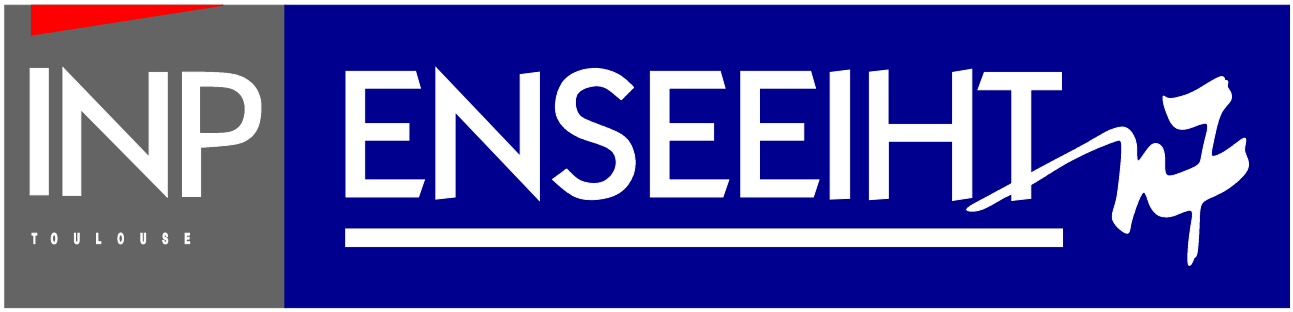


Rapport du projet de programmation impérative

Auteurs:

Amine Akby
Badr Sajid

Encadrant :

Pr. Cregut Xavier

Version 0.1
13 decembre 2019

Sommaire

I – Objectif et contenu du rapport :

L’objectif du projet est de faire le point sur les compétences et les connaissances acquises lors des séances de programmation impérative.

Ce rapport a pour objet de présenter l’ensemble des taches effectuées , d’expliquer les choix réalisés lors du projet , de montrer les difficultés rencontrées et l’organisation du groupe pour faire avancer le projet , et de présenter les améliorations prévues pour optimiser le fonctionnement du programme.

II – Introduction au problème traité :

Un arbre généalogique d’un individu est une structure de données qui permet de répertorier l’ensemble des ancêtres connus de cet individu. Le but du projet est de réaliser un système qui permet de gérer ces arbres génealogiques de façon à faciliter à l’utilisateur de trouver les ancêtres d’un individu de n’importe quelle génération qui vérifie un critère choisi. Chaque

individu sera représenter par un identifiant. Un registre d’etat civil contiendra les informations de tous les individus. On peut accéder à ces informations par le biais des identifiants dans le registre.

III – Architecture logicielle définie des modules :

On va définir 4 modules pour faciliter la compréhension et la gestion des algorithmes implantés par la suite :

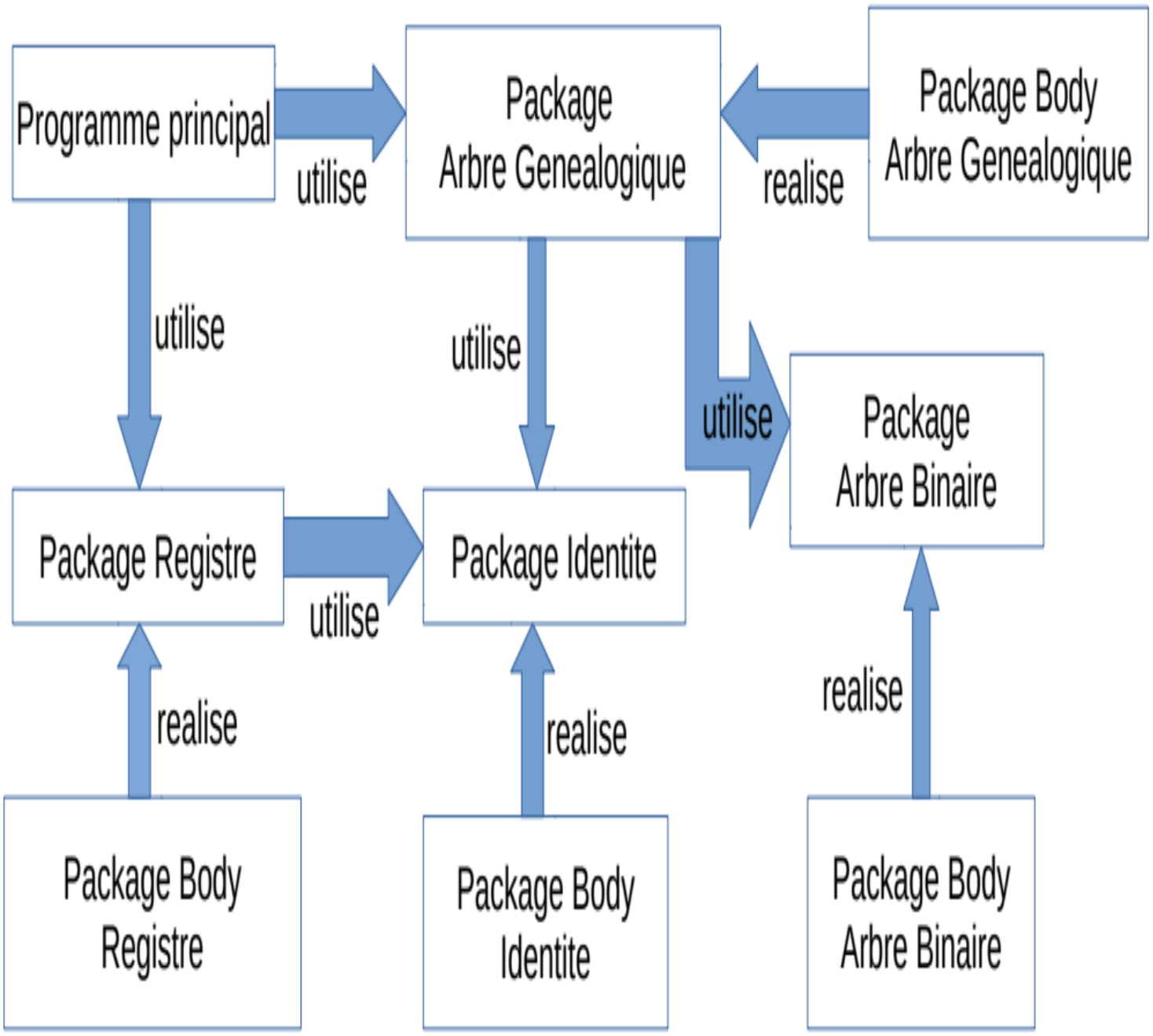
- Module Arbre Binaire : Ce module contient une structure de données et des programmes qui permettent de gérer un arbre binaire simple.

-Module Arbre Généalogique : Ce module se base principalement sur le module Arbre Binaire pour gérer un arbregénéalogique d’un individu.

-Module Registre : Ce module contient une structure de données qui se base sur le module Identité et des programmes qui permettent de gérer un registre d’etat civil qui contient les informations des tous les individus.

-Module Identité : Ce module permet de gérer les informations (nom, prénom , âge) d’un seul individu connu par un identifiant.

Architecture logicielle définie



IV – Principaux choix réalisés :

On a choisi de définir 4 modules liés entres eux comme expliquer dans l'architecture des modules. Chaque module contient une structure de donnée qui permet de gérer une partie du projet.

Dans la partie gestion des arbres , on a décidé de créer une structure usuelle et simplifiée d'un arbre binaire et implanter quelques programmes basiques , ce module sera instantier par la suite dans le package principal Arbre genealogique.

Dans la partie gestion des informations des individus , on a pensé à créer d'abord un module Identité qui contient une structure qui gère les informations d'un seul individu , et ensuite , on se basant sur ce module , on crée le module principal Registre qui gère les informations de toutes les personnes.

V – Algorithmes et Types de données :

Arbre Binaire :

- Ce module permet de gérer une structure d'arbre binaire. On commence par définir un type T_Cellule_Arbre de l'arbre binaire, et puis on définit un type T_arbre qui est un pointeur de type T_Cellule_Arbre. Le type T_Cellule_Arbre est un enregistrement d'une clé de type T_cle , un arbre gauche et une arbre droit de types T_arbre. Les algorithmes écrits dans ce module sont élementaires (Initialiser l'arbre , Ajouter une élement , Supprimer un élement , Savoir si un élement est présent , Calculer la hauteur et la taille de l'arbre , Trouver des feuilles de l'arbre qui vérifient une condition , Afficher l'arbre).

Arbre Généalogique : (AG)

- On instantie le module précédent dans ce module en changeant le type T_cle par un type T_identifiant (un type qui gère les identifiants des individus) . On crée un nouveau type T_AG pour les arbres généalogiques à partir du type T_arbre. Les algorithmes définis dans ce module permettent de réaliser les opérations que l'utilisateur veut effectuer sur l'arbre

généalogique (Initialiser un AG , Ajouter des individus à l'AG , Supprimer des individus de l'AG , Trouver les ancêtres d'un individu de toutes les générations ou d'une génération précise , Trouver les individus qui n'ont qu'un parent connu ou bien qui ont deux ou aucun parents connus , Afficher l'AG , Detruire l'AG). Ces algorithmes seront implanter en utilisant les algorithmes déjà écrits dans le module Arbre Binaire.

Identité :

- Ce module permet de gérer une structure ce carte d'identité d'un individu. On définit un Type T_Cellule_Identite puis un tpe T_idendite qui est un pointeur de T_Cellule_Identite. Le type T_Cellule_Identite est un enregistrement d'une cle (nom , prenom , age , …) , une information qui correspond à cette clé , et Suivant qui est de type T_identite. De tel sorte que la structure soit une liste chainée des informations d'un individu pour la gestion de mémoire. Les algorithmes du module sont très élementaires qui permettent une simple utilisation (Initialiser une carte d'identité , Savoir si une information existe dans la carte , Modifier une information dans la carte , Ajouter

une nouvelle information , Trouver le nombre d’informations dans une carte , Supprimer une information , Afficher la carte d’identité).

Registre :

- Ce module importe le module Identité. L’idée de ce module est de créer une structure de liste chainée , où chaque élement est un identifiant de type T_identifiant qui représente un individu et une carte d’identité de type T_identite. Les algorithmes de ce module se basent sur les algorithmes du module Identité.

VI – Tests des modules :

Arbre Binaire et Arbre Généalogique :

- On instantie d’abord le module à tester en choisissant par exemple comme type T_identifiant des entiers . Pour tester les modules arbre binaire et arbre généalogique , on initialise un arbre , on ajoute la racine de l’arbre , puis on ajoute des élements aux arbres gauches et droits. On teste le module en affichant les ancêtres d’un

individu à une génération précise , en affichant l’arbre à chaque fois qu’on le change.

Registre et Identité:

- On teste le registre et l’identité en ajoutant plusieurs identifiants avec leurs cartes identifiants. On affiche le registre à chaque fois pour visualiser les changements. On essaye de supprimer des individus , de modifier leurs cartes d’identité et de les afficher , etc. On fait à fûr et à mesure des Pragma Assert pour tester les algorithmes Taille , Est_present , Est_vide , etc.

VII – Organisation et Difficultés :

- Comme on a pu choisir les groupes , on a pas eu de problèmes dans l’organisation. On a commencé par se mettre d’accord sur les modules à réaliser , les structures de données , et les types de donnés . Après , on a diviser les tâches à réaliser. Chacun de nous a pris deux modules non liés entre eux. Un a pris les modules Arbre Généalogique et Registre (Amine Akby) et l’autre a pris le modules Arbre Binaire et Identité (Badr Sajid). On était toujours en contact surtout que

les modules de l’un dépendent des modules de l’autre. On devait se mettre d’accord sur les noms des algorithmes , des types , etc pour ne pas se confuser. On a travaillé tous les deux sur le rapport et le manuel utilisateur et le programme principal car ils demandent une vision globale du projet.

-Le première difficulté est le temps : On avait peu de temps pour travailler le projet surtout dans la période des examens. Il y’avait d’autres difficultés dans l’importation des types et de surcharge quand on instantie le module Arbre Binaire dans Arbre Généalogique.

VII – Avancement et Amélioration prévue :

- On a bien avancé dans la première partie du projet. Les modules semblent bien marcher , tous les tests des modules passent. On a fini le programme principal avec plusieurs fonctionnalités et des exceptions bien gérées. On essaye d’ajouter de nouvelles fonctionnalités pour optimiser la gestion des arbres généalogiques.

VII – Améliorations réalisées:

- On a ajouté plusieurs options pour la gestion des arbres généalogiques :

-la possibilité d’initialiser un arbre plein pour facilliter le test des programmes du menu.

-la creation d’un registre rempli par des informations qui sont générés aléatoirement (nom,prenom,ville) sauf l’age qui sera representer par l’identifiant de l’individu pour faciliter l’ajout des parents d’un individu , car on a constater que lorsqu’on choisit l’age aleatoirement , l’utilisateur qui ne connaît pas l’age des individus bloque toujours dans l’ajout des parents d’une personne (l’age des parents doit être superieur à l’age du fils).

-La possibilité d’ajouter d’autres individus au registre d’etat civil.

-La gestion des exceptions pour que le programme ne s’arrete que si l’utilisateur decide de quitter. On donne toujours à l’utilisateur la possibilité de rentrer des informations adéquates.

-La spécification et le debut de l’implantation de la partie 2.