

IFT1169 – TRAVAIL PRATIQUE #1 – 18 septembre 2024**« Entreprise de jeux vidéos »****Mohamed Lokbani****Mise à jour**

17-09-2024 diffusion de l'énoncé.

28-09-2024 ajustement de l'affichage obtenu en sortie.

06-10-2024 un autre ajustement de l'affichage obtenu en sortie.

Équipes : Le travail est à faire en monôme ou en binôme, pas plus de deux. Vous ne remettez qu'un seul travail par équipe.**Remise :** une seule remise est à effectuer par voie électronique le samedi 12 octobre 2024, 23h59 au plus tard, sans possibilités de prorogation.**Conseils :** n'attendez pas le dernier jour avant la remise pour engager votre travail. Vous n'aurez pas le temps nécessaire pour le réaliser.

But : ce TP a pour but de vous faire pratiquer les différents paradigmes de la programmation orientée objet. Nous allons examiner en particulier les notions d'héritage. Ce travail va vous permettre aussi de vous familiariser avec la syntaxe de base d'un fichier « makefile ».**Énoncé :** une entreprise de développement de jeux vidéo emploie deux catégories d'employés : des développeurs et des managers. Chaque développeur est spécialisé dans un langage de programmation particulier, possède un certain niveau d'expérience en programmation et est associé à un ensemble de projets. Le manager, quant à lui, dispose d'un budget fixe et supervise un ensemble de projets réalisés par une équipe de développeurs qui lui est associée. Pour chaque employé de cette entreprise, les renseignements personnels suivants sont enregistrés : nom, prénom, âge et salaire.**Description :** pour ce travail pratique, nous fournissons deux fichiers « managers.csv » et « programmeurs.csv ». Chaque fichier contient des données relatives à différentes catégories d'employés.

Vous allez d'abord organiser l'énoncé sous forme de classes avec une structure hiérarchique. Ensuite, vous lirez les données des fichiers et les stockerez dans une structure de données dynamique. Enfin, vous réaliserez les opérations suivantes :

Q1/ Combien de managers sont associés au projetH?

R1/ 2: **Douglas Joseph**, **Angela Stone**

Q2/ Combien de programmeurs sont associés au projetH?

R2/ 3: **Carrie Ingram**, **Frank Roberson**, **Kimberly Pearson**

Q3/ Le programmeur **Day** est associée à quel manager?

R3/ **Kenneth Parker**

Q4/ Le programmeur **Day** est associé à quel projet?

R4/ **projetA**

Q5/ Le manager **Barton** est associé à quel programmeur?

R5/ **Michael Weaver**, **Phillip Hicks**, **Raven Jensen**, **Lisa Thompson**

Q6/ Le manager **Barton** est associé à quel projet?

R6/ **projetA**

L'idée est de voir comment réaliser ce genre d'opérations.

Hypothèses et contraintes

- Vous devez utiliser le standard « c++20 » comme référence.
- Votre programme ne doit pas utiliser de tableaux statiques; toutes les déclarations doivent être dynamiques.
- Vous ne pouvez pas utiliser les conteneurs de la « STL », ni ses algorithmes.
- Vous pouvez utiliser le type « string » pour représenter une chaîne de caractères.
- Vous ne devez pas programmer directement les affichages désirés dans votre code. Ce serait le comble! Vous allez plutôt programmer les fonctionnalités permettant d'obtenir ces affichages en sortie.
- Vous devez utiliser les fonctionnalités d'affichage en sortie, décrites dans « format » du standard « c++20 ».

Fichiers de test : nous allons regrouper les questions dans un fichier commun. Nous utiliserons ensuite les redirections (le symbole « < ») pour fournir ces données à votre programme.

L'affichage étant sur la console, nous allons le rediriger vers un fichier (le symbole « > ») que nous comparerons ensuite avec la sortie désirée.

Nous utiliserons des scripts pour compiler, exécuter et corriger votre programme. Assurez-vous de respecter les noms des fichiers. Pour comparer votre sortie avec celle fournie en exemple, redirigez votre sortie dans un fichier avec le symbole « > » comme suit : « tp1.exe > masortie.txt ». Ensuite, comparez-la au caractère près avec la sortie fournie en exemple, en utilisant la commande « diff » (sous « MinGW/MSys » ou la commande « fc » sous « DOS », ajustez juste les options) comme suit :

diff -b -w -i -B sortietest.txt votresortie.txt

Si les fichiers sont identiques, vous n'obtiendrez rien en sortie. Sinon, vous verrez les différences entre les deux fichiers.

Fichiers à remettre : vous devez remettre le fichier compressé « tp1A24.zip » via le système de remise de « Studium ». Incluez également un rapport décrivant le travail effectué.

Remise : vérifiez que le code que vous avez produit (qui devrait normalement fonctionner correctement chez vous) fonctionne aussi bien sur les ordinateurs de la DESI.

Barème

Ce TP1 est noté sur 15 points.

- Conception Orientée Objet : 2 points
- Fonctionnalités : 5 points
- Rapport : 3 points

Nous ne vous demandons pas d'expliquer à quoi sert chaque variable « i », boucle « for » ou boucle « while ». Vos commentaires dans le code devraient expliquer leur utilité. Mettez-vous à la place d'une personne qui connaît l'objectif de votre programme sans connaître votre code. En lisant votre rapport, elle devra pouvoir naviguer avec aisance dans votre programme. Elle saura, par exemple, comment la programmation orientée objet a été introduite et pour quelle raison, ainsi que l'utilité des différentes fonctionnalités, etc.

- Makefile : 1 point

Ne remettez pas celui fourni par votre éditeur. Votre « makefile » doit pouvoir fonctionner en ligne de commande. Assurez-vous également qu'il puisse tester votre programme en comparant son résultat au fichier fourni en référence.

- Affichage en sortie : 2 points
- Avis global : 2 points

Programmation, clarté du code, etc.

Par ailleurs, nous utiliserons d'autres jeux de test pour la correction.

En plus du précédent barème, vous risquez de perdre des points dans les cas suivants ...

- La non-remise électronique (volontaire ou par erreur) est sanctionnée par la note 0.
- Un programme qui ne compile pas : 0.
- Un programme qui compile, mais ne réalise pas les choses prévues dans la spécification : 0.
- Les avertissements (warnings) non corrigés : cela dépend de la quantité! À partir de -0.25 et plus.
- Aberration dans le codage : même si tous les chemins mènent à Rome, faites l'effort nécessaire pour éviter de prendre le plus long!

Documents fournis

- Les résultats obtenus
- Le fichier « tp01A24.cpp »
- Feuille de route « A24Tp1_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 tp01.

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