IFT1169 – TRAVAIL PRATIQUE #3 – 11 novembre 2025

« Sécurité avancée des mots de passe, 3e partie »
Mohamed Lokbani

Mise à jour
11-11-2025 diffusion de l'énoncé.

Équipes : le travail est à faire en **monôme ou en binôme**. Vous ne devez pas être plus de deux par équipe et ne remettre qu'un seul travail.

Remise : une seule remise est à effectuer par voie électronique au plus tard le **vendredi 12 décembre 2025 à 23h59**. Aucune prorogation ne sera accordée.

Conseils : n'attendez pas le dernier jour pour commencer votre travail. Vous n'aurez pas le temps nécessaire pour le réaliser.

But : ce TP vise à intégrer l'utilisation des conteneurs de la STL, les algorithmes de la STL, et la gestion de la persistance des données (fichiers) pour implémenter des fonctionnalités de sécurité avancée des mots de passe. Il s'appuie et étend les fonctionnalités développées dans les TP1 et TP2.

Énoncé : dans le prolongement du travail réalisé pour le générateur de mots de passe du TP1 et du TP2, vous devez maintenant développer une application capable de se souvenir des derniers mots de passe générés pour un utilisateur donné. Ceci va permettre d'éviter que ce dernier puisse utiliser un « vieux » mot de passe.

1. Gestion des mots de passe avec la STL

L'interdiction des conteneurs de la STL est levée. La structure interne de la classe Compte (ou une nouvelle structure de gestion) doit évoluer pour gérer l'historique des mots de passe. Elle doit stocker les 5 derniers (ou plus) mots de passe utilisés par cet utilisateur. Le nombre « N » de mots de passe est défini à travers l'interface. La valeur par défaut est 5.

Lorsque l'utilisateur génère et valide un nouveau mot de passe, l'ancien mot de passe « actuel » est déplacé dans le vecteur historique. Si l'historique des mots de passe dépasse sa taille maximale (par exemple 5), le plus ancien mot de passe est retiré (FIFO - First In, First Out). Le mot de passe actuel est mis à jour.

Lors de la génération d'un nouveau mot de passe, vous devez implémenter une vérification stricte. Utilisez un algorithme de la STL, comme « find » ou « any_of » (ou un autre) pour vérifier si le mot de passe nouvellement généré est présent dans l'historique. Vous devez expliquer dans le rapport le choix de l'algorithme STL utilisé et pourquoi il est plus adapté que l'implémentation d'une boucle simple.

Si le mot de passe est trouvé, il est refusé et un nouveau doit être généré.

L'utilisateur d'un conteneur associatif est recommandé pour ce genre de tâche. La clé sera le login pour permettre une recherche de compte efficace. Mais libre à vous de choisir un autre conteneur, à condition de bien documenter les raisons dans le rapport.

2. Gestion de la sauvegarde

La sauvegarde dans le fichier doit maintenant inclure le login, le mot de passe actuel, ET l'historique.

Vous devrez définir un nouveau format de fichier pour structurer les données, par exemple : login;mdp_actuel;ancien_mdp1,ancien_mdp2,ancien_mdp3,ancien_mdp4,ancien_mdp5

Réutilisez et modifiez les opérateurs surchargés du TP2 pour gérer la lecture et l'écriture de cette nouvelle structure (login + historique). La lecture devra utiliser des techniques d'analyse de chaînes de caractères pour séparer les anciens mots de

passe.

Toutes les opérations de lecture et d'écriture sur les fichiers doivent inclure une vérification explicite de l'ouverture du flux et de l'état des opérations. En cas d'échec (fichier introuvable, un problème de permissions), un message d'erreur clair doit être affiché via l'interface graphique.

3. Extension de l'interface graphique

L'interface doit permettre de générer 5 mots de passe de test pour un utilisateur donné (login). Ces 5 mots de passe sont générés selon **une nouvelle règle décrite dans l'annexe 1**.

Pour chacun de ces 5 mots de passe temporaires, l'utilisateur doit pouvoir sélectionner un mot de passe et effectuer le test de saisie (ressaisie du mot de passe pour prouver la lecture) comme défini dans le TP1.

Pour vérifier l'historique : un bouton ou une section devrait permettre à l'utilisateur de tester si un mot de passe donné (saisi manuellement) pourrait être utilisé par un utilisateur spécifique. L'application répondra : « Mot de passe acceptable » ou « Mot de passe rejeté, car il figure dans l'historique des 5 derniers ».

Ces 5 mots de passe de test sont temporaires et ne sont pas sauvegardés dans l'historique, sauf si l'utilisateur les valide dans la section de génération habituelle.

L'interface doit clairement distinguer la validation/sauvegarde d'un nouveau mot de passe (qui met à jour l'historique) de la simple génération d'un mot de passe temporaire (qui ne modifie pas l'historique, comme pour la section de test).

Fichier projet : si vous réalisez le travail avec **Codelite**, vous devrez remettre l'espace de travail, après l'avoir préalablement nettoyé. Si vous travaillez « à l'ancienne », vous devrez inclure le fichier « **makefile** » qui permet de recompiler votre projet.

Hypothèses et contraintes

- Vous devez utiliser le standard « **c++20** » comme référence.
- Faites attention au format d'affichage des lettres accentuées.
- Nous validerons votre travail sur une machine de la **DESI**. Assurez-vous d'avoir effectué toutes les validations nécessaires sur une telle machine avant de le remettre.

Fichiers à remettre : vous devez remettre le fichier compressé « **tp3A25.zip** » via le système de remise de « **Studium** ». Incluez également un rapport décrivant le travail effectué.

Barème

Ce TP est noté sur 15 **points**, répartis comme suit :

- **Organisation du code :** 3 points
- **Fonctionnalités :** 4 points
- **Interface graphique :** 4 points
- **Rapport :** 2 points
- **Avis global :** 2 points (programmation, clarté du code, etc.)

Pénalités

En plus des points indiqués ci-dessus, vous risquez d'en perdre dans les cas suivants :

- **Non-remise électronique :** 0 (volontaire ou par erreur)
- **Programme qui ne compile pas :** 0
- **Programme qui compile, mais ne respecte pas les spécifications :** 0
- **Avertissements (warnings) non corrigés :** la pénalité dépend de la quantité, à partir de -0.25 point.
- **Aberrations de codage :** même si « tous les chemins mènent à Rome », faites l'effort nécessaire pour éviter de prendre le plus long.

Documents fournis

- Feuille de route « A25Tp3_feuille_de_route.odt »

Communications

- Par courriel : une seule adresse « dift1169@iro.umontreal.ca ».

Pour faciliter le traitement de votre requête, inclure dans le sujet de votre courriel, au moins la chaîne [IFT1169] et une référence au tp03.

- En personne : à la démo ou sur rendez-vous.

Annexe

Contexte : dans le TP#01, le mot de passe généré était une combinaison aléatoire de chiffres, de lettres et de caractères spéciaux. Bien que sécurisé, il était difficile à retenir. Ce type de mot de passe est rarement apprécié par un utilisateur lambda, car il est complexe à mémoriser.

L'utilisateur préfère généralement un mot de passe simple et facile à retenir. Cependant, utiliser son nom ou celui de son animal n'est pas une solution sécurisée, car ces choix sont faciles à deviner.

Solution proposée : une alternative consiste à utiliser une série de mots courants, faciles à retenir. Le fichier « wordlist_fr_4d.txt », fourni avec cet énoncé, contient des mots français générés à partir d'une liste initiale, selon la méthode « Diceware » (jet de 4 dés). Le fichier doit-être chargé une seule fois au démarrage de l'application. Les mots peuvent être préservés par exemple dans un conteneur approprié pour simplifier l'accès.

Exemple : après un jet de 4 dés, nous obtenons « 1141 », ce qui correspond au mot « aide ».

Ce fichier provient de :

https://github.com/mbelivo/diceware-wordlists-fr/blob/master/wordlist_fr_4d.txt

Travail demandé :
• Ajuster votre programme pour générer un mot de passe composé par défaut de 4 mots tirés au hasard dans le fichier.
• Assembler ces mots avec un tiret comme séparateur.
• Permettre à l'utilisateur de modifier le nombre de mots via l'interface.

L'interface graphique doit implémenter une validation de la saisie pour le nombre de mots Diceware (Nmots), s'assurant que la valeur entrée par l'utilisateur reste entre 4 et 8 (bornes incluses).

Exemple :

No du tirage	Résultat (4 dés)	Mot
1er	1631	cette
2e	6231	tarte
3e	2451	est
4e	1451	bonne

Mot de passe final : **cette-tarte-est-bonne**

C'est nettement plus facile à retenir que « RFNEy6p97Pw501my » !

Remarque : Le fichier utilisé ne contient pas de mots accentués pour simplifier les tests.