EXAMPLE TEST - SKILLS ONTARIO CONTESTS
EXEMPLE D'EXAMEN - OLYMPIADES DE COMPÉTENCES ONTARIO

CODING / PROGRAMMATION
TI APPLICATION LOGICIELLE

Welcome to the Competition!

The purpose of this contest is to evaluate your understanding and ability in solving a problem using the software as well as displaying coding skills. Different projects have been developed that will require you to build flowcharts and create programs.

Competition Description

As a competitor, you should be prepared to use your own computer with an internet connection operating with Windows or Mac. You are allowed to use the following programming languages: C#, C++, C, and JAVA.

If you are using C#, C++, or C, you should ensure that your code is openable in Visual Studio.

JAVA projects are required to use and submit a **Maven pom.xml** file.

Note that only standard libraries are allowed: Java SE 8, .NET 4.7+, and Microsoft C/C++ runtime.

You must submit an executable .exe for all C language projects or an executable .jar for projects completed with JAVA.

Coding standards, such as proper use of comments and spacing, will be marked. **All files should be submitted in a single .zip file. The final submission file should be named FirstName_LastName.zip.**

Good Luck!

Part A – Knowledge and Problem Solving

Please read this section carefully:

- You will submit your answers as a Microsoft Word, Apple Pages, PDF, or image file.
- The section is valued at 25%.
- There should be no communication between candidates.
- There are no resources allowed except online help.

Part A – Knowledge and Problem Solving

1. Create a flowchart walking through the process of purchasing online, installing, and using a new piece of software or video game. Solution should have at least 5 processes.
2. Explain the difference between a frontend and a backend.
3. Explain what happens when you instantiate a new instance of a class in as much detail as you can. For example, 'new AnimalAdoptionAgency(logger)'.

The flowchart and answers can be submitted as a Microsoft Word, Apple Pages, PDF, or image file. Hand-written answers may also be submitted as long as they are legible. Neatness and attention to detail count!

Requirements for Completion:

- Save the file for Part A as **YourName_PartA**.
- Include required files in your submission .zip archive named FirstName_LastName.zip.

Part A – Judging Criteria:

Criteria	Grade Range		
Knowledge Questions	All questions were correctly answered in a detailed manner. **(8-10)**	Most of the questions were answered correctly; solutions could be more detailed. **(4-7)**	Questions were not answered correctly. No attention was given to a detailed answer. **(0-3)**
Flowchart	The flowchart was detailed and contained all the necessary steps for solving the problem. The flowchart was professional, clean, and readable. Proper shapes and symbols were used correctly. **(0-15)**		

Part B – Code Review and Deployment

Please read this section carefully:

- You will use one of the following programming languages: C#, C++, C, or JAVA.
- If you are using C#, C++, or C, you should ensure that your code is openable in Visual Studio.
- JAVA projects are required to use and submit a **Maven pom.xml** file.
- Only standard libraries are allowed: Java SE 8, .NET 4.7+, and Microsoft C/C++ runtime.
- Submit an executable .exe for C language projects or an executable .jar for projects completed with JAVA.
- The section is valued at 75%.
- There should be no communication between candidates.
- There are no resources allowed except online help.
- Save the files you create with the names provided.

Part B – Code Review and Deployment

You will be creating an application for helping an Animal rescue not-for-profit organization manage incoming animals. This Application can be written in either C#, C++, C, or Java and submitted as a .exe or .jar file. Your submission should also include the source code, documentation, and deployment files or installations script.

Requirements for Completion:

The first step in managing the animal rescue is creating a flat file with all the animals in the shelter now. This task will need all the basic CRUD functions (create, read, update, delete).

The file must include the following:

- ID: A generated incrementing 0 padded 8-digit number.
- Species: Dog, Cat, Bird, Rabbit, Small & Furry, Fish, Barnyard, Other
- Name: The animal's name.
- Gender: F or M
- Spayed: Yes or No
- Breed: Collie, Beagle, Siamese, Calico, unknown, etc.
- Colour: Brown, Tabby, White, etc.
- Birthday: Date of the estimated animal's birth. Format dd/mm/yyyy.
- Vaccine Status: Up to date, late, unknown.

- Identification: Bar code, Micro-chipped. If yes, what is the number
- Adoption fee: < $300.

The Application itself can function in one of two ways, either by displaying a user interface or by accepting command line arguments.

Your application must have the following functionalities:

- Add Animal.
    - The breed should be valid
- Remove animal by ID.
- Search for an animal by name or species.
- Display animals sorted by species.
- Display the three oldest animals for each species.
- Usage instructions or a help option

Deployment (No Presentation)

- Create a README.md file that contains how to use your application.
- Create multiple printscreens of each process when running the application such as Add, Delete, and Edit.
- Document your printscreens above.

Post-Secondary Only

Above are the mandatory requirements for the Application for all contestants. The following is a list of required additions for post-secondary contestants only:

- In addition to the basic CRUD functions (create, read, update, delete), your application should be able to **archive** and **restore** the information of animals that have been adopted. Add any fields necessary.
    - An archived animal will not show up in the regular search
    - Include an option to search for archived animals within a timeframe
    - Include an option to archive all animals adopted at least three months ago
- Your application should be able to calculate the adoption fee for each animal based on the criteria below:
    - Kittens, puppies, and other young animals (below one year of age): $300
    - Senior animals (above ten years of age): $100
    - All other animals: $200

Bonus (Optional) Marks

Bonus marks will be given for:

- Creativity and innovation.
- Including a "Docker file" and instructions on how to run it.
- Create a new addition to the existing program, except related to adopting an animal. For example, include the person interested in adopting's name, contact information, the animal they want to adopt, why they are a great fit for the animal, etc.

Submission requirements

- Save the files for Part B in a folder named **YourName_PartB.**
- Make sure all your printscreens are included.

Make sure your Application runs and does not crash!

A penalty will be given to any applications that crashes!

Judging Criteria:

Criteria	Grade Range		
Elements Included	All required elements were included. **(15-25)**	Most of the required elements were included. **(5-15)**	Missing more than 2 of the required elements. **(0-5)**
Program Content and Functionality	The program was easy to understand and use. The content was clearly displayed. Data and functions worked as intended. **(30-40)**	The program could be clearer to understand and use. Most functions worked properly. **(20-30)**	The program was not functional and/or crashed. Most functions were missing. **(0-20)**
Code Quality	The code was well organized and easy to read. Good use of comments. **(7-10)**	The code could be better organized and easier to read. Missing some comments. **(4-7)**	The code was disorganized and hard to read. No comments were used. **(0-4)**

Bienvenue à tous les concurrents!

Le but de ce concours est d'évaluer votre compréhension et votre capacité à résoudre un problème à l'aide d'un logiciel ainsi qu'à faire valoir vos compétences en codage. Divers projets ont été développés qui nécessiteront la création d'organigrammes et programmes.

Description du concours

Vous (les concurrents) devez être prêt à utiliser votre propre ordinateur Windows ou Mac doté d'une connexion Internet. Les langages de programmation suivants peuvent être utilisés : C#, C++, C et JAVA.

Si vous utilisez C#, C++ ou C, vous devez vous assurer que votre code peut s'ouvrir dans Visual Studio.

Pour les projets JAVA, il est obligatoire d'utiliser et de soumettre un fichier **Maven pom.xml**.

Veuillez prendre note que seules les bibliothèques standard sont permises : Java SE 8, .NET 4.7+ et Microsoft C/C++ runtime.

Vous devez soumettre un fichier exécutable .exe pour tous les projets en langage C ou un fichier exécutable .jar pour les projets JAVA.

Les normes de codage, notamment l'utilisation appropriée des commentaires et de l'espacement, seront évaluées. **Tous les fichiers doivent être soumis dans un seul fichier compressé (.zip). Le dossier de soumission officiel doit être nommé Prénom_Nomdefamille.zip.**

Bonne chance!

Volet A – Connaissances et résolution de problèmes

Veuillez prendre le temps de lire attentivement :

- Vous devrez soumettre vos réponses sous forme de fichier Microsoft Word, Apple Pages, .pdf, ou image.
- Cette section compte pour 25 % de la note finale.
- Il ne doit y avoir aucune communication entre les concurrents.
- Aucune ressource n'est permise sauf pour l'aide en ligne.

Volet A – Connaissances et résolution de problèmes

1. Créez un organigramme qui décrit le processus pour l'achat en ligne, l'installation, et l'utilisation d'une nouvelle pièce d'un logiciel ou d'un jeu vidéo. La solution doit englober au moins 5 étapes au processus.
2. Expliquez la différence entre les applications *frontend et backend*.
3. Expliquez, en incluant autant de détails que possible, ce qui se passe lorsque vous instanciez une nouvelle instance d'une classe. Par exemple, 'nouvelle AgenceAdoptionAnimaux (enregistreur de données)'

L'organigramme et les réponses peuvent être soumis sous forme de fichier Microsoft Word, Apple Pages, .pdf ou image. Les réponses manuscrites peuvent également être soumises pour autant qu'elles soient lisibles. La clarté et le souci du détail sont importants.

Exigences pour la soumission officielle

- Enregistrez le fichier du volet A en lui donnant le nom de Votrenom_VoletA.
- Veuillez inclure les fichiers requis dans un fichier compressé (.zip) portant le nom de Prénom_Nomdefamille.zip.

Volet A – Critères d'évaluation :

Critères	Pointage		
Questions portant sur les connaissances	Concurrent a répondu correctement et de manière détaillée à toutes les questions. **(8-10)**	Concurrent a répondu correctement à la plupart des questions; les solutions auraient pu être plus détaillées. **(4-7)**	Concurrent a répondu incorrectement aux questions. N'a pas tenté de fournir une réponse détaillée. **(0-3)**
Organigramme	L'organigramme était détaillé et contenait toutes les étapes nécessaires pour corriger les problèmes. L'organigramme était de qualité professionnelle, clair et lisible. Les formes et symboles pertinents ont été utilisés. **(0-15)**		

Volet B – Examen et déploiement du code

Veuillez lire attentivement cette section

- Vous utiliserez l'un des langages de programmation suivants : C#, C++, C, or JAVA.
- Si vous utilisez C#, C++ ou C, vous devez vous assurer que votre code peut s'ouvrir dans Visual Studio.
- Pour les projets JAVA, il est obligatoire d'utiliser et de soumettre un fichier **Maven pom.xml.**
- Veuillez prendre note que seules les bibliothèques standard sont permises : Java SE 8, .NET 4.7+ et Microsoft C/C++ runtime.
- Vous devez soumettre un fichier exécutable .exe pour tous les projets en langage C ou un fichier exécutable .jar pour les projets JAVA.
- Cette section compte pour 25 % de la note finale.
- Il ne doit y avoir aucune communication entre les concurrents.
- Aucune ressource n'est permise sauf pour l'aide en ligne.
- Enregistrez les fichiers créez en tenant compte des noms fournis.

Volet B – Examen et déploiement du code

Vous devrez créer une application pour aider un refuge pour animaux sans but lucratif à gérer les animaux qui entrent au refuge. Cette application peut être écrite en langage C#, C++, C ou Java et soumise sous forme de fichier .exe ou .jar. Votre soumission doit également inclure le code source, la documentation et les fichiers de déploiement ou le script d'installation.

Exigences pour la soumission officielle

La première étape dans la gestion du refuge pour animaux est la création d'un fichier plat qui énumère tous les animaux présentement dans le refuge. Cette tâche nécessitera toutes les fonctions de base (créer, lire, mettre à jour, supprimer).

Le fichier doit comprendre les éléments suivants :

- ID : un code à 8 chiffres, généré automatiquement, rempli par des zéros
- Espèces : chien, chat, oiseau, lapin, petit et poilu, poisson, basse-cour, autre
- Name : nom de l'animal
- Genre : femelle ou mâle
- Stérilisé : oui ou non
- Race : Colley, Beagle, Siamois, Calico, inconnue, etc.

- Couleur : brun, tigré, blanc, etc.
- Date de naissance : date de naissance approximative de l'animal. Format jj/mm/aaaa.
- Statut vaccinal : à jour, en retard, inconnu.
- Identification : code à barres, puce. Si oui, le numéro
- Frais d'adoption : < 300 $

L'application en soi peut fonctionner de l'une des deux manières suivantes, soit en affichant une interface utilisateur, soit en acceptant des arguments de ligne de commande.

Votre application doit comporter les fonctionnalités suivantes :

- Ajouter animal
    - Race doit être valide
- Retirer l'animal par ID
- Chercher l'animal par son nom ou sa race
- Afficher les animaux par race
- Afficher les trois animaux les plus âgés par race
- Instructions concernant l'usage ou l'option Aide

Déploiement (sans présentation)

- Créez un fichier README.md qui explique comment utiliser votre application
- Créez plusieurs impressions d'écran de chaque processus lors de l'exécution de l'application, notamment Ajouter, Supprimer et Modifier
- Documentez les impressions d'écran ci-dessus

Postsecondaire seulement

Vous trouverez ci-dessus les exigences obligatoires concernant l'application qui s'appliquent à tous les participants. Voici une liste des ajouts requis pour les candidats de niveau postsecondaire seulement :

- En plus des fonctions de base (créer, lire, mettre à jour, supprimer), votre application devrait être en mesure d'**archiver** et de **restaurer** les informations des animaux adoptés. Ajoutez tous les champs nécessaires.
    - Informations archivées d'un animal n'apparaîtront pas dans la recherche régulière
    - Inclure une option de recherche d'animaux dont les informations ont été archivées au cours d'une période
    - Inclure une option d'archivage des informations de tous les animaux adoptés il y a au moins trois mois
- Votre demande devrait être en mesure de calculer les frais d'adoption pour chaque animal en fonction des critères ci-dessous :
    - Chatons, chiots et autres jeunes animaux (moins d'un an) : 300 $
    - Animaux âgés (de plus de dix ans) : 100 $

- Tous les autres animaux : 200 $

Points bonis (optionnels)

Des points bonis seront accordés pour :

- Créativité et innovation.
- Y compris un « fichier Docker » et des instructions sur la façon de l'exécuter.
- Création d'un ajout au programme existant, lié à l'adoption d'un animal. Par exemple, capacité d'inclure le nom de la personne intéressée à adopter, ses coordonnées, l'animal qu'elle souhaite adopter, les raisons pour lesquelles il s'agit d'un jumelage parfait, etc.

Exigences pour la soumission

- Enregistrez les fichiers pour le volet B dans un dossier nommé **Votrenom_VoletB**.
- Assurez-vous que toutes vos impressions d'écran sont incluses.

Assurez-vous que votre application s'exécute et ne plante pas!

Une pénalité s'imposera pour toutes les applications qui se plantent!

Critères d'évaluation :

Critères	Pointage		
Éléments inclus	Tous les éléments requis ont été inclus. **(15-25)**	La plupart des éléments requis ont été inclus. **(5-15)**	Il manque plus de 2 des éléments requis. **(0-5)**
Contenu et fonctionnalité du programme	Le programme était facile à comprendre et à utiliser. Le contenu était clairement affiché. Les données et fonctions ont fonctionné tel que prévu. **(30-40)**	Le programme pourrait être plus convivial. La plupart des fonctions fonctionnaient correctement. **(20-30)**	Le programme n'était pas fonctionnel et/ou a planté. La plupart des fonctions étaient manquantes. **(0-20)**
Qualité du code	Le code était bien organisé et facile à lire. Bon usage des commentaires. **(7-10)**	Le code pourrait être mieux organisé et plus facile à lire. Il manque quelques commentaires. **(4-7)**	Le code était désorganisé et difficile à lire. Aucun commentaire n'a été utilisé. **(0-4)**