre les commandes et consulter des statistiques sur les ventes pour améliorer le service.

1. Architecture modulaire (3 points)

Expliquer les principes clés d’une *architecture logicielle d’infonuagique* et justifiez pourquoi ils sont essentiels pour répondre aux besoins de Rêva Construction.

2. Stack technologique (3 points)

Proposez un stack technologique complet pour développer cette application, en incluant le choix du frontend, du backend, de la base de données, ainsi que tout outil ou librairie nécessaire pour gérer la logique métier et l’algorithme complexe. Justifiez vos choix en fonction des besoins de l’application et de la qualité globale du produit.

3. Architecture logicielle (4 points)

Proposer une architecture logicielle adaptée à Rêva Construction (par exemple : microservices, monolithique, architecture en couches, etc.), détailler ses composants et expliquez vos choix. Fournissez le diagramme de déploiement de votre solution.

4. Qualité et pérennité (3 points)

Identifiez 4 outils, pratiques ou méthodologies pour garantir la qualité du code, la maintenabilité et la pérennité du système à long terme ? Justifiez vos réponses.

Mandat 2 – Database - Somnia Data (23 points)

Somnia Data est une entreprise spécialisée dans la gestion et l'analyse de données, avec pour slogan : « On garde vos rêves en mémoire ». Elle souhaite développer une plateforme web permettant aux utilisateurs de stocker leurs profils de sommeil, recevoir des recommandations personnalisées et suivre des plans d'intervention visant à améliorer la qualité de leur sommeil.

Chaque utilisateur doit avoir un compte sur la plateforme avant de pouvoir accéder aux services. Un utilisateur est caractérisé par un identifiant unique, un prénom, un nom, une adresse (rue, ville, code postal) et un courriel. Les utilisateurs peuvent enregistrer plusieurs profils de sommeil, chacun comprenant des préférences détaillées (type de matelas, fermeté, température, position de sommeil, habitudes nocturnes).

Pour chaque profil, la plateforme peut générer plusieurs plans d'intervention personnalisés, identifiés par un numéro unique et une date de création. Chaque plan d'intervention peut contenir plusieurs recommandations (exercices de relaxation, ajustements d'habitudes, horaires de sommeil), une description et un statut indiquant s'il est actif, terminé ou en attente.

Pour l'administration et l'optimisation des performances, la plateforme garde une trace des logs et statistiques sur les profils, plans d'intervention et recommandations. Chaque log est identifié par un identifiant unique et contient l'action réalisée, une description et la date. Les logs sont accédés très rarement et servent uniquement à des fins de débogage.

Vous êtes chargé de modéliser une base de données destinée à gérer le fonctionnement de la plateforme Somnia Data. Vous devrez identifier toutes les entités, leurs attributs, les associations entre elles, les cardinalités et les types d'associations (binaire 1:n, n:m, etc.).

* Certaines questions nécessitent de vous appuyer sur vos réponses aux questions 1 et 2. Si vos réponses diffèrent de celles du solutionnaire, nous effectuerons la correction en fonction de vos éléments de réponse. Si jamais vous vous sentez incapable de répondre aux première questions, on peut vous fournir les réponses, évidemment cela résulte en aucun point pour les 2 premières questions

1. Diagramme Entité-Association (6 points)

Dessinez un diagramme Entité-Association UML ou un schéma relationnel représentant la base de données nécessaire pour l'application Somnia Data. Ne vous en faites pas avec les conventions particulières du langage de représentation que vous avez apprises. L'important c'est que vous identifiez toutes les entités, leur clée, leurs attributs, les associations et leur type (binaire, 1:n, n:m, etc.).

2. Schéma SQL (4 points)

En vous aidant de votre Diagramme Entité-Association, rédigez le schéma SQL dans le fichier SCHEMA.sql du Mandat 2 - Somnia Data dans le repo Github. Utilisez le langage MySQL si vous pouvez.

3. Requêtes SQL (5 points)

Pour le schéma que vous avez proposé à la question 2, écrivez les requêtes SQL dans les fichiers correspondants REQUEST_X.sql du mandat 2 - Somnia Data dans le repo Github. Utilisez le langage MySQL si vous pouvez. Voici ce que vous devez aller chercher avec vos requêtes.

- a. Récupérer la liste des utilisateurs qui ont plus de 3 plans d'intervention actifs.
- b. Récupérer la liste de tous les profils d'un utilisateur donné (id utilisateur = 42) avec son nombre de plans d'intervention en attente.
- c. Récupérer toutes les recommandations non actives pour un plan donné (id plan = 30).
- d. Récupérer l'historique des actions dans les logs pour un utilisateur donné (id utilisateur = 42) sur ses plans et recommandations, triés en ordre décroissant selon leur date de création.
- e. Récupérer tous les plans d'intervention avec leur profil et utilisateur associé créés entre le 1er décembre et le 1er janvier, triées en ordre chronologique selon la date de création.

4. Indexation et performance (3 points)

Pour le schéma que vous avez proposé à la question 2, indiquez où vous ajouterez des index pour améliorer la performance des requêtes. Justifiez chaque index : pourquoi est-il nécessaire, sur quelle(s) colonne(s) et pour quel type de requête. Expliquez également les avantages et les inconvénients de manière générale de l'utilisation des indexes en SQL.

5. Base de données orientée objet (2 points)

Expliquez **les principales différences** entre une base de données relationnelle (SQL) et une base orientée objets/documents (NoSQL).

6. Réflexion finale (3 points)

Selon vous, quel type de base de données (SQL) ou (NoSQL) est mieux appliquée aux besoins de Somnia Data? Justifiez votre réponse avec les points forts/faibles que vous avez identifiés à la question 5.

Mandat 3 – Algorithmes - Algoria Labs (32 points)

Algoria Labs est une entreprise de recherche spécialisée dans la conception d'algorithmes capables de transformer les rêves en solutions concrètes. Leur terrain de jeu : optimiser, analyser et résoudre des problèmes complexes où logique et créativité se rencontrent.

Votre équipe est mandatée pour relever une série de défis algorithmiques inspirés du monde du sommeil et des songes.

1. Dream Navigation (6 points)

Le département *Dream Navigation™* d'Algoria Labs étudie les mécanismes de déplacement dans les rêves lucides. Leurs recherches portent sur la navigation optimale dans des environnements oniriques où les lois physiques traditionnelles ne s'appliquent pas toujours. Votre objectif est de développer un algorithme de recherche de chemin intelligent pour naviguer dans des labyrinthes oniriques complexes contenant des portails de téléportation, zones d'accélération et obstacles temporels. Le code à compléter ainsi que les instructions se trouvent [dans le repo Github](#).

2. To Infinity And Beyond (6 points)

Le département *To Infinity And Beyond™* d'Algoria Labs veut empiler des blocs de différentes dimensions afin de former des tours les plus hauts possibles, dans l'ambition d'atteindre les étoiles. Votre objectif est de développer un algorithme d'optimisation intelligent pour maximiser la hauteur d'empilage de blocs tout en respectant les contraintes de résistance, dimensions et fragilité. Le code à compléter ainsi que les instructions se trouvent [dans le repo Github](#).

3. Tic Tac Throw! (7 points)

Le département *Idiot Factory™* d'Algoria Labs s'intéresse à une problématique très particulière : développer des agents « intelligents »... qui perdent toujours. Votre objectif sera d'implémenter un agent artificiel qui joue à une version modifiée 4x4 de Tic Tac Toe de façon systématiquement sous-optimale, afin de maximiser ses chances de perdre contre divers types d'adversaires. Le code à compléter ainsi que les instructions se trouvent [dans le repo Github](#).

4. Complexité asymptotique (3 points)

Pour chacun des extraits de code suivants, évaluez la complexité asymptotique en pire cas. Indiquez, pour chaque fonction, la complexité temporelle et la complexité spatiale, en utilisant la notation en grand O.

```
def func1(n):
    total = 0
    i = 1
    while i < n:
        j = i
        while j > 0:
            total += 1
            j //= 2
        i *= 2
    return total
```

Complexité temporelle:

Complexité spatiale:

```
def func2(n):
    if n <= 1:
        return 1
    return func2(n - 1) + func2(n // 2) + func2(1)
```

Complexité temporelle:

Complexité spatiale:

```
def func3(n):
    arr = [0] * n
    for i in range(n):
        j = i
        while j < n:
            arr[j] += 1
            j += (j - i + 1)
```

Complexité temporelle:

Complexité spatiale:

```
def func4(n):
    if n <= 0:
        return 0
    if n == 1:
        return 1
    return func4(n - 1) + func4(n - 1)
```

Complexité temporelle:

Complexité spatiale:

```
def func5(n):
    s = set()
    for i in range(n):
        for j in range(i):
            s.add((i, j))
    for k in s:
        x = k[0] * k[1]
```

Complexité temporelle:

Complexité spatiale:

5. Structures de données appliquées (5 points)

Pour chacun des scénarios suivants déterminez quelle structure de données (Heap, Stack, Tree, Linked List, etc.) est la mieux appliquée au problème. Justifiez votre réponse en termes de complexité spatiale et temporelle des opérations de la structure choisie et comment cela s'applique aux besoins du problème.

- a. On veut gérer un stream en temps réel d'entiers qui représentent des sessions IDs actifs sur un serveur.
 - À chaque étape, on doit insérer un nouvel ID, checker si un ID est actif, et retirer les IDs expirés.
 - Les opérations doivent être rapides.

- b. On veut analyser un stream massif de nombres entrants.
 - On veut pouvoir garder les K plus gros éléments vus jusqu'à présent,
 - On veut pouvoir récupérer le minimum parmi ces K à tout moment
 - On veut mettre à jour la structure efficacement quand un nouveau nombre arrive.

- c. On veut coder une feature d'autocomplete dans une app de messagerie.
 - À partir d'un préfixe, tu dois retourner tous les mots stockés qui commencent par ce préfixe.
 - Le dictionnaire contient des millions de mots.
 - Les queries doivent être rapides et la mémoire bien optimisée.

- d. On veut gérer une séquence de nombres qui peut changer dans le temps.
- On doit pouvoir faire des requêtes sur le minimum dans un range $[l, r]$ plusieurs fois.
 - Les requêtes et les mises à jour doivent être rapides.

- e. On veut développer un système de routing pour un réseau de transport.
- Le graphe est sparse, pondéré, avec des poids non-négatifs.
 - On doit pouvoir trouver le plus court chemin entre deux villes plusieurs fois par seconde.

6. Ensembles et relations (2 points)

Soit la relation R définie sur l'ensemble $\{1,2,3,4\}$ par :

$$R = \{(1,1), (2,2), (3,3), (4,4), (1,2), (2,3), (3,4)\}$$

Déterminez quelles propriétés la relation R possède :

- a) Réflexive, symétrique, transitive, et c'est une relation d'ordre partiel
- b) Réflexive et transitive, mais pas symétrique ; c'est une relation d'ordre partiel
- c) Symétrique et transitive, mais pas réflexive ; ce n'est pas une relation d'ordre partiel
- d) Réflexive seulement, ni symétrique ni transitive ; ce n'est pas une relation d'ordre partiel

7. Graphes (2 points)

Soit le graphe G défini par les sommets $V = \{A, B, C, D, E\}$ et les arêtes $E = \{\{A,B\}, \{A,C\}, \{B,C\}, \{B,D\}, \{C,E\}\}$.

Parmi les affirmations suivantes, laquelle décrit correctement les propriétés de ce graphe ?

- a) Le graphe est connexe, contient un cycle, et n'est pas biparti
- b) Le graphe est connexe, ne contient pas de cycle, et est biparti
- c) Le graphe n'est pas connexe, contient un cycle, et est biparti
- d) Le graphe n'est pas connexe, ne contient pas de cycle, et n'est pas biparti

8. Arbres (1 points)

Soit un arbre binaire complet de hauteur $h = 4$.

Quelle affirmation décrit correctement le nombre maximal de sommets (ou nœuds) et de feuilles dans cet arbre ?

- a) Maximum de 15 sommets et 7 feuilles
- b) Maximum de 31 sommets et 16 feuilles
- c) Maximum de 16 sommets et 8 feuilles
- d) Maximum de 30 sommets et 15 feuilles

Mandat 4 – Sécurité - Nightmare Reaper (20 points)

Nightmare Reaper est une entreprise de cybersécurité spécialisée dans les tests Red Team. Son objectif est d'identifier et d'exploiter différentes vulnérabilités afin de démontrer à ses clients et partenaires les risques associés et de montrer la méthodologie.

1. Capture the Flag (3x3 points)

Votre objectif est d'identifier plusieurs *flags* cachés (au format **FLAGXYZ**), chacun représentant une vulnérabilité distincte. Le code à investiguer ainsi que les instructions se trouvent dans [le repo Github](#).

Pour chaque flag, décrivez de façon structurée et reproductible : les étapes chronologiques et actions précises ; les outils, paramètres et techniques utilisés (ex. scan, brute-force, injection, escalade) ; et justifiez vos réponses.

2. Stéganographie (3x3 points)

Votre objectif est d'identifier plusieurs *flags* cachés (au format **FLAGXYZ**), chacun représentant une façon de dissimuler des informations dans des images, textes etc. Les fichiers ainsi que les instructions se trouvent dans [le repo Github](#).

Pour chaque flag, décrivez de façon structurée et reproductible : les étapes chronologiques et actions précises ; les outils, paramètres et techniques utilisés (ex. scan, brute-force, injection, escalade) ; et justifiez vos réponses.

3. Mot de passe (2 points)

Un mot de passe est une suite de 5 caractères choisis parmi les 26 lettres majuscules de l'alphabet anglais et les 10 chiffres (0-9), soit 36 caractères possibles.

Sélectionnez l'affirmation qui est vrai concernant le nombre de mots de passe possibles dans les 3 situations suivantes:

- Sans restriction
 - Au moins 2 caractères distincts
 - Tous les caractères distincts
- a) - Sans restriction : 60 466 176
- 2 lettres distinctes : 6500
- Tous les caractères distincts: 45239040
- b) - Sans restriction : 60 466 176
- Au moins 2 caractères distincts: 60 466 140
- Tous les caractères distincts: 45239040
- c) - Sans restriction : 60 466 176
- Au moins 2 caractères distincts: 1,413,720;
- Tous les caractères distincts : 60 466 176
- d) - Sans restriction : 67 523 422
- Au moins 2 caractères distincts: $10 \times C(5,2)$
- Tous les caractères distincts : 60 466 176

Mandat 5 – Devops - Oléoducs Scripts (12 points)

Oléoducs Scripts est une entreprise de développement des opérations avec une expertise dans les technologies Docker et leur ordonnancement.

Vous serez chargés de corriger / mettre en place un environnement Docker qui gère son networking avec différentes applications qui roulent dans les containers.

1. Docker config (5 points)

Faites fonctionner l'environnement avec des valeurs "hardcode" (url fixes, port fixes, nombre de conteneurs fixes etc.). Le code à compléter ainsi que les instructions se trouvent [dans le repo Github](#).

2. Scale up (5 points)

Ajout des fonctionnalités de «Scale up» avec leurs scripts associés (nombre de "slave" pouvant être changés avec 1 seul paramètre et qui peuvent être contactés par le load-balancer. Le code à compléter ainsi que les instructions se trouvent [dans le repo Github](#).

3. Déploiement blue green (2 points)

Considérez un cluster Kubernetes exécutant un microservice sans état. Un développeur a implémenté une mise à jour progressive en utilisant la configuration suivante:

```
spec:  
  strategy:  
    type: RollingUpdate  
    rollingUpdate:  
      maxSurge: 25%  
      maxUnavailable: 0
```

Laquelle des situations suivantes ne serait pas une utilisation valide de cette configuration? Encerclez votre réponse et justifiez pourquoi.

- a) Mise à jour progressive d'un service ayant un grand nombre d'utilisateurs tout en maintenant une disponibilité totale.
- b) Tester une nouvelle fonctionnalité sur un sous-ensemble d'instances avant de déployer à l'ensemble des instances.
- c) S'assurer d'aucune interruption lors d'une mise à jour pendant les heures critiques de l'entreprise.
- d) Remplacer une version défectueuse du service avec une version corrigée.

Mandat 6 – Backend - Evening Dream (14 points)

Evening Dream est une jeune entreprise de développement logiciel qui se spécialise dans l'art de la brosse (Organiser des soirées bien arrosées entre amis). Votre équipe est mandatée pour concevoir et mettre en place un serveur backend capable de supporter une application qui aidera les utilisateurs à planifier leur soirée.

1. Serveur Backend (8 points)

Rédigez un serveur qui puisse écrire la planification générale d'une brosse. Le serveur devra être capable de communiquer avec différents API externes pour avoir des informations de comment construire cette soirée. Le code à compléter ainsi que les instructions se trouvent [dans le repo Github](#).

2. Cache (3 points)

Dans la création d'une cache pour votre service, plus précisément sur l'entrée qui demande beaucoup de processing (voir la route qui crée les "soirées") vous prévoyez recevoir jusqu'à 20 paramètres qui peuvent être utilisés en même temps (ces paramètres ont une cardinalité moyenne de 100). Comment procéderiez vous pour mettre en cache une partie des données pour réduire la charge serveur? Considérations: Le processus suivrait celui pour la question précédente, mais avec une plus grande possibilité de paramètres, et c'est une cache de style clé-valeur (key-value), sans metadata.

3. Les aventuriers du rail (3 points)

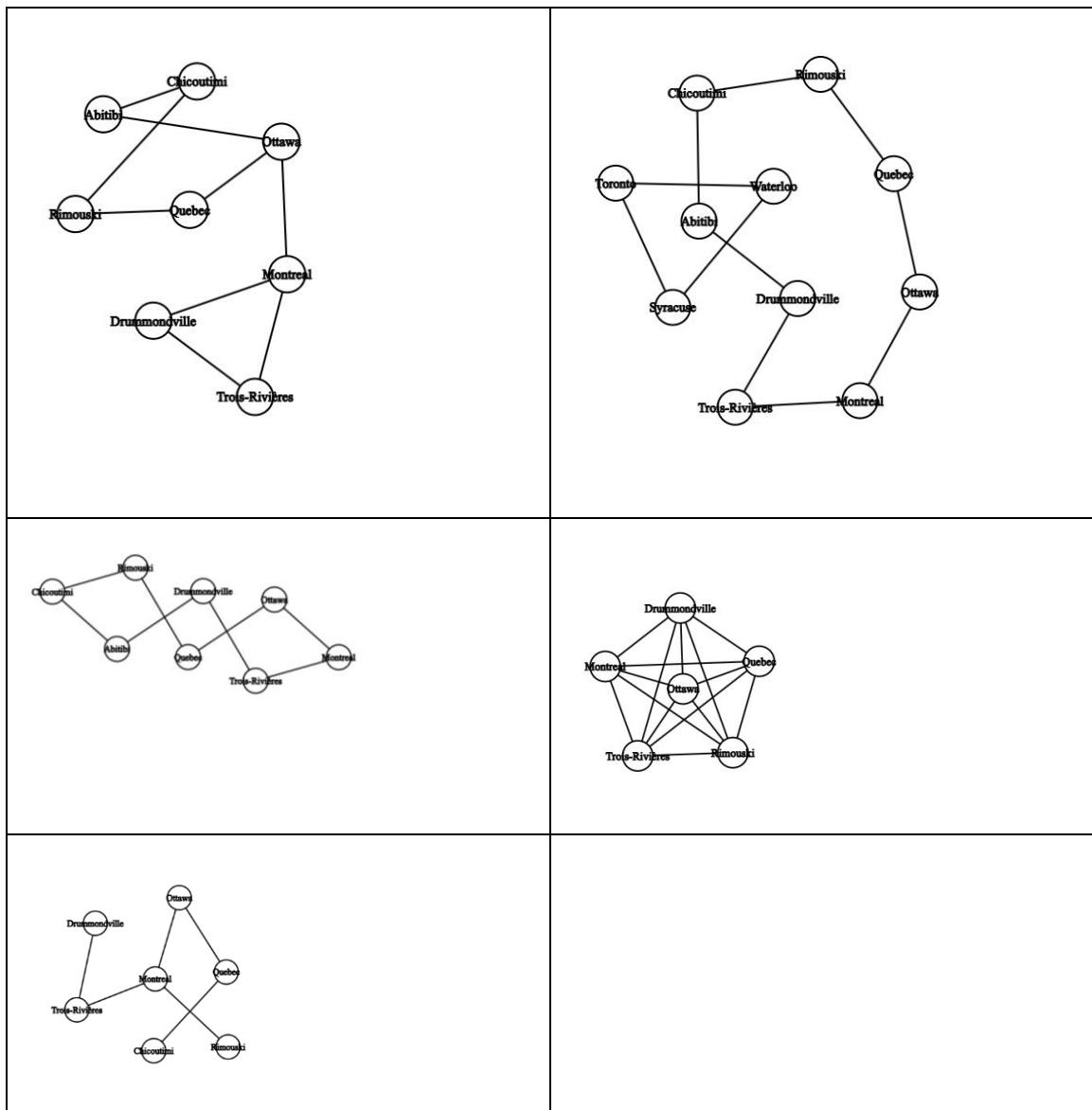
Pour l'application développée pour Evening Dream, il sera nécessaire de concevoir des algorithmes spécialisés pour valider de nouveaux plateaux de jeu du titre *Les Aventuriers du Rail*.

Votre mission consistera à décrire certaines cartes créées par l'entreprise à l'aide de concepts issus de la théorie des graphes, puis à associer les termes pertinents de la banque de mots à chaque carte.

Exemple : Une carte « X » contient à la fois des chemins hamiltoniens et des chemins eulériens, mais n'est pas connexe.

Banque de termes à appliquer:

Hamiltonien	Eulérien	Connexe (/non-connexe)
Planaire	Biconnexe (2-connexe)	Complet
Régulier	Acyclique	



Mandat 7 – Systèmes - DreamKernel (11 points)

DreamKernel est une entreprise de développement informatique bas niveau qui s'occupe de vos besoins en matière d'opérations et/ou d'applications bas niveau.

1. Deadlocks (5pts)

Une de leur application tente de construire des chaînes de caractères aléatoires comme Cloudflare et son fameux mur de «Lava lamps», mais le jet initial, écrit par un employé qui a quitté récemment, manque de tolérance aux deadlocks. En effet, l'application devrait pouvoir rouler éternellement, mais elle se retrouve dans un état d'attente en continu, sans pouvoir y en sortir. Votre objectif étant de rendre l'application résiliente à ce problème. Le code à compléter ainsi que les instructions se trouvent [dans le repo Github](#).

2. Locks (3 points)

Plusieurs systèmes couramment utilisés de nos jours utilisent un mix de Mutex et de sémaphore pour la lecture/écriture de données. Expliquer la différence entre un Mutex, une Sémaphore, et comment une BD utiliserait ces concepts pour ses lectures / écritures (vous pouvez ignorer le concept de transaction que SQL et d'autres technologies offrent).

3. Threads (3 points)

Selon le code suivant, déterminez quelles sont les sorties possibles de ce programme par sa sortie standard. Expliquez le processus qui fait en sorte que ces sorties soient possibles, puis expliquez la différence que l'on aurait si on appliquait un sleep(1) entre chaque paire d'instruction printf (A1-A2, B1-B2, C1-C2).

```
#include <stdio.h>
#include <pthread.h>

void* workerA(void* arg) {
    printf("A1\n");
    printf("A2\n");
    return NULL;
}

void* workerB(void* arg) {
    printf("B1\n");
    printf("B2\n");
    return NULL;
}

void* workerC(void* arg) {
    printf("C1\n");
    printf("C2\n");
    return NULL;
}

int main() {
    int thread = 0;
    pthread_t t1, t2, t3;

    if (thread == 0) {
        pthread_create(&t1, NULL, workerA, NULL);
        pthread_create(&t2, NULL, workerB, NULL);
        pthread_create(&t3, NULL, workerC, NULL);

        pthread_join(t1, NULL);
        pthread_join(t2, NULL);
        pthread_join(t3, NULL);

        printf("Main done\n");
    }
    return 0;
}
```

Mandat 8 – UI/UX – Evident – Question partenaire (14 points)



Evident est une entreprise internationale reconnue pour ses solutions de contrôle non destructif permettant de détecter des défauts internes comme des fissures, porosités, vides ou irrégularités dans divers matériaux. L'entreprise développe également des solutions logicielles spécialement conçues pour des techniciens et inspecteurs sur le terrain, généralement non-informaticiens. Ces outils doivent être extrêmement simples à utiliser, robustes, ergonomiques, et fonctionner de manière fiable dans des environnements exigeants (usines, chantiers, zones bruyantes ou faiblement éclairées).

Dans le cadre de cette mise en situation fictive, **Evident** souhaite explorer de nouvelles approches UX pour de futurs outils numériques. Un prototype expérimental (fictif) a donc été développé : une petite application web permettant d'enregistrer des données d'inspection, visualiser des métriques simples et recevoir des recommandations automatiques.

Or, malgré un fonctionnement technique correct, ce prototype fictif a généré plusieurs retours négatifs de la part d'utilisateurs tests. Evident aimerait donc une analyse critique de l'expérience utilisateur. Votre tâche : effectuer une évaluation critique UX du logiciel, en pointant des obstacles potentiels et en proposant des améliorations. Le prototype (fictif) se trouve dans repo Github du mandat 9, vous trouverez le code vers l'application (HTML/JS/CSS), ainsi que les explications pour rouler celle-ci.

1. Charge cognitive (2 points)

Analysez l'interface du prototype et identifiez deux éléments qui pourraient augmenter la charge cognitive des utilisateurs. Proposez une solution pour réduire cette charge tout en conservant les fonctionnalités.

2. Dark patterns (2 points)

Expliquer ce qu'est un Dark pattern dans le domaine du UI/UX. Identifier un dark pattern dans l'application Justifiez pourquoi c'est un dark pattern et comment vous feriez pour le régler.

3. Lacunes et améliorations UX (4 points)

- Identifiez au moins 3 lacunes ou points de friction dans l'expérience utilisateur de l'outil.
- Pour chaque lacune faites un lien avec les principes d'utilisabilité de Tognazzini (lien dans le repo) et comment ceux-ci ne sont pas suivis.
- Ensuite proposez une amélioration concrète par lacune, en justifiant comment elle réglerait celle-ci.

4. Interface moderne (6 points)

Comme vous l'avez probablement remarqué, interface actuelle ne correspond pas aux standards modernes d'ergonomie et de lisibilité. Votre mission : Modifiez les fichiers *index.html* et *styles.css* tout en conservant tous les éléments fonctionnels. Votre objectif est de rendre l'application plus "clean" et moderne, mais adapté au bon type d'utilisateur.

L'évaluation se fera en comparaison avec les autres équipes. Le code à compléter ainsi que les instructions se trouvent dans [le repo Github](#).

Mandat 9 – Logique numérique – Wired Dreams (13 points)

Wired Dreams, populaire pour conceptions décousues de bas niveau, vous demande de les aider à créer leurs circuits logiques pour l'alimentation de LED avec des entrées et des séquences un peu particulières.

1. Conception d'équations (5 points)

Faites la table de karnaugh et donnez l'équation simplifiée (minterm ou maxterm) de l'ouverture d'une LED, selon les consignes suivantes. Les entrées sont libellées A, B, C, D, E et si deux consignes sont en conflits, la priorité va à la première consigne, suivie de la deuxième etc. les cas ignorés par les consignes sont considérés comme des "DON'T CARE".

Considération: Les équations peuvent utiliser les points d'exclamation, les apostrophes ou une ligne au-dessus de toute la formule, ou les \neg pour la **négation**, les \wedge (et), \vee (ou), etc.

Toutefois faites attention d'utiliser une forme de notation valide sinon des pénalités peuvent s'appliquer. Par exemple, «!A» n'est pas valide.

* Des tables de Karnaugh vides sont disponible pour vous aider

- 1e. La LED est **ALLUMÉE** si (A et B), mais pas (C ou !E).
- 2e. La LED est **ALLUMÉE** si D est allumée.
- 3e. La LED est **FERMÉE** si B ou E est allumée.
- 4e. La LED est **ALLUMÉE** si (C et E), mais pas B.
- 5e. La LED est **ALLUMÉE** si !A et (B ou C).
- 6e. La LED est **FERMÉE** si A et D sont allumés.
- 7e. La LED est **ALLUMÉE** si exactement deux des entrées A, C, E sont à 1.
- 8e. La LED est **ALLUMÉE** si toutes les entrées sont à 0.
- 9e. La LED est **FERMÉE** si C est allumé et E est éteint.

Exemple pour les priorités des règles

Si j'ai les règles

1. Si A ou B => LED Allumée (priorité)
2. Si A et B => LED Fermée

A \ B	0	1
0	X	1
1	1	1

Mais si j'ai les règles

1. Si A et B => LED Fermé (priorité)
2. Si A ou B => LED Allumé

A \ B	0	1
0	X	1
1	1	0

Tables de karnaugh vide

(Barrez les tables brouillons si vous en utilisez)

		E=0				E=1					
		CD	00	01	11	10	CD	00	01	11	10
AB	00						AB				
	01										
AB	11						AB				
	10										
		CD	00	01	11	10	CD	00	01	11	10
AB	00						AB				
	01										
AB	11						AB				
	10										

Entrez votre formule ici:

2. Circuit et Mealy (5 points)

Concevez le circuit électrique et sa (ou ses) machines de mealy, en utilisant que des D-Latch, des ET, OU et des inverseurs, la situation suivante.

Considérez 4 boutons: **B1, B2, B3 et R**
Et 2 LED: **L1 et L2**

Règles pour **R**:

C'est le bouton de **reset**, peu importe l'état, il rapporte à l'état initial.

Règles pour **L1**:

Si j'appuie sur **B1**, suivi de **B3** suivi de **B2 et B3 en même temps**, **L1** est allumé, et reste allumée jusqu'à ce que **R** ou **B1** suivi de **B2** soit appuyé.

Règles pour **L2**:

Si j'appuie sur **B1** deux fois d'affilée, je n'aurai pas besoin de réappuyer sur **2xB1** pour allumer **L2**, toute mauvaise manœuvre après revient au dernier **2xB1** complété. La seule chose qui permet de sauter l'état, c'est **R**.

La séquence d'activation va comme suit: **2xB1, 1xB2, 2xB1, B3**.

Pour désactiver **L2** une fois allumée: **R** ou **B1x2**

3. Preuves (2 points)

On considère la proposition suivante : $P(n)$: $n^2 - n$ est pair pour tout entier $n \geq 0$. On souhaite démontrer cette proposition par induction. Quelle suite de raisonnements permet de valider correctement cette démonstration?

- a) - $P(0)$ est faux
 - L'hypothèse d'induction suppose que $k^2 - k$ est pair
 - $(k+1)^2 - (k+1) = k^2 - k + 2k + 2$
 - On en déduit que $P(n)$ est vraie pour n pair uniquement.
- b) - $P(0)$, $P(1)$ et $P(2)$ sont vraies
 - L'hypothèse d'induction suppose que $k^2 - k$ est pair
 - $(k+1)^2 - (k+1) = k^2 - k + 2k + 2$
 - Comme la somme de deux nombres pairs est paire, $P(k+1)$ est vraie
 - On en déduit donc que $P(n)$ est vraie pour tout $n \geq 0$.
- c) - $P(0)$, $P(1)$ et $P(2)$ sont vraies
 - $P(0)$ est vrai, mais $P(1)$ est faux
 - L'hypothèse d'induction suppose que $k^2 - k$ est impair
 - $(k+1)^2 - (k+1) = k^2 - k + 2k + 2$
 - On en déduit donc que $P(n)$ est vraie pour n impair uniquement.
- d) - La vérification de la base est suffisante
 - Il n'est pas nécessaire d'utiliser l'induction
 - On en déduit donc que $P(n)$ est vraie pour tout n pair.

4. Prédicats (1 points)

Soit la formule: $(p \vee q) \wedge (\neg p \vee r)$

Sélectionnez la/les nature de cette formule :

- a) Tautologie
- b) Contradiction
- c) Satisfaisable

Mandat 10 – AI - SchleepShop (12 points)

SchleepShop est une entreprise spécialisée dans le développement de systèmes d'intelligence artificielle pour améliorer la qualité du sommeil. Leur objectif est d'offrir des recommandations personnalisées basées sur les habitudes de sommeil, les données physiologiques et les préférences des utilisateurs.

La plateforme de SchleepShop utilise des algorithmes d'apprentissage automatique pour :

- Prédire les meilleures conditions de sommeil pour un utilisateur donné.
- Générer des plans de sommeil personnalisés.
- Déetecter des anomalies ou tendances dans le comportement de sommeil.

La compagnie souhaite que votre équipe évalue et propose des solutions théoriques pour améliorer leurs modèles et l'utilisation de l'IA dans la plateforme.

1. Modèles prédictifs (3 points)

Expliquez quelles méthodes d'apprentissage supervisé pourraient être utilisées pour prédire la qualité du sommeil d'un utilisateur en fonction de ses habitudes. Considérez que SchleepShop possède énormément de données par rapport aux habitudes de ces utilisateurs, ainsi que des métriques de sommeil de ceux-ci. Justifiez pourquoi vous pensez que ces méthodes seraient mieux appliquées dans ce contexte en présentant les avantages et limites de chaque méthode proposée.

2. Prétraitement des données (2 points)

Identifiez les types de données pertinentes pour entraîner les modèles de prédiction sur le sommeil (Soyez créatifs). En fonction de ces données, expliquez quelles étapes de prétraitement sont nécessaires pour obtenir des données fiables et exploitables.

3. Évaluation et validation (2 points)

Proposez une méthode d'évaluation pour mesurer la performance des modèles prédictifs. Expliquez comment éviter le surapprentissage (overfitting) et garantir la généralisation du modèle.

4. Éthique et biais (2 points)

Identifiez 3 risques de biais dans les recommandations personnalisées de sommeil. Proposez une stratégie pour minimiser chaque biais et assurer une utilisation éthique de l'IA.

5. Prompt Engineering (3 points)

SchleepShop veut utiliser l'IA générative pour créer ses plans de sommeil personnalisés. Aidez-les à rédiger des prompts robustes pour permettre à leur IA de retourner un message qu'ils pourront utiliser par la suite. Vous n'avez pas le droit de tester vos prompts puisque l'IA est restreinte, utilisez donc uniquement la théorie de prompt engineering pour concevoir vos prompts. Le code à compléter ainsi que les instructions se trouvent dans [le repo Github](#).

Mandat 11 – Créativité Informatique - Nocturna Solution (28 points)

Nocturna Solutions, votre propre compagnie **Nocturna Solutions** doit aussi évoluer et améliorer ses pratiques internes. En tant qu'équipe de 7 consultants, vous êtes parfois appelés à travailler sur des projets plus créatifs, ludiques ou organisationnels qui concernent directement la culture et l'image de l'entreprise.

1. Logology (7 points)

Nocturna Solutions veut moderniser son image et se doter d'un logo original. Cependant, par manque de budget, l'équipe marketing a décidé que ce logo devait être généré en ASCII art directement depuis du code. Le code à compléter ainsi que les instructions se trouvent dans [le repo Github](#).

2. Sloganology (7 points)

En pair avec son nouveau logo, Nocturna veut se doter d'un outil de génération de slogans intelligent. La qualité de la solution sera évaluée en fonction de son originalité, la qualité des résultats en lien avec l'entreprise, l'unicité des résultats et leur justesse grammaticale. Le code à compléter ainsi que les instructions se trouvent dans [le repo Github](#).

3. Officiology (7 points)

Nocturna Solutions veut redéfinir son bureau idéal pour maximiser la joie et la productivité de ses consultants. L'espace de travail est représenté comme une grille en 2D composée de tuiles disponibles. Plusieurs éléments doivent être placés stratégiquement dans cet espace, comme : les bureaux des consultants, des plantes, une machine à café, des fauteuils de sieste et autres. Votre objectif est de proposer un layout optimal pour maximiser la valeur totale de joie de l'équipe. Le code à compléter ainsi que les instructions se trouvent dans [le repo Github](#).

4. Emojiology (7 points)

Nocturna Solutions veut analyser les tendances de sommeil de ses employés. Pour ce faire, elle possède des données sur l'heure de début de sommeil, et l'état de sommeil à chaque heure subséquente des ses employées. L'objectif est donc de déterminer quelle est la tendance moyenne de sommeil de ses employés à chaque heure de la journée. La twist? Les entrées de données sont en emojis, le code doit être rédigé en *Emojicode* et la sortie doit être fournie en emojis également. Le code à compléter ainsi que les instructions se trouvent dans [le repo Github](#).

Fin de l'examen 😊