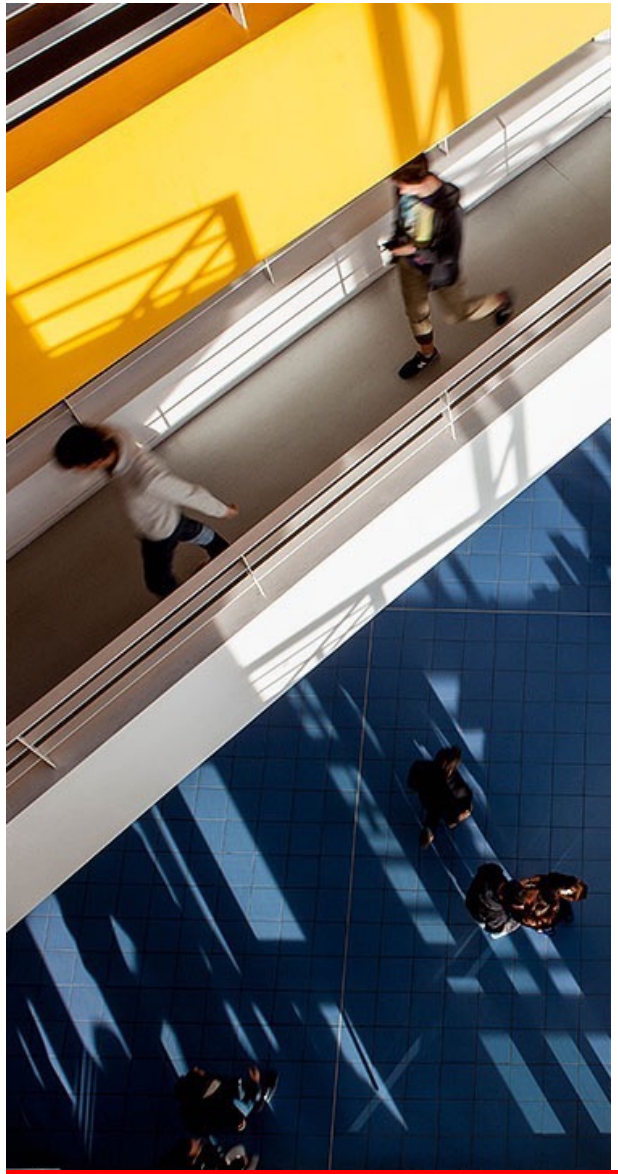


Projet C & Web

Présentation générale

CIR1 Brest/Paris - 2024

francois.legras@isen-ouest.yncrea.fr

sylvain.lefebvre@isen-ouest.yncrea.fr

Sujet

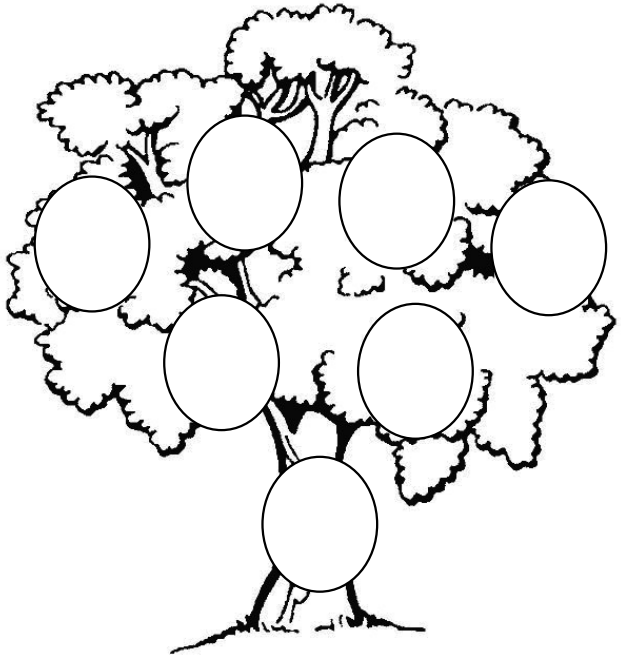
Objectif

Génération d'arbres généalogiques à partir de données existantes

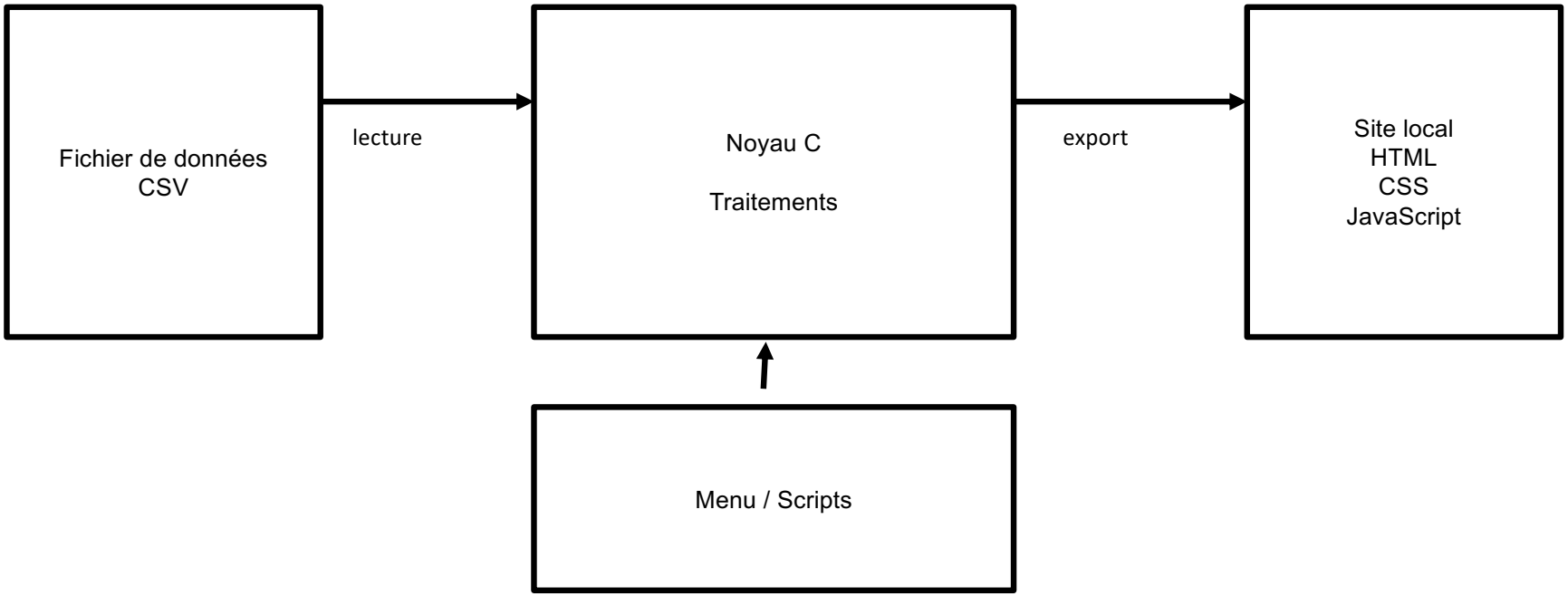
Lecture de données au format CSV

Organisation des données en C

Affichage des données en HTML/CSS



Architecture globale du système



Architecture du système

- Fichier de données CSV (Comma Separated Values) :
  - 3 fichiers de données sont fournis
  - 1 ligne = 1 personne
  - Un fichier CSV peut s'ouvrir et être édité avec un simple éditeur de texte
  - Sur la ligne : id, id_pere, id_mere, nom_de_famille, prenom, date_de_naissance (au format anglo-saxon), ville_de_naissance

```
1,2,3,Hatton,Andrew,9/2/1801,Yorkshire
2,0,0,Hatton,Steve,25/7/1775,Yorkshire
3,0,0,Vockins,Mary,11/10/1777,West Yorkshire
```

- Noyau en C :
  - Lire les données du fichier CSV
  - **Organiser les données dans des structures**
  - **Traiter les données avec des algorithmes**
  - **Exporter les données dans des fichiers HTML**
  - Ecrire des menus et/ou des scripts linux

Architecture du système
Noyau C – structures de données et traitements

- Structure de données
  - **Personne** : id, id_pere, id_mere…
    - Utiliser des pointeurs pour lier certains éléments (père, mère)
  - **Population** : table de hachage utilisant les id ou tableau statique (moins bien) ou autre solution intéressante

- Traitements des données :
  - Deux algorithmes de recherche de données **obligatoires** : ancêtres sur deux générations d'une personne ET fratrie d'une personne (à partir d'un ID)
  - Optionnels : plus lointain ancêtre, recherche d'un ancêtre commun…

> **Attention**
>
> Ne faire que ce qui est obligatoire dans un premier temps. Portez une attention particulière à la complexité algorithmique du code écrit et écrivez un code commenté.

Architecture du système
Noyau C – export HTML & interactions avec le noyau

- L'export HTML est géré par le noyau C
  - Le code HTML est simplement un fichier texte généré directement en C (traitement de chaines de caractères)
  - L'export doit obligatoirement permettre d'obtenir les fichiers générés par les 2 traitements obligatoires à réaliser (ancêtres sur deux générations d'un personne ET fratrie d'une personne)
  - L'export doit être **automatique**. Aucun fichier exporté ne doit être modifié à la main ensuite.

- Création de menus/scripts linux
  - Pourquoi ? Aide à la création du site & réponses à des questions spécifiques
  - Un système pour interagir avec votre noyau est obligatoire
    - Soit un menu en C
    - Soit des scripts linux (bash) qui génèrent les exportations
    - Pourquoi pas utiliser ncurses (attention, difficile)

```
What do you want to do?
-----------------------
1. Show family tree info
2. Export HTML family tree files
3. Export HTML info files
4. Export all files
5. Query family tree
6. Go back to principal menu
-----------------------
Your choice:
```

Architecture du système
Site local

- Affichage des données contenant les informations généalogiques sur des personnes
  - Utiliser HTML, CSS et JS
  - Le noyau C génère les fichiers HTML : cette génération doit être **automatique**
  - Le style des pages HTML est libre (mais obligatoire) : CSS à écrire
  - Site statique (généré 1 seule fois) et local

- Pour résumer, votre projet doit fonctionner grâce aux étapes suivantes :
  - Lecture d'un fichier CSV avec le noyau C
  - Possibilité d'effectuer des requêtes/traitements avec le menu ou via des scripts linux
  - Export de fichiers en HTML
  - Visualisation en local du résultat via un navigateur

Phases du projet

Phase 1 : étude préliminaire

- Présentation du projet
- Lecture et compréhension du sujet
- Réflexions sur la planification et la répartition des tâches au sein du groupe (détail des fonctionnalités principales de votre application)
- Rédaction du livrable

Rendu sur moodle

Phase 2 : implémentation, tests et rendu final

- Implémentation des fonctionnalités principales
- Tests et validation
- Implémentation des fonctionnalités additionnelles
- Tests et validation
- Préparation du livrable et de la recette (présentation)

Rendu sur moodle

Phase 1 : étude préliminaire

- Cahier des charges / planification
  - Lecture et compréhension du cahier des charges
  - Identification des tâches
  - Gestion des dépendances
  - Répartition des tâches dans le temps et par personne
  - Identification des points difficiles et/ou bloquant dans l'implémentation

	J1	J1	J2	J2	J3	J3	…	…	J7
Eleve1		A ?	B ?						
Eleve2		A ?	B ?						

- Identification des tâches
  - Tâche A : structure personne et population
  - Tâche B : remplir la population avec un fichier
- Gestion des dépendances
  - B dépend de A
- Répartition des tâches dans le temps
  - Traiter A avant B, temps nécessaire…
- Répartition des tâches par personnes
  - Elève 1 fait A, Elève 2 fait B (attente …)
  - Elève 1 fait A puis B (seul)
  - Elève 1 et 2 travaillent en binôme sur A puis B (en échangeant)

Phase 1 : étude préliminaire

Une fois la 1ère planification terminée

- Cahier des charges / planification
  - Prévoir les ajouts au projet
  - Décrire et décomposer en sous-tâches : spécifiques, mesurables, atteignables, réalistes, temporelles
  - Ajouter ces tâches au planning
  - Ne pas oublier les tâches suivantes :
    - Intégration (cf slide suivante)
    - Tests et validation
    - Préparation du rendu
    - Préparation de la recette
    - Relecture du code
    - Création et dépôt de l'archive (éviter de s'y prendre à la dernière minute…)

Phase 2 : implémentation

- Implémenter le code
- Travail à 2 : relire et commenter le code de l'autre
- Communiquer et échanger le plus possible avec son binôme : attention aux incompréhensions qui peuvent engendrer des catastrophes (phase "d'intégration")
- Sauvegarder régulièrement son code pour éviter la réflexion "ce matin, ca marchait, mais là, plus rien ne fonctionne" ! C'est l'occasion de se mettre à git

Organisation du projet (30h)
Jeudi 13/06 au mercredi 19/06 2021

Jeudi	Vendredi	WE	WE
Présentation	Étude Rendu 12h		
Étude	Implémentation		

- horaire : 9h-12h et 13h30 - 16h30
- lieu : en Salle

Pas de nourriture dans les salles pendant les heures projet.

C'est un devoir -> absence => 0 / 20 (possible)

Dans les faits : 1h = -1pt

Lundi	Mardi	Mercredi
Implémentation	Implémentation / Relecture	Finalisation du code Envoi sur Moodle à 12h
Implémentation	Implémentation / Relecture	Présentation

Une absence ?

-> Je préviens mon responsable avant !

Livrables

Dates

- Phase 1 : Etude préliminaire

  → vendredi 14/06, 12h00 :

  - Au format pdf, un document regroupant votre planning prévisionnel détaillé, vos différents choix d'implémentation (structures de données, utilisation des menus/scripts…), ainsi que la démonstration (photo ou scan) demandée dans le document du projet.

- Phase 2 : Rendu final

  → mercredi 19/06, 12h00 sur Moodle :

  - Rendu du code source : code organisé et commenté + 1 fichier `readme.md` (Markdown) contenant les procédures de compilation et d'exécution de votre programme
  - Rendu du support de présentation (pdf)

1 seul rendu par groupe

Attention

Archives sur moodle : format zip, nommage : `projetCIR1_Ville_groupeX.zip`

X représente votre numéro de groupe. Tout retard sera sanctionné (l'heure du réseau fait foi). Les fichiers au mauvais format ou avec mauvais nommage seront pénalisés.

Présentation

- Modalités
  - 6 minutes de présentation (2x3min ou 3x2min) + 4 minutes de question
  - Doit utiliser des supports (pdf ou ppt)
  - Les membres du groupe **doivent** prendre la parole de façon équilibrée
- Contenu attendu
  - Rappel du contexte (besoin, organisation, etc.)
  - Vos grands choix de conception/structures de données, avec les arguments clés
  - Ce qui fonctionne et ne fonctionne pas
  - Captures d'écran
  - Courte démonstration en option
  - **Bilan technique** (principales difficultés rencontrées, etc.). Pas de considérations personnelles et subjectives.
- Conseils :
  - Pas de diapo vide
  - Évitez de mettre du code
  - Privilégiez les schémas, en ne gardant que l'information pertinente
  - Tout ce qui est présent dans une diapo doit être utile (et présenté). Sinon, épurez !

Notation

- Barème indicatif :
  - Phase 1 : étude préliminaire : 25%
  - Rendu de code final : 50%
  - Présentation : 25%

- Remarques :
  - malus possible sur des membres du groupe si l'investissement est jugé trop faible
  - possibilité d'être interrogé durant le projet de façon individuelle
  - plagiat sévèrement sanctionné pour TOUS les membres du/des groupe(s)
  - Utilisation de ChatGPT, CoPilot interdit. (possibilité de conseil de discipline)
  - Le code sera évalué uniquement d'un point de vue fonctionnel. Toutefois, des mauvaises pratiques significatives seront sanctionnées par des malus.
  - Commentez votre code (-25% sur la note si mal commenté)

Mises en garde

- Le ou les encadrants ne font pas le code à votre place. **Ils vous aident** éventuellement **à corriger vos bugs** et à trouver vos erreurs.
- **Votre code est unique et vous devez le comprendre**. Copier/coller le code source sans le citer d'un autre groupe vous expose, ainsi que vos camarades, à une sanction dans la note du projet.
- **Sauvegardez régulièrement** votre travail. Dès que vous avez une version opérationnelle, mettez la de côté au cas où…
- **Ne misez pas tout sur le mercredi** pour terminer le travail (dernier jour réservé aux finitions et commentaires).
- **Validez étape par étape**, découpez votre code en fonctions, testez le plus souvent possible. Réfléchissez à des solutions partielles pour pouvoir avancer.
- Ce projet pour être mené à bien nécessite **un investissement personnel et de groupe,** et de nombreux tests.
- malus.
- Ne pas se tromper de page **moodle** (Brest+Paris / Rennes / Caen / Nantes)

Questions

Des questions ?