

Prédiction Prix Logement Airbnb

Généré par Doxygen 1.8.17

1 Index des classes	1
1.1 Liste des classes	1
2 Index des fichiers	3
2.1 Liste des fichiers	3
3 Documentation des classes	5
3.1 Référence de la structure defHousing	5
3.1.1 Documentation des données membres	5
3.1.1.1 accommodates	5
3.1.1.2 bathrooms	5
3.1.1.3 bedrooms	5
3.1.1.4 beds	6
3.1.1.5 distance	6
3.1.1.6 max_nights	6
3.1.1.7 min_nights	6
3.1.1.8 number_of_reviews	6
3.1.1.9 price	6
3.2 Référence de la structure defregisterMae	6
3.2.1 Documentation des données membres	7
3.2.1.1 characteristic_num	7
3.2.1.2 k	7
3.2.1.3 mae	7
3.3 Référence de la structure Housing	7
3.3.1 Description détaillée	7
3.4 Référence de la structure RegisterMae	7
3.4.1 Description détaillée	7
4 Documentation des fichiers	9
4.1 Référence du fichier src/defines.h	9
4.2 Référence du fichier src/functions_algo_KNN.c	9
4.2.1 Description détaillée	10
4.2.2 Documentation des fonctions	10
4.2.2.1 calculateDistance()	10
4.2.2.2 calculateMAE()	11
4.2.2.3 evalPredictionModel()	11
4.2.2.4 housingXDataFilling()	12
4.2.2.5 impementationKnnAlgorithm()	12
4.2.2.6 inputInt()	12
4.2.2.7 inputNumCharac()	13
4.2.2.8 pricePrediction()	13
4.2.2.9 shuffleHousingsArray()	14
4.3 Référence du fichier src/functions_algo_KNN.h	14

4.3.1 Description détaillée	15
4.3.2 Documentation des fonctions	15
4.3.2.1 evalPredictionModel()	15
4.3.2.2 housingXDataFilling()	15
4.3.2.3 impementationKnnAlgorithm()	16
4.3.2.4 inputInt()	16
4.3.2.5 inputNumCharac()	16
4.4 Référence du fichier src/functions_sort.c	17
4.4.1 Description détaillée	17
4.4.2 Documentation des fonctions	18
4.4.2.1 floatComparator0()	18
4.4.2.2 floatComparator1()	18
4.4.2.3 floatComparator2()	19
4.4.2.4 floatComparator3()	19
4.4.2.5 floatComparator4()	20
4.4.2.6 floatComparator5()	20
4.4.2.7 floatComparator6()	21
4.4.2.8 maeComparator()	21
4.4.2.9 sortHousingArray()	22
4.5 Référence du fichier src/functions_sort.h	22
4.5.1 Description détaillée	23
4.5.2 Documentation des fonctions	23
4.5.2.1 floatComparator0()	23
4.5.2.2 floatComparator1()	24
4.5.2.3 floatComparator2()	24
4.5.2.4 floatComparator3()	25
4.5.2.5 floatComparator4()	25
4.5.2.6 floatComparator5()	26
4.5.2.7 floatComparator6()	26
4.5.2.8 maeComparator()	27
4.5.2.9 sortHousingArray()	27
4.6 Référence du fichier src/main.c	28
4.6.1 Description détaillée	28
4.7 Référence du fichier src/structures.h	28
4.7.1 Description détaillée	28

Chapitre 1

Index des classes

1.1 Liste des classes

Liste des classes, structures, unions et interfaces avec une brève description :

defHousing	5
defregisterMae	6
Housing		
Structure d'un logement	7
RegisterMae		
Structure d'un enregistrement de mae	7

Chapitre 2

Index des fichiers

2.1 Liste des fichiers

Liste de tous les fichiers documentés avec une brève description :

src/ defines.h	
Fichier de définitions	9
src/ functions_algo_KNN.c	
Fonctions principales de l'algorithme KNN	9
src/ functions_algo_KNN.h	
Fichier d'en tête des fonctions de fonctions_algo_KNN.c	14
src/ functions_fileToArray.h	??
src/ functions_sort.c	
Fonctions traitant des fichiers et des tableaux	17
src/ functions_sort.h	
Fichier d'en tête des fonctions de fonctions_sort.c	22
src/ main.c	
Main program	28
src/ structures.h	
Fichier de structures	28

Chapitre 3

Documentation des classes

3.1 Référence de la structure defHousing

Attributs publics

- float `accommodates`
- float `bedrooms`
- float `bathrooms`
- float `beds`
- float `price`
- float `min_nights`
- float `max_nights`
- float `number_of_reviews`
- float * `distance`

3.1.1 Documentation des données membres

3.1.1.1 `accommodates`

```
float defHousing::accommodates
```

Nombre de places

3.1.1.2 `bathrooms`

```
float defHousing::bathrooms
```

Nombre de salles de bains

3.1.1.3 `bedrooms`

```
float defHousing::bedrooms
```

Nombre de chambres

3.1.1.4 beds

```
float defHousing::beds
```

Nombre de lits

3.1.1.5 distance

```
float* defHousing::distance
```

Tableau des distances

3.1.1.6 max_nights

```
float defHousing::max_nights
```

Nombre de nuits maximum

3.1.1.7 min_nights

```
float defHousing::min_nights
```

Nombre de nuits minimum

3.1.1.8 number_of_reviews

```
float defHousing::number_of_reviews
```

Nombre de reviews

3.1.1.9 price

```
float defHousing::price
```

Prix

La documentation de cette structure a été générée à partir du fichier suivant :

— [src/structures.h](#)

3.2 Référence de la structure defregisterMae

Attributs publics

- float [mae](#)
- int [characteristic_num](#)
- int [k](#)

3.2.1 Documentation des données membres

3.2.1.1 characteristic_num

```
int defregisterMae::characteristic_num
```

Numero de la caractéristique ayant donnée cette MAE

3.2.1.2 k

```
int defregisterMae::k
```

Valeur de k ayant donnée cette MAE

3.2.1.3 mae

```
float defregisterMae::mae
```

Valeur de la MAE

La documentation de cette structure a été générée à partir du fichier suivant :

— src/[structures.h](#)

3.3 Référence de la structure Housing

Structure d'un logement.

```
#include <structures.h>
```

3.3.1 Description détaillée

Structure d'un logement.

La documentation de cette structure a été générée à partir du fichier suivant :

— src/[structures.h](#)

3.4 Référence de la structure RegisterMae

Structure d'un enregistrement de mae.

```
#include <structures.h>
```

3.4.1 Description détaillée

Structure d'un enregistrement de mae.

La documentation de cette structure a été générée à partir du fichier suivant :

— src/[structures.h](#)

Chapitre 4

Documentation des fichiers

4.1 Référence du fichier src/defines.h

Fichier de définitions.

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

4.2 Référence du fichier src/functions_algo_KNN.c

Fonctions principales de l'algorithme KNN.

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <time.h>
#include "structures.h"
#include "defines.h"
#include "functions_algo_KNN.h"
#include "functions_fileToArray.h"
#include "functions_sort.h"
```

Graphe des dépendances par inclusion de functions_algo_KNN.c:

Fonctions

- void [calculateDistance](#) ([Housing](#) *housingArray, int housingArraylength, [Housing](#) housingX, char *characteristics[], int number_characteristics)
Fonction calculant les distances entre tous les logements dont le prix est connu avec un logement dont on cherche à prédire le prix.
- [Housing](#) [housingXDataFilling](#) ()
Permet d'entrer les valeurs des caractéristiques d'un logement.
- void [shuffleHousingsArray](#) ([Housing](#) *housingArray, size_t housingArrayLength)
Fonction permettant de mélanger aléatoirement un tableau.
- float [pricePrediction](#) ([Housing](#) *housingArray, size_t housingArrayLength, int k)
Fonction calculant le prix prédit.
- float [calculateMAE](#) ([Housing](#) *housingArray, size_t housingArrayLength, float *pricePredictions)
Fonction calculant la MAE.
- int [inputInt](#) (int lowerLimit, int upperLimit)

- *Fonction d'entrée d'un entier dans des bornes.*
int `inputNumCharac` ()
- *Menu de choix de la caractéristique selon laquelle le prix va être prédit.*
void `impementationKnnAlgorithm` ()
- *Fonction principale permettant de prédire le prix d'un logement d'après ses caractéristique - Partie 1 du sujet.*
void `evalPredictionModel` ()
Fonction principale permettant de déterminer la caractéristique et le k permettant des prédictions plus précises - Partie 2(Bonus) du sujet.

4.2.1 Description détaillée

Fonctions principales de l'algorithme KNN.

Auteur

Lucas Velay

Date

10 décembre 2022

4.2.2 Documentation des fonctions

4.2.2.1 `calculateDistance()`

```
void calculateDistance (  
    Housing * housingArray,  
    int housingArraylength,  
    Housing housingX,  
    char * characteristics[],  
    int number_characteristics )
```

Fonction calculant les distances entre tous les logements dont le prix est connu avec un logement dont on cherche à prédire le prix.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<code>housingArray</code>	Tableau de logements d'on le prix est connu
<code>housingArraylength</code>	Taille du tableau de logements
<code>housingX</code>	Logement dont on cherche à déterminer le prix
<code>characteristics</code>	Tableau contenant le nom des caractéristiques d'un logement
<code>number_characteristics</code>	Nombre de caractéristiques

4.2.2.2 calculateMAE()

```
float calculateMAE (
    Housing * housingArray,
    size_t housingArrayLength,
    float * pricePredictions )
```

Fonction calculant la MAE.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>housingArray</i>	Tableau de logements tests
<i>housingArraylength</i>	Taille du tableau de logements
<i>pricePredictions</i>	Tableau de prix prédit

Renvoie

MAE

4.2.2.3 evalPredictionModel()

```
void evalPredictionModel ( )
```

Fonction principale permettant de déterminer la caractéristique et le k permettant des prédictions plus précises - Partie 2(Bonus) du sujet.

Auteur

Lucas Velay

Date

10 décembre 2022

4.2.2.4 housingXDataFilling()

```
Housing housingXDataFilling ( )
```

Permet d'entrer les valeurs des caractéristiques d'un logement.

Auteur

Lucas Velay

Date

10 décembre 2022

Renvoie

Le logement

4.2.2.5 impementationKnnAlgorithm()

```
void impementationKnnAlgorithm ( )
```

Fonction principale permettant de prédire le prix d'un logement d'après ses caractéristique - Partie 1 du sujet.

Auteur

Lucas Velay

Date

10 décembre 2022

4.2.2.6 inputInt()

```
inputInt (
    int lowerLimit,
    int upperLimit )
```

Fonction d'entrée d'un entier dans des bornes.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>lowerLimit</i>	Borne inférieur
<i>upperLimit</i>	Borne superieur

Renvoie

Valeur entrée

4.2.2.7 inputNumCharac()

```
int inputNumCharac ( )
```

Menu de choix de la caracteristique selon laquelle le prix va être prédit.

Auteur

Lucas Velay

Date

10 décembre 2022

Renvoie

Numero de la caracteristique

4.2.2.8 pricePrediction()

```
float pricePrediction (
    Housing * housingArray,
    size_t housingArrayLength,
    int k )
```

Fonction calculant le prix prédit.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>housingArray</i>	Tableau de logements d'on le prix est connu
<i>housingArraylength</i>	Taille du tableau de logements
<i>k</i> Généré par Doxygen	Paramètre k déterminant le nombre d'élément du tableau impliqués dans la moyenne déterminant le prix prédit

Renvoie

Prix prédit

4.2.2.9 shuffleHousingsArray()

```
void shuffleHousingsArray (
    Housing * housingArray,
    size_t housingArrayLength )
```

Fonction permettant de mélanger aléatoirement un tableau.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>housingArray</i>	Tableau de logements d'on le prix est connu
<i>housingArrayLength</i>	Taille du tableau de logements

4.3 Référence du fichier src/functions_algo_KNN.h

Fichier d'en tête des fonctions de fonctions_algo_KNN.c.

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- void **calculateDistance** (Housing housingArray[], int housingArraylength, Housing housingX, char *characteristics[], int number_characteristics)
- Housing housingXDataFilling ()
Permet d'entrer les valeurs des caractéristiques d'un logement.
- void **impementationKnnAlgorithm** ()
Fonction principale permettant de prédire le prix d'un logement d'après ses caractéristique - Partie 1 du sujet.
- void **evalPredictionModel** ()
Fonction principale permettant de déterminer la caractéristique et le k permettant des prédictions plus précises - Partie 2(Bonus) du sujet.
- int **inputInt** (int lowerLimit, int upperLimit)
Fonction d'entrée d'un entier dans des bornes.
- int **inputNumCharac** ()
Menu de choix de la caractéristique selon laquelle le prix va être prédit.
- int **inputK** (int housingArrayLength)

4.3.1 Description détaillée

Fichier d'en tête des fonctions de fonctions_algo_KNN.c.

Auteur

Lucas Velay

Date

10 décembre 2022

4.3.2 Documentation des fonctions

4.3.2.1 evalPredictionModel()

```
void evalPredictionModel ( )
```

Fonction principale permettant de déterminer la caractéristique et le k permettant des prédictions plus précises - Partie 2(Bonus) du sujet.

Auteur

Lucas Velay

Date

10 décembre 2022

4.3.2.2 housingXDataFilling()

```
Housing housingXDataFilling ( )
```

Permet d'entrer les valeurs des caractéristiques d'un logement.

Auteur

Lucas Velay

Date

10 décembre 2022

Renvoie

Le logement

4.3.2.3 `impementationKnnAlgorithm()`

```
void impementationKnnAlgorithm ( )
```

Fonction principale permettant de prédire le prix d'un logement d'après ses caractéristique - Partie 1 du sujet.

Auteur

Lucas Velay

Date

10 décembre 2022

4.3.2.4 `inputInt()`

```
int inputInt (
    int lowerLimit,
    int upperLimit )
```

Fonction d'entrée d'un entier dans des bornes.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>lowerLimit</i>	Borne inférieur
<i>upperLimit</i>	Borne superieur

Renvoie

Valeur entrée

4.3.2.5 `inputNumCharac()`

```
int inputNumCharac ( )
```

Menu de choix de la caracteristique selon laquelle le prix va être prédit.

Auteur

Lucas Velay

Date

10 décembre 2022

Renvoi

Numero de la caracteristique

4.4 Référence du fichier src/functions_sort.c

Fonctions traitant des fichiers et des tableaux.

```
#include <stdio.h>
#include <stdlib.h>
#include "structures.h"
#include "defines.h"
#include "functions_sort.h"
Graphe des dépendances par inclusion de functions_sort.c:
```

Fonctions

- int [floatComparator0](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 0 du tableau de distance.
- int [floatComparator1](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 1 du tableau de distance.
- int [floatComparator2](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 2 du tableau de distance.
- int [floatComparator3](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 3 du tableau de distance.
- int [floatComparator4](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 4 du tableau de distance.
- int [floatComparator5](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 5 du tableau de distance.
- int [floatComparator6](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 6 du tableau de distance.
- void [sortHousingArray](#) ([Housing](#) *housingArray, size_t housingArrayLength, int num_charac_sort)
Fonction de tri du tableau de logements utilisant un switch pour choisir le champ du tableau de distance à utiliser pour le tri.
- int [maeComparator](#) (const void *first, const void *second)
Fonction permettant de comparer le tableau de MAE.

4.4.1 Description détaillée

Fonctions traitant des fichiers et des tableaux.

Fonctions de tri.

Auteur

Lucas Velay

Date

10 décembre 2022

4.4.2 Documentation des fonctions

4.4.2.1 floatComparator0()

```
int floatComparator0 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 0 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.4.2.2 floatComparator1()

```
int floatComparator1 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 1 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.4.2.3 floatComparator2()

```
int floatComparator2 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 2 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.4.2.4 floatComparator3()

```
int floatComparator3 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 3 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.4.2.5 floatComparator4()

```
int floatComparator4 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 4 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.4.2.6 floatComparator5()

```
int floatComparator5 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 5 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.4.2.7 floatComparator6()

```
int floatComparator6 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 6 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.4.2.8 maeComparator()

```
int maeComparator (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le tableau de MAE.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.4.2.9 sortHousingArray()

```
void sortHousingArray (
    Housing * housingArray,
    size_t housingArrayLength,
    int num_charac_sort )
```

Fonction de tri du tableau de logements utilisant un switch pour choisir le champ du tableau de distance à utiliser pour le tri.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>housingArray</i>	Tableau de logements à trier
<i>housingArrayLength</i>	Taille du tableau de logements à trier
<i>num_charac_sort</i>	Le numero de la caractéristique selon laquelle le tableau est trié

Renvoie

l'état de la comparaison

4.5 Référence du fichier src/functions_sort.h

Fichier d'en tête des fonctions de fonctions_sort.c.

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Fonctions

- int [floatComparator0](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 0 du tableau de distance.
- int [floatComparator1](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 1 du tableau de distance.
- int [floatComparator2](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 2 du tableau de distance.
- int [floatComparator3](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 3 du tableau de distance.
- int [floatComparator4](#) (const void *first, const void *second)
Fonction permettant de comparer le champ 4 du tableau de distance.

- int `floatComparator5` (const void *first, const void *second)
Fonction permettant de comparer le champ 5 du tableau de distance.
- int `floatComparator6` (const void *first, const void *second)
Fonction permettant de comparer le champ 6 du tableau de distance.
- void `sortHousingArray` (`Housing` *housingArray, size_t housingArrayLength, int num_charac_sort)
Fonction de tri du tableau de logements utilisant un switch pour choisir le champ du tableau de distance à utiliser pour le tri.
- int `maeComparator` (const void *first, const void *second)
Fonction permettant de comparer le tableau de MAE.

4.5.1 Description détaillée

Fichier d'en tête des fonctions de fonctions_sort.c.

Auteur

Lucas Velay

Date

10 décembre 2022

4.5.2 Documentation des fonctions

4.5.2.1 floatComparator0()

```
int floatComparator0 (  
    const void * first,  
    const void * second )
```

Fonction permettant de comparer le champ 0 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.5.2.2 floatComparator1()

```
int floatComparator1 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 1 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.5.2.3 floatComparator2()

```
int floatComparator2 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 2 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.5.2.4 floatComparator3()

```
int floatComparator3 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 3 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.5.2.5 floatComparator4()

```
int floatComparator4 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 4 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.5.2.6 floatComparator5()

```
int floatComparator5 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 5 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.5.2.7 floatComparator6()

```
int floatComparator6 (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le champ 6 du tableau de distance.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.5.2.8 maeComparator()

```
int maeComparator (
    const void * first,
    const void * second )
```

Fonction permettant de comparer le tableau de MAE.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>first</i>	Première valeur à comparer
<i>second</i>	Seconde valeur à comparer

Renvoie

l'état de la comparaison

4.5.2.9 sortHousingArray()

```
void sortHousingArray (
    Housing * housingArray,
    size_t housingArrayLength,
    int num_charac_sort )
```

Fonction de tri du tableau de logements utilisant un switch pour choisir le champ du tableau de distance à utiliser pour le tri.

Auteur

Lucas Velay

Date

10 décembre 2022

Paramètres

<i>housingArray</i>	Tableau de logements à trier
<i>housingArrayLength</i>	Taille du tableau de logements à trier
<i>num_charac_sort</i>	Le numero de la caractéristique selon laquelle le tableau est trié

Renvoie

l'état de la comparaison

4.6 Référence du fichier src/main.c

Main program.

```
#include <stdio.h>
#include <stdlib.h>
#include "structures.h"
#include "defines.h"
#include "functions_algo_KNN.h"
#include "functions_sort.h"
```

Graphe des dépendances par inclusion de main.c:

Fonctions

— int **main** (int argc, char const *argv[])

4.6.1 Description détaillée

Main program.

Auteur

Lucas Velay

Date

10 décembre 2022

4.7 Référence du fichier src/structures.h

Fichier de structures.

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

Classes

— struct [defHousing](#)
— struct [defregisterMae](#)

Définitions de type

— typedef struct [defHousing](#) **Housing**
— typedef struct [defregisterMae](#) **RegisterMae**

4.7.1 Description détaillée

Fichier de structures.

Auteur

Lucas Velay

Date

10 décembre 2022

Index

- accommodates
 - defHousing, [5](#)
- bathrooms
 - defHousing, [5](#)
- bedrooms
 - defHousing, [5](#)
- beds
 - defHousing, [5](#)
- calculateDistance
 - functions_algo_KNN.c, [10](#)
- calculateMAE
 - functions_algo_KNN.c, [11](#)
- characteristic_num
 - defregisterMae, [7](#)
- defHousing, [5](#)
 - accommodates, [5](#)
 - bathrooms, [5](#)
 - bedrooms, [5](#)
 - beds, [5](#)
 - distance, [6](#)
 - max_nights, [6](#)
 - min_nights, [6](#)
 - number_of_reviews, [6](#)
 - price, [6](#)
- defregisterMae, [6](#)
 - characteristic_num, [7](#)
 - k, [7](#)
 - mae, [7](#)
- distance
 - defHousing, [6](#)
- evalPredictionModel
 - functions_algo_KNN.c, [11](#)
 - functions_algo_KNN.h, [15](#)
- floatComparator0
 - functions_sort.c, [18](#)
 - functions_sort.h, [23](#)
- floatComparator1
 - functions_sort.c, [18](#)
 - functions_sort.h, [23](#)
- floatComparator2
 - functions_sort.c, [19](#)
 - functions_sort.h, [24](#)
- floatComparator3
 - functions_sort.c, [19](#)
 - functions_sort.h, [24](#)
- floatComparator4
 - functions_sort.c, [20](#)
 - functions_sort.h, [25](#)
- floatComparator5
 - functions_sort.c, [20](#)
 - functions_sort.h, [25](#)
- floatComparator6
 - functions_sort.c, [21](#)
 - functions_sort.h, [26](#)
- functions_algo_KNN.c
 - calculateDistance, [10](#)
 - calculateMAE, [11](#)
 - evalPredictionModel, [11](#)
 - housingXDataFilling, [11](#)
 - impementationKnnAlgorithm, [12](#)
 - inputInt, [12](#)
 - inputNumCharac, [13](#)
 - pricePrediction, [13](#)
 - shuffleHousingsArray, [14](#)
- functions_algo_KNN.h
 - evalPredictionModel, [15](#)
 - housingXDataFilling, [15](#)
 - impementationKnnAlgorithm, [15](#)
 - inputInt, [16](#)
 - inputNumCharac, [16](#)
- functions_sort.c
 - floatComparator0, [18](#)
 - floatComparator1, [18](#)
 - floatComparator2, [19](#)
 - floatComparator3, [19](#)
 - floatComparator4, [20](#)
 - floatComparator5, [20](#)
 - floatComparator6, [21](#)
 - maeComparator, [21](#)
 - sortHousingArray, [22](#)
- functions_sort.h
 - floatComparator0, [23](#)
 - floatComparator1, [23](#)
 - floatComparator2, [24](#)
 - floatComparator3, [24](#)
 - floatComparator4, [25](#)
 - floatComparator5, [25](#)
 - floatComparator6, [26](#)
 - maeComparator, [26](#)
 - sortHousingArray, [27](#)
- Housing, [7](#)
- housingXDataFilling
 - functions_algo_KNN.c, [11](#)
 - functions_algo_KNN.h, [15](#)

- impementationKnnAlgorithm
 - functions_algo_KNN.c, [12](#)
 - functions_algo_KNN.h, [15](#)
- inputInt
 - functions_algo_KNN.c, [12](#)
 - functions_algo_KNN.h, [16](#)
- inputNumCharac
 - functions_algo_KNN.c, [13](#)
 - functions_algo_KNN.h, [16](#)
- k
 - defregisterMae, [7](#)
- mae
 - defregisterMae, [7](#)
- maeComparator
 - functions_sort.c, [21](#)
 - functions_sort.h, [26](#)
- max_nights
 - defHousing, [6](#)
- min_nights
 - defHousing, [6](#)
- number_of_reviews
 - defHousing, [6](#)
- price
 - defHousing, [6](#)
- pricePrediction
 - functions_algo_KNN.c, [13](#)
- RegisterMae, [7](#)
- shuffleHousingsArray
 - functions_algo_KNN.c, [14](#)
- sortHousingArray
 - functions_sort.c, [22](#)
 - functions_sort.h, [27](#)
- src/defines.h, [9](#)
- src/functions_algo_KNN.c, [9](#)
- src/functions_algo_KNN.h, [14](#)
- src/functions_sort.c, [17](#)
- src/functions_sort.h, [22](#)
- src/main.c, [28](#)
- src/structures.h, [28](#)