v1.1 ChenYl-TECH

FILIERE préING1 • 2024-2025
AUTEURS E.ANSERMIN – R. GRIGNON
E-MAILS eva.ansermin@cyu.fr – romuald.grignon@cyu.fr

DESCRIPTION DU PROJET

Le but de ce projet est de **gérer les adorables pensionnaire du chenil ChenYl-Tech qui s'occupe de différents types d'animaux et les propose à l'adoption**.
Il vous faudra développer une application en **langage C** permettant de gérer le chenil de manière efficace en stockant les informations dans des **fichiers**.
Ce programme offrira une interface textuelle permettant d'ajouter, rechercher, modifier et supprimer des fiches animales ainsi que de gérer d'autres aspects du chenil.

INFORMATIONS GENERALES

- **Taille de l'équipe**
  Ce projet est un travail d'équipe. Il est autorisé de se réunir en groupe de 3 au maximum. Si le nombre total d'étudiants n'est pas un multiple de 3 et/ou si des étudiants n'arrivent pas à constituer des groupes, c'est au chargé de TD de statuer sur la formation des groupes. Pensez-donc à anticiper la constitution de vos groupes pour éviter des décisions malheureuses.
- **Démarrage du projet**
  Vous obtiendrez de plus amples informations quant aux dates précises de rendu, de soutenance, les critères d'évaluation, le contenu du livrable, ..., quand le projet démarrera officiellement.
- **Dépôt de code**
  Vous devrez déposer la totalité des fichiers de votre projet sur un dépôt central Git. Il en existe plusieurs disponibles gratuitement sur des sites web comme ***github.com*** ou ***gitlab.com***.

- **Rapport du projet**
  Un rapport écrit est requis, contenant une brève description de l'équipe et du sujet. Il décrira les différents problèmes rencontrés, les solutions apportées et les résultats. L'idée principale est de montrer comment l'équipe s'est organisée, et quel était le flux de travail appliqué pour atteindre les objectifs du cahier des charges. Le rapport du projet peut être rédigé en français.
- **Démonstration**
  Le jour de la présentation de votre projet, votre code sera exécuté sur la machine de votre chargé(e) de TD. La version utilisée sera la **dernière** fournie sur le dépôt Git **avant** la date de rendu. Même si vous avez une nouvelle version qui corrige des erreurs ou implémente de nouvelles fonctionnalités le jour de la démonstration, c'est bien la version du rendu qui sera utilisée.
- **Organisation**
  Votre projet complet devrait **(dans l'idéal)** être stocké sur un **dépôt git** (ou un outil similaire) tout au long du projet pour au moins trois raisons : éviter de perdre du travail tout au long du développement de votre application, être capable de travailler efficacement en équipe, et partager vos progrès de développement facilement avec votre chargé de projet. De plus il est **recommandé** de mettre en place un **environnement** de travail en **équipe** en utilisant divers outils pour cela (Slack, Trello, Discord, ...)

CRITERES GENERAUX

- Le **but principal** du projet est de fournir une **application fonctionnelle** pour l'utilisateur. Le programme doit correspondre à la description en début de document et implémenter toutes les fonctionnalités listées.
- Votre code sera généreusement **commenté**.
- Tous les éléments de **votre code** (variables, fonctions, commentaires) seront écrits dans la **même langue**. Langue anglaise conseillée mais pas obligatoire.
- Votre application ne doit jamais s'interrompre de manière intempestive (crash), ou tourner en boucle indéfiniment, quelle que soit la raison. Toutes les erreurs doivent être gérées correctement. Il est préférable de d'avoir une application stable avec moins de fonctionnalités plutôt qu'une application contenant toutes les exigences du cahier des charges mais qui plante trop souvent. Une application qui se stoppe de manière imprévue à cause d'une erreur de segmentation ou d'une exception, par exemple, sera un événement très pénalisant.
- Votre application devra être **modulée** afin de ne pas avoir l'ensemble du code dans un seul et même fichier par exemple. Apportez du soin à la conception de votre projet avant de vous lancer dans le code.
- Le livrable fourni à votre chargé(e) de TD sera simplement l'**URL** de votre **dépôt Git** accessible **publiquement**. Même si vous n'avez pas utilisé ce dépôt régulièrement au cours du projet, le code final sera livré dessus.

FONCTIONNALITES DU PROJET

STOCKAGE DES DONNEE SUR LES RESIDENTS:

- Les informations des différents animaux doivent être enregistrées dans un fichier ou plusieurs fichiers.
- Ces fichiers pourront être textuels (ASCII) ou binaires.
- Ces fichiers devront être placés dans un dossier «*animaux*» qui pourra, si vous le souhaitez, également contenir des sous-dossiers.
- Les informations à stocker pour chaque animal sera :
  - Un numéro d'identification unique à l'animal
  - Son nom
  - Son espèce (le chenil doit gerer au minimum 4 espèces différentes) : les chiens, les chats, les hamsters et les autruches. Vous être libre d'etendre cette liste à d'autres espèces d'animaux.
  - Son année de naissance
  - Son poids
  - Un commentaire / une phrase pour décrire l'animal (exemple : « p*etit chien très joueur* » ou bien « *sera toujours content de participer à votre projet d'informatique en marchant sur le clavier*»). **Ce commentaire n'est pas obligatoire** (tous les animaux n'en auront pas) mais la fonctionnalité, elle, est obligatoire.
- Vous êtes libres d'organiser ce/ces fichiers comme vous le souhaitez tant que ces informations puissent être récupérées par votre programme.

FONCTIONNALITES

- Votre application doit dans un premier afficher un menu permettant de choisir l'action à réaliser. Ces actions sont les suivantes :

  - **Rechercher un/des animaux** : il devra être possible de rechercher un ou plusieurs animaux selon les critères suivants :
    1. Le nom
    2. L' espèce
    3. Le type d'age : rechercher un animal jeune (<2 ans), ou sénior (>10 ans).

    Il doit être possible de rechercher selon plusieurs de ces critères de recherche. Si un animal correspond aux critères recherchés, ses différentes informations devront être affichées.

  - **Ajouter un animal qui vous a été confié :** il faudra alors saisir toutes ses information à l'exception de son numéro d'identification qui devra être généré automatiquement. **Attention : votre refuge ne peut pas contenir plus de 50 animaux !**

- **Adoption d'un animal** : bonne nouvelle, un de vos résidents va être accueilli dans une famille. Il vous faudra saisir son numero d'identification pour le faire disparaître de votre base de données.

VARIANTES

- **Inventaire** du refuge :
  - variante **INV_NB_DESC** : Le programme doit afficher le nombre total d'animaux dans le refuge, ainsi que le détail du nombre d'animaux de chaque espèce (par ordre décroissant).
  - variante **INV_AGE_ASC** : Le programme doit afficher le nombre total d'animaux dans le refuge, ainsi que le détail du nombre d'animaux par tranche d'âge (affichage par quartile).
- Gestion du **quotidien**
  - variante **DAY_FOOD** : le programme doit afficher la quantité de croquettes nécessaires au nourrissage quotidien de tous les animaux en respectant les règles suivantes :
    - un hamster reçoit 20g de croquettes par jour
    - une autruche recoit 2.5 kg de croquettes par jour
    - un chat ou un chien recoivent une quantité de croquettes dependant de leur âge. Si le chat/chien a moins de deux ans, il recevra 500g de croquettes, sinon il recevra 10% de son poids.
  - variante **DAY_CLEAN** : le programme doit prévoir la charge de travail hebdomadaire pour nettoyer les abris de chaque animal. Les règles à respecter sont les suivantes :
    - Un abri / une cage vide, ne prendra que 2 minutes à dépoussiérer complètement chaque jour.
    - Une cage contenant un hamster, ou un abri pour un chat prendront chacun 10 minutes par jour + 20 minutes par semaine (pour changer la litière complètement)
    - Un abri avec une autruche prendra 20 minutes par jour et 45 minutes par semaine.
    - Enfin pour un chien, comme il fait ses besoins au dehors, le nettoyage quotidien moyen sera de 5 min, et une session de 20min par semaine supplémentaire.

RESSOURCES UTILES

- **Github**
  https://www.github.com
  https://docs.github.com/en/get-started/quickstart/hello-world
- **Couleurs dans le terminal**
  http://sdz.tdct.org/sdz/des-couleurs-dans-la-console-linux.html
- **Emojis dans le terminal**
  https://unicode.org/emoji/charts/full-emoji-list.html