



Objectifs

- **Réaliser** l'itération 2 de l'application Gestion Centralisée de logs
 - **Améliorer** l'interaction utilisateur
- **Maîtriser** les outils DevOps

Contenu technique

- Programmer une alternative complexe
- Programmer une boucle

Durée 2h30 itération, 2h30 exercices

I. Améliorer l'interaction utilisateur

Votre première itération est fonctionnelle. Nous allons améliorer celle-ci en ajoutant des fonctionnalités au programme.

I.1 Alternative complexe.

Pour traiter l'ensemble des choix de menu, l'alternative `if...else` devient très lourde.

Q1. Utiliser l'instruction **switch case** pour améliorer la lisibilité de votre code.

I.2 Répétition

Le langage C++ offre 3 possibilités de répétitions d'un traitement (bloc de programme).
Votre programme doit pouvoir être répété autant que l'utilisateur le souhaite.

Q2. En consultant le cahier des charges, déterminez l'exigence correspondante à la répétition du programme.

Q3. Quelle est la touche du clavier utilisée pour quitter le programme ?

Q4. Utiliser la boucle de votre choix pour traiter le cas ci-dessus.

II. Outils DevOps

II.1 Formatage du code

Suivre la page [Standard de codage](#) pour formater correctement votre code

Résultat attendu :

- Le code doit être correctement indenté
- Les noms de variables doivent respecter la convention de nommage



II.2 Documentation du code

Suivre la page [Documenter son code](#) pour documenter correctement votre code

Résultats attendus :

- Le fichier main.cpp doit être correctement documenté
- La fonction main() doit être correctement documentée
- Les points de fonctionnement délicats du programme doivent être explicités

II.3 Sauvegarde de votre travail

Suivre la page [Git premiers pas](#) pour sauvegarder votre travail sur le serveur GIT de la section.

Résultats attendus :

- Un dépôt **<VOTRE_NOM>_Gestion_Log** doit être configuré sur le serveur de la section
- Un tag (v2.0 avec un message indiquant la fin de l'itération) doit être créé sur le serveur.
- Votre travail (**liste fichiers ci-dessous**) doit être sauvegardé sur ce dépôt
 - Le dossier test
 - CMakeLists.txt
 - cmake-modules
 - conanfile.txt
 - Doxyfile
 - main.cpp
 - readme.md
 - .gitignore
 - .gitlab-ci.yml

II.4 Pipeline d'intégration

Le projet modèle que vous utilisez est livré avec un pipeline d'intégration DevOps. Il contient 3 étapes :

- Compilation de votre projet
- Tests SAST
- Génération de la documentation

Q5. Vérifier sur gitlab que le pipeline est correctement exécuté.

Q6. Intégrer un screenshot du résultat dans votre compte rendu.

La documentation est téléchargeable à la fin du pipeline.

Q7. Personnaliser le fichier Doxyfile de votre projet (nom, logo, ...) en fonction de votre projet.



III. Livrable

Sur le moodle de la section

→ Votre compte rendu de TP avec les réponses aux questions de celui-ci

Sur le serveur GIT de la section

→ Le projet <VOTRE_NOM>_Gestion_Log avec le tag correspondant à la fin de l'itération2.

IV. Entraînement

Des exercices vous sont proposés pour maîtriser les notions abordées dans cette itération.

Q8. Faire au moins 3 exercices (au choix selon vos besoins)

Le texte des exercices sont sur le site du [bts CIEL](#), code à remettre avec votre compte rendu.

IV.1 Affectation de variables

Exercice 2.a: Permutation de deux variables.

IV.2 Afficher un texte, entrer une variable

Exercice 3.b : Déclarer, afficher

Exercice 3.d : Par ici la monnaie

IV.3 Conditions et alternatives

Exercices 4.a : Positif ?

Exercice 4.c : Voyelle ou consonne ?

IV.4 Les itérations

Exercice 5.a : while do for

Exercice 5.d : Dessin