Rapport de Projet

Projet d'Algorithmique S6

RÉALISÉ PAR

CATALA Alexandre - BOUCHUT Tom - GOUTAL GUERIN Yanis - MONTANARI Charlotte

ISEN ALL IS DIGITAL! MÉDITERRANÉE | yncréa

ORGANISATION DE L'ÉQUIPE EN FACE DU PROJET

- o Découpage en tâches et sous-tâches
- o Affectation des tâches aux membres de l'équipe
- o Déroulement de la phase de développement planning prévisionnel / planning réalisé

Tâches	Sous-Tâches	Membre(s) de l'équipe	Planning prévisionnel	Planning réalisé
Conception du projet	Définition des objectifs	Tous	11 mars - 12 mars	11-mars
	Élaboration du plan de projet	Tous	11 mars - 12 mars	12-mars
Développement	Création de la structure de base du code	Tous	13 mars - 14 mars	13-mars
	Implémentation des fonctions de lecture	Alexandre, Tom	13 mars - 14 mars	13-mars
	Implémentation des fonctions de gestion de la mémoire	Alexandre, Tom	13 mars - 14 mars	14-mars
	Implémentation des fonctions de manipulation des faits	Alexandre, Tom	13 mars - 14 mars	14-mars
	Implémentation des fonctions de manipulation des caractères	Yanis, Charlotte	13 mars - 14 mars	14-mars
	Implémentation du chaînage avant	Yanis, Charlotte	13 mars - 14 mars	13-mars
	Implémentation du chaînage arrière	Yanis, Charlotte	13 mars - 14 mars	14-mars
Test et débogage	Test des fonctions individuelles	Tous	14 mars - 16 mars	15-mars
	Test du système dans son ensemble	Tous	16 mars - 17 mars	16-mars
	Débogage et résolution des problèmes	Tous	16 mars - 17 mars	17-avr
Documentation et présentation	Rédaction de la documentation / Excel	Tous	18 mars - 20 mars	18 mars - 20 mars

Fichier	Exigences	Plan de Test	Exemples de Tests
main.c	Fournit le menu interactif et gère l'exécution globale du programme.	Tester le menu interactif et l'exécution globale du programme.	Exécuter le programme et naviguer dans le menu interactif.
fact.c	Gère les opérations sur les faits, y compris la création, l'ajout et la libération des faits.	Tester la création, l'ajout et la libération des faits.	Créer un fait, l'ajouter à la liste des faits, puis libérer la liste des faits.
chainage_avant.c	Implémente le chaînage avant. Les exemples donnés ont été vérifiés et validés.	Tester le chaînage avant avec différents ensembles de règles et de faits.	Exécuter le chaînage avant avec un ensemble de règles et de faits et vérifier que les nouveaux faits sont correctement ajoutés.
chainage_arriere.c	Implémente le chaînage arrière. Les exemples donnés ont été vérifiés et validés.	Tester le chaînage arrière avec différents ensembles de règles et de faits.	Exécuter le chaînage arrière avec un ensemble de règles et de faits et vérifier que le fait à prouver peut être prouvé.
lecture.c	Implémente les fonctions de lecture	Tester l'ouverture de fichier	Exécuter l'ouverture de fichier et vérifier si aucune erreur n'est rencontré

LE RAPPORT QUI SUIT RÉPONDRA À LA CONSIGNE 7, UN DOCUMENT DE VALIDATION DES EXIGENCES POUR LE NIVEAU 3 :

- o Chainage avant ✓
- o Chainage arrière optionnel ✓
- o Exemple d'une base de connaissance ✓
- o Architecture en groupe ✓
- o Découpage et attribution des tâches par binômes dans l'équipe ✓

```
|-----------------------------------------------------|
Choisissez d'effectuer du chainage avant ou arriere :

(0) Quitter
(1) Chainage avant
(2) Chainage arrière
(3) Solution Easter Egg
Votre choix: 1

Pour le chainage avant, vous disposez d'une grande base règles (+ de 1500 comprenant des caractères spéciaux). Alors n'hési
tez pas à tester beaucoup de faits ! (Easter egg : essayez d'obtenir le +)

Combien de faits voulez-vous ajouter ? 3
Saisir un fait: a
Saisir un fait: 3
Saisir un fait: c
Faits insérés: a -> 3 -> c -> NULL
Saisir le but à atteindre: +
Le but "+" est atteint !
Chainage avant: a -> 3 -> c -> L -> X -> m -> 5 -> W -> 6 -> 7 -> Y -> 8 -> V -> @ -> G -> ? -> $ -> ! -> 2 -> D -> F -> H
-> I -> K -> P -> d -> = -> e -> 4 -> f -> g -> q -> R -> h -> i -> M -> E -> O -> r -> Q -> p -> j -> k -> T -> 1 -> S ->
Z -> 0 -> A -> J -> 9 -> % -> & -> # -> B -> U -> t -> w -> y -> o -> N -> b -> l -> C -> n -> s -> u -> v -> x -> z -> + -
> NULL
```

```
|-----------------------------------------------------|
Choisissez d'effectuer du chainage avant ou arriere :

(0) Quitter
(1) Chainage avant
(2) Chainage arrière
(3) Solution Easter Egg
Votre choix: 2

Pour le chainage arriere, la base de règles disponible est celle du cahier des charges (i.e :b d e -> f)

Combien de faits voulez-vous ajouter ? 2
Saisir un fait: b
Saisir un fait: c
Faits insérés: b -> c -> NULL
Saisir le but à atteindre: h
Le but "g" n'est pas atteint
Le but "d" est atteint !
Le but "f" est atteint !
Le but "a" est atteint !
Le but "h" est atteint !
```

1 Contenu du Projet **1**

1.1 src . . . 1

1.2 include . . . 1

1.3 kbs . . . 2

1.4 Documentation . . . 2

1.5 build . . . 2

1.6 Compilation et Exécution . . . 2

1.7 Exemple d'Utilisation . . . 2

1.8 Gestion du Projet . . . 2

1.8.1 Création d'un GitHub . . . 2

1.8.2 Déroulement du Projet . . . 3

2 Data Structure Index **5**

2.1 Data Structures . . . 5

3 File Index **7**

3.1 File List . . . 7

4 Data Structure Documentation **9**

4.1 Faits Struct Reference . . . 9

4.1.1 Detailed Description . . . 9

4.1.2 Field Documentation . . . 9

4.1.2.1 identifiant . . . 9

4.1.2.2 nombre_faits . . . 9

4.1.2.3 suiv . . . 9

5 File Documentation **11**

5.1 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/caractere.h File Reference . . . 11

5.1.1 Detailed Description . . . 11

5.1.2 Function Documentation . . . 12

5.1.2.1 AjouterCaractereSiAbsent() . . . 12

5.1.2.2 CaractereDansListe() . . . 12

5.2 caractere.h . . . 13

5.3 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/chainage_arriere.h File Reference . . . 13

5.3.1 Detailed Description . . . 14

5.3.2 Function Documentation . . . 14

5.3.2.1 ChainageArriere() . . . 14

5.4 chainage_arriere.h . . . 15

5.5 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/chainage_avant.h File Reference . . . 15

5.5.1 Detailed Description . . . 15

5.5.2 Function Documentation . . . 16

5.5.2.1 ChainageAvant() . . . 16

5.6 chainage_avant.h . . . 16

5.7 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/fait.h File Reference . . . 17

5.7.1 Detailed Description . . . 17
5.7.2 Typedef Documentation . . . 18
5.7.2.1 Faits . . . 18
5.7.3 Function Documentation . . . 18
5.7.3.1 AfficherFaits() . . . 18
5.7.3.2 AjouterFaits() . . . 18
5.7.3.3 AjouterFaitsSuite() . . . 19
5.7.3.4 CreerFaits() . . . 19
5.7.3.5 FaitPresent() . . . 19
5.7.3.6 InsererFaits() . . . 20
5.7.3.7 LibererFaits() . . . 20
5.8 fait.h . . . 21
5.9 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/lecture.h File Reference . . . 21
5.9.1 Detailed Description . . . 22
5.9.2 Function Documentation . . . 22
5.9.2.1 ouvrir_fichier() . . . 22
5.10 lecture.h . . . 22
5.11 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/memoire.h File Reference . . . 23
5.11.1 Detailed Description . . . 23
5.11.2 Function Documentation . . . 23
5.11.2.1 allouer_malloc() . . . 23
5.12 memoire.h . . . 24
5.13 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/menu.h File Reference . . . 24
5.13.1 Detailed Description . . . 25
5.13.2 Function Documentation . . . 25
5.13.2.1 menu() . . . 25
5.14 menu.h . . . 25
5.15 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/README.md File Reference . . . 26
5.16 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/caractere.c File Reference . . . 26
5.16.1 Detailed Description . . . 26
5.16.2 Function Documentation . . . 26
5.16.2.1 AjouterCaractereSiAbsent() . . . 26
5.16.2.2 CaractereDansListe() . . . 27
5.17 caractere.c . . . 27
5.18 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/chainage_arriere.c File Reference . . . 28
5.18.1 Detailed Description . . . 28
5.18.2 Function Documentation . . . 29
5.18.2.1 ChainageArriere() . . . 29
5.19 chainage_arriere.c . . . 29
5.20 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/chainage_avant.c File Reference . . . 30
5.20.1 Detailed Description . . . 30
5.20.2 Function Documentation . . . 31

5.20.2.1 ChainageAvant() . . . 31
5.21 chainage_avant.c . . . 31
5.22 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/fait.c File Reference . . . 32
5.22.1 Detailed Description . . . 33
5.22.2 Function Documentation . . . 33
5.22.2.1 AfficherFaits() . . . 33
5.22.2.2 AjouterFaits() . . . 33
5.22.2.3 AjouterFaitsSuite() . . . 34
5.22.2.4 CreerFaits() . . . 34
5.22.2.5 FaitPresent() . . . 34
5.22.2.6 InsererFaits() . . . 35
5.22.2.7 LibererFaits() . . . 35
5.23 fait.c . . . 35
5.24 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/lecture.c File Reference . . . 37
5.24.1 Detailed Description . . . 37
5.24.2 Function Documentation . . . 37
5.24.2.1 ouvrir_fichier() . . . 37
5.25 lecture.c . . . 38
5.26 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/main.c File Reference . . . 38
5.26.1 Detailed Description . . . 39
5.26.2 Function Documentation . . . 39
5.26.2.1 main() . . . 39
5.27 main.c . . . 40
5.28 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/memoire.c File Reference . . . 40
5.28.1 Detailed Description . . . 41
5.28.2 Function Documentation . . . 41
5.28.2.1 allouer_malloc() . . . 41
5.29 memoire.c . . . 41
5.30 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/menu.c File Reference . . . 42
5.30.1 Detailed Description . . . 42
5.30.2 Function Documentation . . . 43
5.30.2.1 menu() . . . 43
5.31 menu.c . . . 43

Index **45**

Chapter 1

Contenu du Projet

1.1 src

Ce répertoire contient le code source de l'application.

- **main.c** (p. 38) : Fichier principal contenant le menu interactif.
- **lecture.c** (p. 37) : Fichier contenant les fonctions pour la lecture des fichiers.
- **memoire.c** (p. 40) : Fichier contenant les fonctions pour la gestion de la mémoire.
- **fait.c** (p. 32) : Fichier contenant les opérations à effectuer sur les faits.
- **caractere.c** (p. 26) : Fichier contenant les opérations à effectuer sur les caractères.
- **chainage_avant.c** (p. 30) : Fichier contenant la fonction pour le chaînage avant.
- **chainage_arriere.c** (p. 28) : Fichier contenant la fonction pour le chaînage arrière.
- **menu.c** (p. 42) : Fichier contenant contenant la procédure pour créer le menu.

1.2 include

Ce répertoire contient les fichiers d'en-tête.

- **lecture.h** (p. 21) : Définition des fonctions pour la lecture des fichiers.
- **memoire.h** (p. 23) : Définition des fonctions pour la gestion de la mémoire.
- **fait.h** (p. 17) : Définition de la structure **Faits** (p. 9) et des fonctions associées.
- **caractere.h** (p. 11) : Définition des fonctions pour la manipulation des caractères.
- **chainage_avant.h** (p. 15) : Définition de la fonction pour le chaînage avant.
- **chainage_arriere.h** (p. 13) : Définition de la fonction pour le chaînage arrière.
- **menu.h** (p. 24) : Définition de la fonction pour le menu.

1.3 kbs

Ce répertoire contient les fichiers .kbs utilisés pour les tests, incluant une base de données représentant le cryptage que nous avons choisi comme sujet pour expérimenter les différents chainages (avant et arriere).

1.4 Documentation

Ce répertoire contient la documentation du projet.

1.5 build

Ce répertoire contient les fichiers générés lors de la compilation.

1.6 Compilation et Exécution

1. Cloner le dépôt.
2. Lancer la commande `make clean && make run` dans votre terminal en étant à la racine du projet.

1.7 Exemple d'Utilisation

1. Sélectionnez l'option dans le menu interactif.
2. Suivez les instructions pour fournir les fichiers .kbs nécessaires.
3. Obtenez les résultats dans votre terminal.

1.8 Gestion du Projet

1.8.1 Création d'un GitHub

Pour nous organiser, nous avons mis en place un dépôt GitHub dans lequel nous avons chacun notre « branche » afin de pouvoir travailler de manière indépendante sur le projet. Nous pouvions donc suivre les avancements de chacun, et avancer en autonomie.

1.8.2 Déroulement du Projet

1. **Compréhension du Sujet, des Exigences, et du Format .kbs:** Nous avons dans un premier temps commencé à travailler sur la compréhension de la logique derrière les systèmes experts. Ensuite, il a fallu qu'on comprenne les principes de chaînage avant / arrière pour qu'on arrive à saisir ses subtilités et coder en C. On a aussi beaucoup bloqué pour récupérer et stocker les règles à partir d'un fichier .kbs.

2. **Blocage:** Nous nous sommes rendus compte au cours de notre projet que notre approche n'était pas optimale. En effet, nos fonctions géraient beaucoup de choses en même temps. Par exemple, notre fonction qui convertissait les .kbs en règles et faits s'occupait à la fois :

   - de l'ouverture des fichiers
   - de la gestion de la mémoire
   - de la conversion des données
   - du calcul et de l'écriture des données dans les structures

   Suite à ces problèmes, nous avons décidé de repartir du début pour repartir sur une base plus adaptée. Nous avons essayé de séparer le plus possible le code en plusieurs fonctions pour éviter les erreurs et les répétitions.

3. **Répartition des Tâches:** Nous avons travaillé en binôme pour diviser les tâches et travailler de manière plus efficace. Alexandre et Tom se sont occupés des fonctions de lecture, de gestion de la mémoire, des faits et des caractères. Yanis et Charlotte ont travaillé sur les chaînages avant et arrière. Cette répartition nous a permis de travailler de manière plus organisée et de progresser plus rapidement.

4. **Finalisation:** Nous avons recommencé plusieurs fois avant d'obtenir une version stable de notre projet. Chaque fois, nous avons appris de nos erreurs et amélioré notre code.

Merci de votre attention et bonne utilisation de notre système expert !

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Faits

Structure pour représenter un fait 9

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/ **caractere.h**
Fichier d'en-tête pour les fonctions de manipulation des caractères . . . 11
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/ **chainage_arriere.h**
Fichier d'en-tête pour la fonction de chaînage arrière . . . 13
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/ **chainage_avant.h**
Fichier d'en-tête pour la fonction de chaînage avant . . . 15
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/ **fait.h**
Fichier d'en-tête pour les fonctions de manipulation des faits . . . 17
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/ **lecture.h**
Fichier d'en-tête pour la lecture de fichier et le stockage de données . . . 21
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/ **memoire.h**
Fichier d'en-tête pour la gestion de la mémoire . . . 23
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/ **menu.h**
Fichier d'en-tête pour la fonction de menu . . . 24
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/ **caractere.c**
Fichier **caractere.c** (p. 26) contenant l'ensemble des fonctions pour les opérations concernant les opérations sur les caractères . . . 26
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/ **chainage_arriere.c**
Fichier **chainage_arriere.c** (p. 28) contenant l'ensemble des procédures pour le chainage arrière 28
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/ **chainage_avant.c**
Fichier **chainage_avant.c** (p. 30) contenant l'ensemble des procédures pour le chainage avant 30
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/ **fait.c**
Fichier **fait.c** (p. 32) contenant l'ensemble des procédures pour la gestion des faits . . . 32
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/ **lecture.c**
Fichier **lecture.c** (p. 37) contenant l'ensemble des fonctions pour la lecture de fichier et de données .kbs . . . 37
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/ **main.c**
Fichier **main.c** (p. 38) contenant le code qui sera éxécuté par l'utilisateur . . . 38
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/ **memoire.c**
Fichier **memoire.c** (p. 40) contenant l'ensemble des fonctions pour les opérations concernant la mémoire . . . 40
C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/ **menu.c**
Fichier **menu.c** (p. 42) contenant la procédure pour créer le menu . . . 42

Chapter 4

Data Structure Documentation

4.1 Faits Struct Reference

Structure pour représenter un fait.

`#include <fait.h>`

Data Fields

- int **nombre_faits**
- char * **identifiant**
- struct **Faits** * **suiv**

4.1.1 Detailed Description

Structure pour représenter un fait.

Cette structure contient un identifiant pour le fait, le nombre de faits et un pointeur vers le fait suivant dans la liste.

Definition at line **23** of file **fait.h**.

4.1.2 Field Documentation

4.1.2.1 identifiant

`char* identifiant`

Definition at line **26** of file **fait.h**.

4.1.2.2 nombre_faits

`int nombre_faits`

Definition at line **25** of file **fait.h**.

4.1.2.3 suiv

`struct Faits* suiv`

Definition at line **27** of file **fait.h**.

The documentation for this struct was generated from the following file:

- C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/ **fait.h**

Chapter 5

File Documentation

5.1 C:/Users/Alexandre/OneDrive/Bureau/projetalgo↩ S6/include/caractere.h File Reference

Fichier d'en-tête pour les fonctions de manipulation des caractères.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "fait.h"
```

Functions

- int **CaractereDansListe** (**Faits** ∗tete, char caractere)
  Vérifie si un caractère est présent dans une liste de faits.
- **Faits** ∗ **AjouterCaractereSiAbsent** (**Faits** ∗liste, char caractere)
  Ajoute un caractère à une liste de faits s'il n'est pas déjà présent.

5.1.1 Detailed Description

Fichier d'en-tête pour les fonctions de manipulation des caractères.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Definition in file **caractere.h**.

5.1.2 Function Documentation

5.1.2.1 AjouterCaractereSiAbsent()

```
Faits * AjouterCaractereSiAbsent (
            Faits * liste,
           char caractere )
```

Ajoute un caractère à une liste de faits s'il n'est pas déjà présent.

Parameters

liste	Un pointeur vers le premier élément de la liste de faits.
caractere	Le caractère à ajouter à la liste de faits.

Returns

Faits∗ Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Parameters

liste	Pointeur vers le premier élément de la liste de faits.
caractere	Le caractère à ajouter à la liste de faits.

Returns

Faits∗ Retourne un pointeur vers le premier élément de la liste de faits

Definition at line **48** of file **caractere.c**.

5.1.2.2 CaractereDansListe()

```
int CaractereDansListe (
            Faits * tete,
           char caractere )
```

Vérifie si un caractère est présent dans une liste de faits.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.
caractere	Le caractère à rechercher dans la liste de faits.

Returns

int Retourne 1 si le caractère est trouvé dans la liste, sinon retourne 0.

Parameters

tete	Pointeur vers le premier élément de la liste de faits.
caractere	Le caractère à rechercher dans la liste de faits.

Returns

int Retourne 1 si le caractère est trouvé dans la liste, sinon retourne

Definition at line **24** of file **caractere.c**.

5.2 caractere.h

Go to the documentation of this file.

```
00001
00012 #ifndef CARACTERE_H
00013 #define CARACTERE_H
00014
00015 #include <stdio.h>
00016 #include <stdlib.h>
00017 #include <string.h>
00018 #include "fait.h"
00019
00027 int CaractereDansListe(Faits *tete, char caractere);
00028
00036 Faits *AjouterCaractereSiAbsent(Faits *liste, char caractere);
00037
00038 #endif // CARACTERE_H
```

5.3 C:/Users/Alexandre/OneDrive/Bureau/projetalgo↩ S6/include/chainage_arriere.h File Reference

Fichier d'en-tête pour la fonction de chaînage arrière.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "fait.h"
#include "caractere.h"
```

Functions

- int **ChainageArriere** (const char *file, **Faits** *liste_faits, char *but)

  Effectue un chaînage arrière sur une liste de faits pour atteindre un but.

5.3.1 Detailed Description

Fichier d'en-tête pour la fonction de chaînage arrière.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Copyright (c) 2024

Definition in file **chainage_arriere.h**.

5.3.2 Function Documentation

5.3.2.1 ChainageArriere()

```
int ChainageArriere (
            const char * file,
             Faits * liste_faits,
            char * but )
```

Effectue un chaînage arrière sur une liste de faits pour atteindre un but.

Cette fonction lit un fichier de règles et utilise un algorithme de chaînage arrière pour déterminer si un but peut être atteint à partir d'une liste de faits initiale. Le but est atteint si toutes les prémisses d'une règle qui a le but comme conséquence sont présentes dans la liste de faits.

Parameters

file	Le nom du fichier contenant les règles à utiliser pour le chaînage arrière.
liste_faits	Un pointeur vers le premier élément de la liste de faits initiale.
but	La chaîne de caractères représentant le but à atteindre.

Returns

int Retourne 1 si le but est atteint, sinon retourne 0.

Definition at line **28** of file **chainage_arriere.c**.

5.4 chainage_arriere.h

Go to the documentation of this file.

```
00001
00011 #ifndef CHAINAGE_ARRIERE_H
00012 #define CHAINAGE_ARRIERE_H
00013
00014 #include <stdio.h>
00015 #include <stdlib.h>
00016 #include <string.h>
00017 #include "fait.h"
00018 #include "caractere.h"
00019
00030 int ChainageArriere(const char *file, Faits *liste_faits, char *but);
00031
00032 #endif // CHAINAGE_ARRIERE_H
```

5.5 C:/Users/Alexandre/OneDrive/Bureau/projetalgo↩ S6/include/chainage_avant.h File Reference

Fichier d'en-tête pour la fonction de chaînage avant.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "fait.h"
#include "caractere.h"
```

Functions

- **Faits** ∗ **ChainageAvant** (const char ∗file, **Faits** ∗liste_faits)

  Effectue un chaînage avant sur une liste de faits.

5.5.1 Detailed Description

Fichier d'en-tête pour la fonction de chaînage avant.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Copyright (c) 2024

Definition in file **chainage_avant.h**.

5.5.2 Function Documentation

5.5.2.1 ChainageAvant()

```
Faits * ChainageAvant (
            const char * file,
             Faits * liste_faits )
```

Effectue un chaînage avant sur une liste de faits.

Cette fonction lit un fichier de règles et utilise un algorithme de chaînage avant pour déterminer si un but peut être atteint à partir d'une liste de faits initiale. Le but est atteint si toutes les conséquences d'une règle qui a le but comme prémisse sont présentes dans la liste de faits.

Parameters

file	Le nom du fichier contenant les règles à utiliser pour le chaînage avant.
liste_faits	Un pointeur vers le premier élément de la liste de faits initiale.

Returns

Faits* Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Effectue un chaînage avant sur une liste de faits.

Cette fonction lit un fichier de règles et utilise un algorithme de chaînage avant pour déterminer si un but peut être atteint à partir d'une liste de faits initiale. Le but est atteint si toutes les conséquences d'une règle qui a le but comme prémisse sont présentes dans la liste de faits.

Parameters

file	Le nom du fichier contenant les règles à utiliser pour le chaînage avant.
liste_faits	Un pointeur vers le premier élément de la liste de faits initiale.

Returns

Faits* Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Definition at line **26** of file **chainage_avant.c**.

5.6 chainage_avant.h

Go to the documentation of this file.

```
00001
00011 #ifndef CHAINAGE_AVANT_H
00012 #define CHAINAGE_AVANT_H
00013
00014 #include <stdio.h>
00015 #include <stdlib.h>
00016 #include <string.h>
00017 #include "fait.h"
00018 #include "caractere.h"
00019
00029 Faits *ChainageAvant(const char *file, Faits *liste_faits);
00030
00031 #endif // CHAINAGE_AVANT_H
```

5.7 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/fait.h File Reference

Fichier d'en-tête pour les fonctions de manipulation des faits.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Data Structures

- struct **Faits**
  Structure pour représenter un fait.

Typedefs

- typedef struct **Faits** **Faits**
  Structure pour représenter un fait.

Functions

- **Faits** ∗ **CreerFaits** ()
  Crée un nouveau fait.
- **Faits** ∗ **InsererFaits** (**Faits** ∗tete, **Faits** ∗f)
  Insère un fait dans une liste de faits.
- **Faits** ∗ **AjouterFaits** ()
  Ajoute des faits à une liste de faits.
- **Faits** ∗ **AjouterFaitsSuite** (**Faits** ∗liste, const char ∗nouveauFait)
  Ajoute un fait à la suite d'une liste de faits.
- void **AfficherFaits** (**Faits** ∗tete)
  Affiche une liste de faits.
- void **LibererFaits** (**Faits** ∗tete)
  Libère la mémoire allouée à une liste de faits.
- int **FaitPresent** (**Faits** ∗liste_faits, const char ∗but)
  Vérifie si un fait est présent dans une liste de faits.

5.7.1 Detailed Description

Fichier d'en-tête pour les fonctions de manipulation des faits.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Definition in file **fait.h**.

5.7.2 Typedef Documentation

5.7.2.1 Faits

```
typedef struct  Faits  Faits
```

Structure pour représenter un fait.

Cette structure contient un identifiant pour le fait, le nombre de faits et un pointeur vers le fait suivant dans la liste.

5.7.3 Function Documentation

5.7.3.1 AfficherFaits()

```
void AfficherFaits (
            Faits * tete )
```

Affiche une liste de faits.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.

Cette fonction parcourt une liste de faits et affiche l'identifiant de chaque fait.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.

Definition at line **99** of file **fait.c**.

5.7.3.2 AjouterFaits()

```
Faits * AjouterFaits ( )
```

Ajoute des faits à une liste de faits.

Returns

Faits∗ Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Cette fonction demande à l'utilisateur combien de faits il souhaite ajouter, puis ajoute ces faits à la liste.

Returns

Faits∗ Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Definition at line **58** of file **fait.c**.

5.7.3.3 AjouterFaitsSuite()

```
Faits * AjouterFaitsSuite (
            Faits * liste,
           const char * nouveauFait )
```

Ajoute un fait à la suite d'une liste de faits.

Parameters

liste	Un pointeur vers le premier élément de la liste de faits.
nouveauFait	Une chaîne de caractères représentant l'identifiant du nouveau fait à ajouter.

Returns

Faits* Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Cette fonction crée un nouveau fait avec l'identifiant spécifié et l'insère à la fin de la liste de faits donnée.

Parameters

liste	Un pointeur vers le premier élément de la liste de faits.
nouveauFait	Une chaîne de caractères représentant l'identifiant du nouveau fait à ajouter.

Returns

Faits* Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Definition at line **85** of file **fait.c**.

5.7.3.4 CreerFaits()

```
Faits * CreerFaits ( )
```

Crée un nouveau fait.

Returns

Faits* Retourne un pointeur vers le nouveau fait créé.

Cette fonction alloue de la mémoire pour un nouveau fait et initialise ses membres.

Returns

Faits* Retourne un pointeur vers le nouveau fait créé.

Definition at line **24** of file **fait.c**.

5.7.3.5 FaitPresent()

```
int FaitPresent (
            Faits * liste_faits,
           const char * but )
```

Vérifie si un fait est présent dans une liste de faits.

Parameters

liste_faits	Un pointeur vers le premier élément de la liste de faits.
but	Une chaîne de caractères représentant le fait à rechercher.

Returns

int Retourne 1 si le fait est trouvé dans la liste, sinon retourne 0.

Definition at line **129** of file **fait.c**.

5.7.3.6 InsererFaits()

```
Faits * InsererFaits (
            Faits * tete,
            Faits * f )
```

Insère un fait dans une liste de faits.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.
f	Un pointeur vers le fait à insérer.

Returns

Faits∗ Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Cette fonction insère un fait à la fin d'une liste de faits.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.
f	Un pointeur vers le fait à insérer.

Returns

Faits∗ Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Definition at line **42** of file **fait.c**.

5.7.3.7 LibererFaits()

```
void LibererFaits (
            Faits * tete )
```

Libère la mémoire allouée à une liste de faits.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.

Cette fonction parcourt une liste de faits et libère la mémoire allouée à chaque fait et à son identifiant.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.

Definition at line **117** of file **fait.c**.

5.8 fait.h

Go to the documentation of this file.

```
00001
00011 #ifndef FAIT_H
00012 #define FAIT_H
00013
00014 #include <stdio.h>
00015 #include <stdlib.h>
00016 #include <string.h>
00017
00023 typedef struct Faits
00024 {
00025     int nombre_faits;
00026     char *identifiant;
00027     struct Faits *suiv;
00028 } Faits;
00029
00035 Faits *CreerFaits();
00036
00044 Faits *InsererFaits(Faits *tete, Faits *f);
00045
00051 Faits *AjouterFaits();
00052
00060 Faits *AjouterFaitsSuite(Faits *liste, const char *nouveauFait);
00061
00067 void AfficherFaits(Faits *tete);
00068
00074 void LibererFaits(Faits *tete);
00075
00083 int FaitPresent(Faits *liste_faits, const char *but);
00084
00085 #endif // FAIT_H
```

5.9 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/lecture.h File Reference

Fichier d'en-tête pour la lecture de fichier et le stockage de données.

```
#include <stdio.h>
#include <stdint.h>
```

Functions

- FILE * **ouvrir_fichier** (const char *chemin_fichier, char *option)

  Ouvre un fichier.

5.9.1 Detailed Description

Fichier d'en-tête pour la lecture de fichier et le stockage de données.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-15

Copyright

Copyright (c) 2024

Definition in file **lecture.h**.

5.9.2 Function Documentation

5.9.2.1 ouvrir_fichier()

```
FILE * ouvrir_fichier (
            const char * chemin_fichier,
            char * option )
```

Ouvre un fichier.

Parameters

chemin_fichier	Le chemin vers le fichier à ouvrir.
option	Les options d'ouverture du fichier (par exemple, "r" pour lire, "w" pour écrire).

Returns

Un pointeur vers le fichier ouvert.

Definition at line **22** of file **lecture.c**.

5.10 lecture.h

Go to the documentation of this file.

```
00001
00011 #ifndef LECTURE_H
00012 #define LECTURE_H
00013
00014 #include <stdio.h>
00015 #include <stdint.h>
00016
00023 FILE *ouvrir_fichier(const char *chemin_fichier, char *option);
00024
00025 #endif // LECTURE_H
```

5.11 C:/Users/Alexandre/OneDrive/Bureau/projetalgo↩ S6/include/memoire.h File Reference

Fichier d'en-tête pour la gestion de la mémoire.

```
#include <stdio.h>
#include <stdint.h>
```

Functions

- void * **allouer_malloc** (int taille)

  Permet de faire un malloc tout en prenant en compte les erreurs de pointeurs NULL.

5.11.1 Detailed Description

Fichier d'en-tête pour la gestion de la mémoire.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-15

Copyright

Copyright (c) 2024

Definition in file **memoire.h**.

5.11.2 Function Documentation

5.11.2.1 allouer_malloc()

```
void * allouer_malloc (
            int n )
```

Permet de faire un malloc tout en prenant en compte les erreurs de pointeurs NULL.

Parameters

taille	

Returns

void∗

Parameters

n	

Returns

void∗

Definition at line **20** of file **memoire.c**.

5.12 memoire.h

Go to the documentation of this file.

```
00001
00011 #ifndef MEMOIRE_H
00012 #define MEMOIRE_H
00013
00014 #include <stdio.h>
00015 #include <stdint.h>
00016
00023 void *allouer_malloc(int taille);
00024
00025 #endif // MEMOIRE_H
```

5.13 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/include/menu.h File Reference

Fichier d'en-tête pour la fonction de menu.

Functions

- int **menu** ()

  Affiche un menu à l'utilisateur et récupère son choix.

5.13.1 Detailed Description

Fichier d'en-tête pour la fonction de menu.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Definition in file **menu.h**.

5.13.2 Function Documentation

5.13.2.1 menu()

```
int menu ( )
```

Affiche un menu à l'utilisateur et récupère son choix.

Cette fonction affiche un menu avec différentes options à l'utilisateur, puis lit son choix à partir de l'entrée standard.

Returns

int Retourne le choix de l'utilisateur sous forme d'un entier.

Definition at line **22** of file **menu.c**.

5.14 menu.h

Go to the documentation of this file.

```
00001
00011 #ifndef MENU_H
00012 #define MENU_H
00013
00021 int menu();
00022
00023 #endif // MENU_H
```

5.15 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/README.md File Reference

5.16 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/caractere.c File Reference

Fichier **caractere.c** (p. 26) contenant l'ensemble des fonctions pour les opérations concernant les opérations sur les caractères.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "caractere.h"
```

Functions

- int **CaractereDansListe** (**Faits** ∗tete, char caractere)
  Vérifie si un caractère est présent dans une liste de faits.
- **Faits** ∗ **AjouterCaractereSiAbsent** (**Faits** ∗liste, char caractere)
  Ajoute un caractère à une liste de faits s'il n'est pas déjà présent.

5.16.1 Detailed Description

Fichier **caractere.c** (p. 26) contenant l'ensemble des fonctions pour les opérations concernant les opérations sur les caractères.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Copyright (c) 2024

Definition in file **caractere.c**.

5.16.2 Function Documentation

5.16.2.1 AjouterCaractereSiAbsent()

```
Faits * AjouterCaractereSiAbsent (
            Faits * liste,
           char caractere )
```

Ajoute un caractère à une liste de faits s'il n'est pas déjà présent.

Parameters

liste	Pointeur vers le premier élément de la liste de faits.
caractere	Le caractère à ajouter à la liste de faits.

Returns

Faits∗ Retourne un pointeur vers le premier élément de la liste de faits

Definition at line **48** of file **caractere.c**.

5.16.2.2 CaractereDansListe()

```
int CaractereDansListe (
              Faits * tete,
             char caractere )
```

Vérifie si un caractère est présent dans une liste de faits.

Parameters

tete	Pointeur vers le premier élément de la liste de faits.
caractere	Le caractère à rechercher dans la liste de faits.

Returns

int Retourne 1 si le caractère est trouvé dans la liste, sinon retourne

Definition at line **24** of file **caractere.c**.

5.17 caractere.c

Go to the documentation of this file.

```
00001
00012 #include <stdio.h>
00013 #include <stdlib.h>
00014 #include <string.h>
00015 #include "caractere.h"
00016
00024 int CaractereDansListe(Faits *tete, char caractere)
00025 {
00026     Faits *courant = tete;
00027     while (courant != NULL)
00028     {
00029         char *identifiant = courant->identifiant;
00030         while (*identifiant != '\0')
00031         {
00032             if (*identifiant == caractere)
00033                 return 1;
00034             identifiant++;
00035         }
00036         courant = courant->suiv;
00037     }
00038     return 0;
00039 }
00040
00048 Faits *AjouterCaractereSiAbsent(Faits *liste, char caractere)
```

```
00049 {
00050     if (!CaractereDansListe(liste, caractere))
00051     {
00052         char identifiant[100];
00053         identifiant[0] = caractere;
00054         identifiant[1] = '\0';
00055         liste = AjouterFaitsSuite(liste, identifiant);
00056     }
00057     return liste;
00058 }
```

5.18 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/chainage_arriere.c File Reference

Fichier **chainage_arriere.c** (p. 28) contenant l'ensemble des procédures pour le chainage arrière.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "chainage_arriere.h"
#include "lecture.h"
```

Functions

- int **ChainageArriere** (const char *file, **Faits** *liste_faits, char *but)
  Effectue un chaînage arrière sur une liste de faits pour atteindre un but.

5.18.1 Detailed Description

Fichier **chainage_arriere.c** (p. 28) contenant l'ensemble des procédures pour le chainage arrière.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Copyright (c) 2024

Definition in file **chainage_arriere.c**.

5.18.2 Function Documentation

5.18.2.1 ChainageArriere()

```
int ChainageArriere (
            const char * file,
             Faits * liste_faits,
            char * but )
```

Effectue un chaînage arrière sur une liste de faits pour atteindre un but.

Cette fonction lit un fichier de règles et utilise un algorithme de chaînage arrière pour déterminer si un but peut être atteint à partir d'une liste de faits initiale. Le but est atteint si toutes les prémisses d'une règle qui a le but comme conséquence sont présentes dans la liste de faits.

Parameters

file	Le nom du fichier contenant les règles à utiliser pour le chaînage arrière.
liste_faits	Un pointeur vers le premier élément de la liste de faits initiale.
but	La chaîne de caractères représentant le but à atteindre.

Returns

int Retourne 1 si le but est atteint, sinon retourne 0.

Definition at line **28** of file **chainage_arriere.c**.

5.19 chainage_arriere.c

Go to the documentation of this file.

```
00001
00012 #include <stdio.h>
00013 #include <stdlib.h>
00014 #include <string.h>
00015 #include "chainage_arriere.h"
00016 #include "lecture.h"
00017
00028 int ChainageArriere(const char *file, Faits *liste_faits, char *but)
00029 {
00030     FILE *fichier = ouvrir_fichier(file, "r");
00031
00032     char ligne[100];
00033     int but_atteint = 0;
00034     while (fgets(ligne, sizeof(ligne), fichier))
00035     {
00036         char ligne_copie[100];
00037         strcpy(ligne_copie, ligne);
00038         char *premisses = strtok(ligne_copie, "->\n");
00039         char *consequences = strtok(NULL, "->\n");
00040         if (premisses == NULL || consequences == NULL)
00041         {
00042             printf("Ligne invalide: %s\n", ligne);
00043             continue;
00044         }
00045         if (strstr(consequences, but) != NULL) // permet de comaparer 2 char *
00046         {
00047             int toutes_prouvees = 1;
00048             char *caractere = strtok(premisses, "; \n");
00049             while (caractere != NULL)
00050             {
00051                 if (!FaitPresent(liste_faits, caractere))
00052                 {
00053                     if (!ChainageArriere(file, liste_faits, caractere))
```

```
00054                     {
00055                         toutes_prouvees = 0;
00056                         break;
00057                     }
00058                 }
00059                 caractere = strtok(NULL, "; \n");
00060             }
00061             if (toutes_prouvees)
00062             {
00063                 but_atteint = 1;
00064                 break;
00065             }
00066         }
00067     }
00068
00069     fclose(fichier);
00070
00071     if (but_atteint)
00072     {
00073         printf("Le but \"%s\" est atteint !\n", but);
00074         return 1;
00075     }
00076     else
00077     {
00078         printf("Le but \"%s\" n'est pas atteint\n", but);
00079         return 0;
00080     }
00081 }
```

5.20 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/chainage_↩ avant.c File Reference

Fichier **chainage_avant.c** (p. 30) contenant l'ensemble des procédures pour le chainage avant.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "chainage_avant.h"
#include "lecture.h"
```

Functions

- **Faits** ∗ **ChainageAvant** (const char ∗file, **Faits** ∗liste_faits)

  Effectue un chaînage avant sur une liste de faits pour atteindre un but.

5.20.1 Detailed Description

Fichier **chainage_avant.c** (p. 30) contenant l'ensemble des procédures pour le chainage avant.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Copyright (c) 2024

Definition in file **chainage_avant.c**.

5.20.2 Function Documentation

5.20.2.1 ChainageAvant()

```
Faits * ChainageAvant (
            const char * file,
             Faits * liste_faits )
```

Effectue un chaînage avant sur une liste de faits pour atteindre un but.

Effectue un chaînage avant sur une liste de faits.

Cette fonction lit un fichier de règles et utilise un algorithme de chaînage avant pour déterminer si un but peut être atteint à partir d'une liste de faits initiale. Le but est atteint si toutes les conséquences d'une règle qui a le but comme prémisse sont présentes dans la liste de faits.

Parameters

file	Le nom du fichier contenant les règles à utiliser pour le chaînage avant.
liste_faits	Un pointeur vers le premier élément de la liste de faits initiale.

Returns

Faits* Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Definition at line **26** of file **chainage_avant.c**.

5.21 chainage_avant.c

Go to the documentation of this file.

```
00001
00011 #include <stdio.h>
00012 #include <stdlib.h>
00013 #include <string.h>
00014 #include "chainage_avant.h"
00015 #include "lecture.h"
00016
00026 Faits *ChainageAvant(const char *file, Faits *liste_faits)
00027 {
00028     FILE *fichier = ouvrir_fichier(file, "r");
00029
00030     char ligne[100];
00031     int ajout = 1;
00032
00033     char but[100];
00034     printf("Saisir le but à atteindre: ");
00035     scanf("%s", but);
00036
00037     while (ajout)
00038     {
00039         ajout = 0;
00040         fseek(fichier, 0, SEEK_SET);
00041         while (fgets(ligne, sizeof(ligne), fichier))
00042         {
00043             char ligne_copie[100];
00044             strcpy(ligne_copie, ligne);
00045             char *premisses = strtok(ligne_copie, "->\n");
00046             char *consequences = strtok(NULL, "->\n");
00047             if (premisses == NULL || consequences == NULL)
00048             {
00049                 printf("Ligne invalide: %s\n", ligne);
00050                 continue;
00051             }
```

```
00052             int toutes_presentes = 1;
00053             for (int i = 0; i < strlen(premisses); i++)
00054             {
00055                 char caractere = premisses[i];
00056                 if (!CaractereDansListe(liste_faits, caractere) && caractere != ' ')
00057                 {
00058                     toutes_presentes = 0;
00059                     break;
00060                 }
00061             }
00062             if (toutes_presentes)
00063             {
00064                 for (int i = 0; i < strlen(consequences); i++)
00065                 {
00066                     char caractere = consequences[i];
00067                     if (caractere == ';' || caractere == ' ')
00068                         continue;
00069                     if (!CaractereDansListe(liste_faits, caractere))
00070                     {
00071                         liste_faits = AjouterCaractereSiAbsent(liste_faits, caractere);
00072                         ajout = 1;
00073                     }
00074                 }
00075             }
00076         }
00077     }
00078     if (FaitPresent(liste_faits, but))
00079         printf("Le but \"%s\" est atteint !\n", but);
00080     else
00081         printf("Le but \"%s\" n'est pas atteint\n", but);
00082     fclose(fichier);
00083     return liste_faits;
00084 }
```

5.22 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/fait.c File Reference

Fichier **fait.c** (p. 32) contenant l'ensemble des procédures pour la gestion des faits.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "fait.h"
#include "memoire.h"
```

Functions

- **Faits** ∗ **CreerFaits** ()

  Crée un nouveau fait.
- **Faits** ∗ **InsererFaits** (**Faits** ∗tete, **Faits** ∗f)

  Insère un fait dans une liste de faits.
- **Faits** ∗ **AjouterFaits** ()

  Ajoute des faits à une liste de faits.
- **Faits** ∗ **AjouterFaitsSuite** (**Faits** ∗liste, const char ∗nouveauFait)

  Ajoute un fait à la suite d'une liste de faits.
- void **AfficherFaits** (**Faits** ∗tete)

  Affiche une liste de faits.
- void **LibererFaits** (**Faits** ∗tete)

  Libère la mémoire allouée à une liste de faits.
- int **FaitPresent** (**Faits** ∗liste_faits, const char ∗but)

  Vérifie si un fait est présent dans une liste de faits.

5.22.1 Detailed Description

Fichier **fait.c** (p. 32) contenant l'ensemble des procédures pour la gestion des faits.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Copyright (c) 2024

Definition in file **fait.c**.

5.22.2 Function Documentation

5.22.2.1 AfficherFaits()

```
void AfficherFaits (
            Faits * tete )
```

Affiche une liste de faits.

Cette fonction parcourt une liste de faits et affiche l'identifiant de chaque fait.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.

Definition at line **99** of file **fait.c**.

5.22.2.2 AjouterFaits()

```
Faits * AjouterFaits ( )
```

Ajoute des faits à une liste de faits.

Cette fonction demande à l'utilisateur combien de faits il souhaite ajouter, puis ajoute ces faits à la liste.

Returns

Faits* Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Definition at line **58** of file **fait.c**.

5.22.2.3 AjouterFaitsSuite()

```
Faits * AjouterFaitsSuite (
            Faits * liste,
           const char * nouveauFait )
```

Ajoute un fait à la suite d'une liste de faits.

Cette fonction crée un nouveau fait avec l'identifiant spécifié et l'insère à la fin de la liste de faits donnée.

Parameters

liste	Un pointeur vers le premier élément de la liste de faits.
nouveauFait	Une chaîne de caractères représentant l'identifiant du nouveau fait à ajouter.

Returns

Faits* Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Definition at line **85** of file **fait.c**.

5.22.2.4 CreerFaits()

```
Faits * CreerFaits ( )
```

Crée un nouveau fait.

Cette fonction alloue de la mémoire pour un nouveau fait et initialise ses membres.

Returns

Faits* Retourne un pointeur vers le nouveau fait créé.

Definition at line **24** of file **fait.c**.

5.22.2.5 FaitPresent()

```
int FaitPresent (
            Faits * liste_faits,
           const char * but )
```

Vérifie si un fait est présent dans une liste de faits.

Parameters

liste_faits	Un pointeur vers le premier élément de la liste de faits.
but	Une chaîne de caractères représentant le fait à rechercher.

Returns

int Retourne 1 si le fait est trouvé dans la liste, sinon retourne 0.

Definition at line **129** of file **fait.c**.

5.22.2.6 InsererFaits()

```
Faits * InsererFaits (
            Faits * tete,
            Faits * f )
```

Insère un fait dans une liste de faits.

Cette fonction insère un fait à la fin d'une liste de faits.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.
f	Un pointeur vers le fait à insérer.

Returns

Faits∗ Retourne un pointeur vers le premier élément de la liste de faits mise à jour.

Definition at line **42** of file **fait.c**.

5.22.2.7 LibererFaits()

```
void LibererFaits (
            Faits * tete )
```

Libère la mémoire allouée à une liste de faits.

Cette fonction parcourt une liste de faits et libère la mémoire allouée à chaque fait et à son identifiant.

Parameters

tete	Un pointeur vers le premier élément de la liste de faits.

Definition at line **117** of file **fait.c**.

5.23 fait.c

Go to the documentation of this file.

```
00001
00011 #include <stdio.h>
00012 #include <stdlib.h>
00013 #include <string.h>
```

```
00014 #include "fait.h"
00015 #include "memoire.h"
00016
00024 Faits *CreerFaits()
00025 {
00026     Faits *nouveau = allouer_malloc(sizeof(Faits));
00027     nouveau->nombre_faits = 0;
00028     nouveau->identifiant = NULL;
00029     nouveau->suiv = NULL;
00030     return nouveau;
00031 }
00032
00042 Faits *InsererFaits(Faits *tete, Faits *f)
00043 {
00044     if (tete == NULL)
00045         return f;
00046     else
00047         tete->suiv = InsererFaits(tete->suiv, f);
00048     return tete;
00049 }
00050
00058 Faits *AjouterFaits()
00059 {
00060     Faits *liste = NULL;
00061     int nombre_faits;
00062     printf("Combien de faits voulez-vous ajouter ? ");
00063     scanf("%d", &nombre_faits);
00064     for (int i = 0; i < nombre_faits; i++)
00065     {
00066         char identifiant[100];
00067         printf("Saisir un fait: ");
00068         scanf("%s", identifiant);
00069         Faits *nouveau = CreerFaits();
00070         nouveau->identifiant = strdup(identifiant);
00071         liste = InsererFaits(liste, nouveau);
00072     }
00073     return liste;
00074 }
00075
00085 Faits *AjouterFaitsSuite(Faits *liste, const char *nouveauFait)
00086 {
00087     Faits *nouveau = CreerFaits();
00088     nouveau->identifiant = strdup(nouveauFait);
00089     return InsererFaits(liste, nouveau);
00090 }
00091
00099 void AfficherFaits(Faits *tete)
00100 {
00101     Faits *courant = tete;
00102     while (courant != NULL)
00103     {
00104         printf("%s -> ", courant->identifiant);
00105         courant = courant->suiv;
00106     }
00107     printf("NULL\n");
00108 }
00109
00117 void LibererFaits(Faits *tete)
00118 {
00119     Faits *courant = tete;
00120     while (courant != NULL)
00121     {
00122         Faits *suivant = courant->suiv;
00123         free(courant->identifiant);
00124         free(courant);
00125         courant = suivant;
00126     }
00127 }
00128
00129 int FaitPresent(Faits *liste_faits, const char *but)
00130 {
00131     Faits *courant = liste_faits;
00132     while (courant != NULL)
00133     {
00134         if (strcmp(courant->identifiant, but) == 0)
00135         {
00136             return 1;
00137         }
00138         courant = courant->suiv;
00139     }
00140     return 0;
00141 }
```

5.24 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/lecture.c File Reference

Fichier **lecture.c** (p. 37) contenant l'ensemble des fonctions pour la lecture de fichier et de données .kbs.

```
#include <stdio.h>
#include <stdlib.h>
#include "lecture.h"
#include <string.h>
```

Functions

- FILE ∗ **ouvrir_fichier** (const char ∗chemin_fichier, char ∗option)
  Ouvre un fichier.

5.24.1 Detailed Description

Fichier **lecture.c** (p. 37) contenant l'ensemble des fonctions pour la lecture de fichier et de données .kbs.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Definition in file **lecture.c**.

5.24.2 Function Documentation

5.24.2.1 ouvrir_fichier()

```
FILE * ouvrir_fichier (
            const char * chemin_fichier,
            char * option )
```

Ouvre un fichier.

Parameters

chemin_fichier	Le chemin vers le fichier à ouvrir.
option	Les options d'ouverture du fichier (par exemple, "r" pour lire, "w" pour écrire).

Returns

Un pointeur vers le fichier ouvert.

Definition at line **22** of file **lecture.c**.

5.25 lecture.c

Go to the documentation of this file.

```
00001
00011 #include <stdio.h>
00012 #include <stdlib.h>
00013 #include "lecture.h"
00014 #include <string.h>
00015
00022 FILE *ouvrir_fichier(const char *chemin_fichier, char *option)
00023 {
00024     FILE *mon_fichier = fopen(chemin_fichier, option);
00025
00026     if (mon_fichier == NULL)
00027     {
00028         fprintf(stderr, "Impossible d'ouvrir le fichier %s\n", chemin_fichier);
00029         exit(EXIT_FAILURE);
00030     }
00031     return mon_fichier;
00032 }
```

5.26 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/main.c File Reference

Fichier **main.c** (p. 38) contenant le code qui sera éxécuté par l'utilisateur.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "chainage_avant.h"
#include "memoire.h"
#include "chainage_arriere.h"
#include "menu.h"
```

Functions

- int **main** ()

  Point d'entrée principal du programme.

5.26.1 Detailed Description

Fichier **main.c** (p. 38) contenant le code qui sera éxécuté par l'utilisateur.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Definition in file **main.c**.

5.26.2 Function Documentation

5.26.2.1 main()

```
int main ( )
```

Point d'entrée principal du programme.

Cette fonction affiche un menu à l'utilisateur et effectue des opérations basées sur le choix de l'utilisateur. Les opérations incluent l'ajout de faits à une liste, l'affichage de la liste de faits, et l'exécution d'un chaînage avant ou arrière sur la liste de faits pour atteindre un but spécifié par l'utilisateur.

Returns

int Retourne 0 lorsque le programme se termine correctement.

Definition at line **26** of file **main.c**.

5.27 main.c

Go to the documentation of this file.

```
00001
00011 #include <stdio.h>
00012 #include <stdlib.h>
00013 #include <string.h>
00014 #include "chainage_avant.h"
00015 #include "memoire.h"
00016 #include "chainage_arriere.h"
00017 #include "menu.h"
00018
00026 int main()
00027 {
00028     Faits *liste = NULL;
00029     char *but = (char *)allouer_malloc(sizeof(char));
00030     int choix = menu();
00031     while (choix != 0)
00032     {
00033         switch (choix)
00034         {
00035         case 1:
00036             puts("Pour le chainage avant, vous disposez d'une grande base règles (+ de 1500 comprenant
      des caractères spéciaux). Alors n'hésitez pas à tester beaucoup de faits ! (Easter egg : essayez
      d'obtenir le +)\n");
00037             liste = AjouterFaits();
00038             printf("Faits insérés: ");
00039             AfficherFaits(liste);
00040             ChainageAvant("regles_av.kbs", liste);
00041             printf("Chainage avant: ");
00042             AfficherFaits(liste);
00043             break;
00044
00045         case 2:
00046             puts("Pour le chainage arriere, la base de règles disponible est celle du cahier des
      charges (i.e :b d e -> f)\n");
00047             liste = AjouterFaits();
00048             printf("Faits insérés: ");
00049             AfficherFaits(liste);
00050             printf("Saisir le but à atteindre: ");
00051             scanf("%s", but);
00052             ChainageArriere("regles_ar.kbs", liste, but);
00053             break;
00054
00055         case 3:
00056             puts("Nom de l'école (en MAJ) ||| ou mettez en faits les trois suivant : a,3,c / puis en
      but à atteindre : + ");
00057             puts("Pour l'école, dans Combien de faits voulez-vous ajouter ? Entrez 1, puis dans saisir
      fait : ISEN / but à atteindre : +\n");
00058             break;
00059         default:
00060             break;
00061         }
00062         choix = menu();
00063     }
00064     LibererFaits(liste);
00065     return 0;
00066 }
```

5.28 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/memoire.c File Reference

Fichier **memoire.c** (p. 40) contenant l'ensemble des fonctions pour les opérations concernant la mémoire.

```
#include "memoire.h"
#include <stdlib.h>
```

Functions

- void * **allouer_malloc** (int n)

  Permet de faire un malloc tout en prenant en compte les erreurs de pointeurs NULL.

5.28.1 Detailed Description

Fichier **memoire.c** (p. 40) contenant l'ensemble des fonctions pour les opérations concernant la mémoire.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Copyright (c) 2024

Definition in file **memoire.c**.

5.28.2 Function Documentation

5.28.2.1 allouer_malloc()

```
void * allouer_malloc (
            int n )
```

Permet de faire un malloc tout en prenant en compte les erreurs de pointeurs NULL.

Parameters

n	

Returns

void*

Definition at line **20** of file **memoire.c**.

5.29 memoire.c

Go to the documentation of this file.

```
00001
00011 #include "memoire.h"
00012 #include <stdlib.h>
00013
```

```
00020 void *allouer_malloc(int n)
00021 {
00022     void *pointeur;
00023
00024     pointeur = malloc(n);
00025
00026     if (pointeur == NULL)
00027     {
00028         exit(EXIT_FAILURE);
00029     }
00030
00031     return pointeur;
00032 }
```

5.30 C:/Users/Alexandre/OneDrive/Bureau/projetalgoS6/src/menu.c File Reference

Fichier **menu.c** (p. 42) contenant la procédure pour créer le menu.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Functions

- int **menu** ()
  Affiche un menu à l'utilisateur et récupère son choix.

5.30.1 Detailed Description

Fichier **menu.c** (p. 42) contenant la procédure pour créer le menu.

Author

Alexandre, Tom, Yanis, Charlotte

Version

0.1

Date

2024-03-17

Copyright

Copyright (c) 2024

Definition in file **menu.c**.

5.30.2 Function Documentation

5.30.2.1 menu()

```
int menu ( )
```

Affiche un menu à l'utilisateur et récupère son choix.

Cette fonction affiche un menu avec différentes options à l'utilisateur, puis lit son choix à partir de l'entrée standard.

Returns

int Retourne le choix de l'utilisateur sous forme d'un entier.

Definition at line **22** of file **menu.c**.

5.31 menu.c

Go to the documentation of this file.

```
00001
00011 #include <stdio.h>
00012 #include <stdlib.h>
00013 #include <string.h>
00014
00022 int menu()
00023 {
00024     int nb = -1;
00025     printf("\n|-------------------------------------------------------|\n");
00026     puts("Choisissez d'effectuer du chainage avant ou arriere :\n");
00027     printf("(0) Quitter\n");
00028     printf("(1) Chainage avant\n");
00029     printf("(2) Chainage arrière\n");
00030     printf("(3) Solution Easter Egg\n");
00031     printf("Votre choix: ");
00032     scanf("%d", &nb);
00033     printf("\n");
00034     return nb;
00035 }
```