

Voyageur du commerce

Généré par Doxygen 1.8.6

Mardi 15 Avril 2014 21 :19 :16

Table des matières

1	Page principale	1
2	Index des structures de données	3
2.1	Structures de données	3
3	Index des fichiers	5
3.1	Liste des fichiers	5
4	Documentation des structures de données	7
4.1	Référence de la structure file	7
4.1.1	Description détaillée	7
4.1.2	Documentation des champs	7
4.1.2.1	dernier	7
4.1.2.2	plein	7
4.1.2.3	premier	7
4.1.2.4	tableau	7
4.1.2.5	taille	7
4.2	Référence de la structure Matrice	8
4.2.1	Description détaillée	8
4.2.2	Documentation des champs	8
4.2.2.1	dimension	8
4.2.2.2	ref	8
4.2.2.3	tab	8
4.3	Référence de la structure Point	8
4.3.1	Description détaillée	8
4.3.2	Documentation des champs	8
4.3.2.1	visited	8
4.3.2.2	x	8
4.3.2.3	y	9
5	Documentation des fichiers	11
5.1	Référence du fichier Files.c	11
5.1.1	Documentation des définitions de type	11

5.1.1.1	file	11
5.1.1.2	objet	11
5.1.2	Documentation des fonctions	11
5.1.2.1	compteFile	12
5.1.2.2	creerFile	12
5.1.2.3	defiler	12
5.1.2.4	enfiler	12
5.1.2.5	fileVide	12
5.1.2.6	valeur	12
5.2	Référence du fichier Files.h	12
5.2.1	Documentation des définitions de type	12
5.2.1.1	file	12
5.2.1.2	objet	12
5.2.2	Documentation des fonctions	13
5.2.2.1	creerFile	13
5.2.2.2	defiler	13
5.2.2.3	enfiler	13
5.2.2.4	fileVide	13
5.2.2.5	valeur	13
5.3	Référence du fichier main.c	13
5.3.1	Description détaillée	13
5.3.2	Documentation des fonctions	13
5.3.2.1	erreurArguments	13
5.3.2.2	main	14
5.4	Référence du fichier matrice.c	15
5.4.1	Documentation des fonctions	15
5.4.1.1	afficherMatrice	15
5.4.1.2	cloneMatrice	15
5.4.1.3	creerMatriceDesPoints	15
5.4.1.4	creerMatriceVide	16
5.4.1.5	detruireMatrice	16
5.4.1.6	findMin	16
5.4.1.7	getDimensionMatrice	16
5.4.1.8	getDistanceIndice	16
5.4.1.9	getDistancePoint	16
5.4.1.10	getIndicePoint	16
5.4.1.11	getPointIndice	16
5.4.1.12	getTableauPointsMatrice	16
5.4.1.13	lowerBound	16
5.4.1.14	markAsInfinite	16

5.4.1.15	setDistanceIndice	16
5.4.1.16	setPointIndice	17
5.5	Référence du fichier matrice.h	17
5.5.1	Documentation des définitions de type	17
5.5.1.1	matrice	17
5.5.2	Documentation des fonctions	17
5.5.2.1	afficherMatrice	17
5.5.2.2	cloneMatrice	17
5.5.2.3	creerMatriceDesPoints	17
5.5.2.4	creerMatriceTSP	18
5.5.2.5	creerMatriceVide	18
5.5.2.6	detruireMatrice	18
5.5.2.7	getDimensionMatrice	18
5.5.2.8	getDistanceIndice	18
5.5.2.9	getDistancePoint	18
5.5.2.10	getIndicePoint	18
5.5.2.11	getPointIndice	18
5.5.2.12	getTableauPointsMatrice	18
5.5.2.13	lowerBound	18
5.5.2.14	markAsInfinite	18
5.5.2.15	setDistanceIndice	18
5.5.2.16	setPointIndice	19
5.6	Référence du fichier point.c	19
5.6.1	Documentation des fonctions	19
5.6.1.1	afficherListeDesPoints	19
5.6.1.2	afficherPoint	19
5.6.1.3	clone	19
5.6.1.4	creerPoint	19
5.6.1.5	detruirePoint	20
5.6.1.6	distanceEntreDeuxPoints	20
5.6.1.7	distanceManhattan	20
5.6.1.8	equals	20
5.6.1.9	getX	20
5.6.1.10	getY	20
5.6.1.11	isVisited	20
5.6.1.12	markNoVisited	20
5.6.1.13	markVisited	20
5.6.1.14	setX	20
5.6.1.15	setY	20
5.7	Référence du fichier point.h	20

5.7.1	Documentation des définitions de type	21
5.7.1.1	point	21
5.7.2	Documentation des fonctions	21
5.7.2.1	afficherListeDesPoints	21
5.7.2.2	afficherPoint	21
5.7.2.3	clone	21
5.7.2.4	creerPoint	21
5.7.2.5	destruirePoint	21
5.7.2.6	distanceEntreDeuxPoints	21
5.7.2.7	distanceManhattan	22
5.7.2.8	equals	22
5.7.2.9	getX	22
5.7.2.10	getY	22
5.7.2.11	isVisited	22
5.7.2.12	markNoVisited	22
5.7.2.13	markVisited	22
5.7.2.14	setX	22
5.7.2.15	setY	22
5.8	Référence du fichier projetAlgo.c	22
5.8.1	Documentation des fonctions	23
5.8.1.1	branchBound	23
5.8.1.2	bruteForce	23
5.8.1.3	bruteForceRough	23
5.8.1.4	copyList	23
5.8.1.5	copyListIndice	24
5.8.1.6	deleteFromList	24
5.8.1.7	nearestNeighbour	24
5.8.1.8	overallDistance	24
5.8.1.9	overallDistanceVerbose	24
5.8.1.10	PointLePlusProche	25
5.8.1.11	prim	26
5.8.1.12	swap	26
5.9	Référence du fichier projetAlgo.h	26
5.9.1	Documentation des fonctions	27
5.9.1.1	branchBound	27
5.9.1.2	bruteForce	27
5.9.1.3	copyList	27
5.9.1.4	copyListIndice	27
5.9.1.5	nearestNeighbour	28
5.9.1.6	overallDistance	28

5.9.1.7	overallDistanceVerbose	28
5.9.1.8	PointLePlusProche	28
5.9.1.9	prim	28
5.10	Référence du fichier test_BF.c	29
5.10.1	Documentation des fonctions	29
5.10.1.1	main	29
5.11	Référence du fichier test_NN.c	29
5.11.1	Documentation des fonctions	30
5.11.1.1	main	30
5.12	Référence du fichier test_Prim.c	30
5.12.1	Documentation des fonctions	30
5.12.1.1	main	30
5.13	Référence du fichier tspIOtourO.c	30
5.13.1	Documentation des macros	31
5.13.1.1	_GNU_SOURCE	31
5.13.2	Documentation des fonctions	31
5.13.2.1	creerMatriceTSP	31
5.13.2.2	creerTOUR	31
5.13.2.3	creerTSPMatrice	31
5.14	Référence du fichier tspIOtourO.h	31
5.14.1	Documentation des fonctions	31
5.14.1.1	creerMatriceTSP	31
5.14.1.2	creerTOUR	31
5.14.1.3	creerTSPMatrice	31
Index		32

Chapitre 1

Page principale

Cette application a pour vocation de calculer des solutions au problème du voyageur de commerce

Chapitre 2

Index des structures de données

2.1 Structures de données

Liste des structures de données avec une brève description :

file	7
Matrice	8
Point	8

Chapitre 3

Index des fichiers

3.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

Files.c	11
Files.h	12
main.c	
Point (p. 8) d'entrée de l'application	13
matrice.c	15
matrice.h	17
point.c	19
point.h	20
projetAlgo.c	22
projetAlgo.h	26
test_BF.c	29
test_NN.c	29
test_Prim.c	30
tsplOtourO.c	30
tsplOtourO.h	31

Chapitre 4

Documentation des structures de données

4.1 Référence de la structure file

Champs de données

- int **taille**
- int **premier**
- int **dernier**
- bool **plein**
- **objet * tableau**

4.1.1 Description détaillée

Définition à la ligne 14 du fichier Files.c.

4.1.2 Documentation des champs

4.1.2.1 int dernier

Définition à la ligne 18 du fichier Files.c.

4.1.2.2 bool plein

Définition à la ligne 19 du fichier Files.c.

4.1.2.3 int premier

Définition à la ligne 17 du fichier Files.c.

4.1.2.4 objet* tableau

Définition à la ligne 20 du fichier Files.c.

4.1.2.5 int taille

Définition à la ligne 16 du fichier Files.c.

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

- **Files.c**

4.2 Référence de la structure Matrice

Champs de données

- int **dimension**
- **point** * **ref**
- float ** **tab**

4.2.1 Description détaillée

Définition à la ligne 11 du fichier matrice.c.

4.2.2 Documentation des champs

4.2.2.1 int dimension

Définition à la ligne 13 du fichier matrice.c.

4.2.2.2 point* ref

Définition à la ligne 14 du fichier matrice.c.

4.2.2.3 float** tab

Définition à la ligne 15 du fichier matrice.c.

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

- **matrice.c**

4.3 Référence de la structure Point

Champs de données

- signed int **x**
- signed int **y**
- bool **visited**

4.3.1 Description détaillée

Définition à la ligne 10 du fichier point.c.

4.3.2 Documentation des champs

4.3.2.1 bool visited

Définition à la ligne 14 du fichier point.c.

4.3.2.2 signed int x

Définition à la ligne 12 du fichier point.c.

4.3.2.3 signed int y

Définition à la ligne 13 du fichier point.c.

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

— **point.c**

Chapitre 5

Documentation des fichiers

5.1 Référence du fichier Files.c

```
#include <stdlib.h>
#include <stdbool.h>
#include <stdio.h>
#include <string.h>
#include "Files.h"
```

Structures de données

- struct **file**

Définitions de type

- typedef int **objet**
- typedef struct **file** * **file**

Fonctions

- void **creerFile** (**file** F, int taille)
- **objet valeur** (**file** F)
- bool **fileVide** (**file** F)
- bool **enfiler** (**file** F, **objet** x)
- void **defiler** (**file** F)
- int **compteFile** (**file** F)

5.1.1 Documentation des définitions de type

5.1.1.1 typedef struct file* file

Définition à la ligne 22 du fichier Files.c.

5.1.1.2 typedef int objet

Définition à la ligne 12 du fichier Files.c.

5.1.2 Documentation des fonctions

5.1.2.1 `int compteFile (file F)`

Définition à la ligne 62 du fichier Files.c.

5.1.2.2 `void creerFile (file F, int taille)`

Définition à la ligne 24 du fichier Files.c.

5.1.2.3 `void defiler (file F)`

Définition à la ligne 55 du fichier Files.c.

5.1.2.4 `bool enfiler (file F, objet x)`

Définition à la ligne 42 du fichier Files.c.

5.1.2.5 `bool fileVide (file F)`

Définition à la ligne 37 du fichier Files.c.

5.1.2.6 `objet valeur (file F)`

Définition à la ligne 32 du fichier Files.c.

5.2 Référence du fichier Files.h

```
#include <stdbool.h>
```

Définitions de type

- `typedef int objet`
- `typedef struct file * file`

Fonctions

- `void creerFile (file F, int taille)`
- `objet valeur (file F)`
- `bool fileVide (file F)`
- `bool enfiler (file F, objet x)`
- `void defiler (file F)`

5.2.1 Documentation des définitions de type

5.2.1.1 `typedef struct file* file`

Définition à la ligne 9 du fichier Files.h.

5.2.1.2 `typedef int objet`

Définition à la ligne 6 du fichier Files.h.

5.2.2 Documentation des fonctions

5.2.2.1 void creerFile (file *F*, int *taille*)

Définition à la ligne 24 du fichier Files.c.

5.2.2.2 void defiler (file *F*)

Définition à la ligne 55 du fichier Files.c.

5.2.2.3 bool enfiler (file *F*, objet *x*)

Définition à la ligne 42 du fichier Files.c.

5.2.2.4 bool fileVide (file *F*)

Définition à la ligne 37 du fichier Files.c.

5.2.2.5 objet valeur (file *F*)

Définition à la ligne 32 du fichier Files.c.

5.3 Référence du fichier main.c

Point (p. 8) d'entrée de l'application.

```
#include <stdio.h>
#include <stdlib.h>
#include "point.h"
#include "projetAlgo.h"
#include "matrice.h"
#include "tspIOtourO.h"
```

Fonctions

- void **erreurArguments** ()
- int **main** (int argc, char **argv)

Fonction principale permettant à l'utilisateur de choisir une solution de parcours depuis un fichier tsp.

5.3.1 Description détaillée

Point (p. 8) d'entrée de l'application.

Définition dans le fichier **main.c**.

5.3.2 Documentation des fonctions

5.3.2.1 void erreurArguments ()

Définition à la ligne 143 du fichier main.c.

5.3.2.2 `int main (int argc, char ** argv)`

Fonction principale permettant à l'utilisateur de choisir une solution de parcours depuis un fichier tsp.

Paramètres

<i>argc</i>	contient le nombre d'argument passé en parametre
<i>argv</i>	contient les arguments de l'appel sous forme de chaine de caractere

Renvoie

: EXIT_FAILURE si il y a un probleme, EXIT_SUCESS si tout ce passe bien

Définition à la ligne 27 du fichier main.c.

5.4 Référence du fichier matrice.c

```
#include "point.h"
#include <stdio.h>
#include <stdlib.h>
#include <assert.h>
#include <string.h>
#include "matrice.h"
```

Structures de données

— struct **Matrice**

Fonctions

- **matrice creerMatriceVide** (int dimension)
- **matrice creerMatriceDesPoints** (point liste[], int dimension)
- void **detruireMatrice** (matrice m)
- **matrice cloneMatrice** (matrice m)
- int **getIndexPoint** (matrice m, point p)
- point **getPointIndice** (matrice m, int indice)
- float **getDistanceIndice** (matrice m, int ref1, int ref2)
- float **getDistancePoint** (matrice m, point p1, point p2)
- int **getDimensionMatrice** (matrice m)
- void **setDistanceIndice** (matrice m, int ref1, int ref2, float distance)
- void **setPointIndice** (matrice m, int i, point p)
- point * **getTableauPointsMatrice** (matrice m)
- void **afficherMatrice** (matrice m)
- int **findMin** (float *list, int len, int k)
- int **lowerBound** (matrice m)
- void **markAsInfinite** (matrice m, int i, int j)

5.4.1 Documentation des fonctions

5.4.1.1 void afficherMatrice (matrice m)

Définition à la ligne 125 du fichier matrice.c.

5.4.1.2 matrice cloneMatrice (matrice m)

Définition à la ligne 65 du fichier matrice.c.

5.4.1.3 matrice creerMatriceDesPoints (point liste[], int dimension)

Définition à la ligne 39 du fichier matrice.c.

5.4.1.4 matrice creerMatriceVide (int *dimension*)

Définition à la ligne 23 du fichier matrice.c.

5.4.1.5 void detruireMatrice (matrice *m*)

Définition à la ligne 56 du fichier matrice.c.

5.4.1.6 int findMin (float * *list*, int *len*, int *k*)

Définition à la ligne 146 du fichier matrice.c.

5.4.1.7 int getDimensionMatrice (matrice *m*)

Définition à la ligne 100 du fichier matrice.c.

5.4.1.8 float getDistanceIndice (matrice *m*, int *ref1*, int *ref2*)

Définition à la ligne 88 du fichier matrice.c.

5.4.1.9 float getDistancePoint (matrice *m*, point *p1*, point *p2*)

Définition à la ligne 94 du fichier matrice.c.

5.4.1.10 int getIndicePoint (matrice *m*, point *p*)

Définition à la ligne 73 du fichier matrice.c.

5.4.1.11 point getPointIndice (matrice *m*, int *indice*)

Définition à la ligne 82 du fichier matrice.c.

5.4.1.12 point* getTableauPointsMatrice (matrice *m*)

Définition à la ligne 117 du fichier matrice.c.

5.4.1.13 int lowerBound (matrice *m*)

Définition à la ligne 160 du fichier matrice.c.

5.4.1.14 void markAsInfinite (matrice *m*, int *i*, int *j*)

Définition à la ligne 192 du fichier matrice.c.

5.4.1.15 void setDistanceIndice (matrice *m*, int *ref1*, int *ref2*, float *distance*)

Définition à la ligne 105 du fichier matrice.c.

5.4.1.16 `void setPointIndice (matrice m, int i, point p)`

Définition à la ligne 112 du fichier `matrice.c`.

5.5 Référence du fichier `matrice.h`

```
#include "point.h"
#include <stdio.h>
```

Définitions de type

— `typedef struct Matrice * matrice`

Fonctions

- `matrice creerMatriceDesPoints (point liste[], int dimension)`
- `matrice creerMatriceVide (int dimension)`
- `void detruireMatrice (matrice m)`
- `matrice cloneMatrice (matrice m)`
- `int getIndexPoint (matrice m, point p)`
- `point getPointIndice (matrice m, int indice)`
- `float getDistanceIndice (matrice m, int ref1, int ref2)`
- `float getDistancePoint (matrice m, point p1, point p2)`
- `point * getTableauPointsMatrice (matrice m)`
- `void setDistanceIndice (matrice m, int ref1, int ref2, float distance)`
- `void setPointIndice (matrice m, int i, point p)`
- `int getDimensionMatrice (matrice m)`
- `void afficherMatrice (matrice m)`
- `matrice creerMatriceTSP (char *fnom)`
- `int lowerBound (matrice m)`
- `void markAsInfinite (matrice m, int i, int j)`

5.5.1 Documentation des définitions de type

5.5.1.1 `typedef struct Matrice* matrice`

Définition à la ligne 7 du fichier `matrice.h`.

5.5.2 Documentation des fonctions

5.5.2.1 `void afficherMatrice (matrice m)`

Définition à la ligne 125 du fichier `matrice.c`.

5.5.2.2 `matrice cloneMatrice (matrice m)`

Définition à la ligne 65 du fichier `matrice.c`.

5.5.2.3 `matrice creerMatriceDesPoints (point liste[], int dimension)`

Définition à la ligne 39 du fichier `matrice.c`.

5.5.2.4 matrice creerMatriceTSP (char * *fnom*)

Définition à la ligne 11 du fichier `tsplOtourO.c`.

5.5.2.5 matrice creerMatriceVide (int *dimension*)

Définition à la ligne 23 du fichier `matrice.c`.

5.5.2.6 void detruireMatrice (matrice *m*)

Définition à la ligne 56 du fichier `matrice.c`.

5.5.2.7 int getDimensionMatrice (matrice *m*)

Définition à la ligne 100 du fichier `matrice.c`.

5.5.2.8 float getDistanceIndice (matrice *m*, int *ref1*, int *ref2*)

Définition à la ligne 88 du fichier `matrice.c`.

5.5.2.9 float getDistancePoint (matrice *m*, point *p1*, point *p2*)

Définition à la ligne 94 du fichier `matrice.c`.

5.5.2.10 int getIndicePoint (matrice *m*, point *p*)

Définition à la ligne 73 du fichier `matrice.c`.

5.5.2.11 point getPointIndice (matrice *m*, int *indice*)

Définition à la ligne 82 du fichier `matrice.c`.

5.5.2.12 point* getTableauPointsMatrice (matrice *m*)

Définition à la ligne 117 du fichier `matrice.c`.

5.5.2.13 int lowerBound (matrice *m*)

Définition à la ligne 160 du fichier `matrice.c`.

5.5.2.14 void markAsInfinite (matrice *m*, int *i*, int *j*)

Définition à la ligne 192 du fichier `matrice.c`.

5.5.2.15 void setDistanceIndice (matrice *m*, int *ref1*, int *ref2*, float *distance*)

Définition à la ligne 105 du fichier `matrice.c`.

5.5.2.16 void setPointIndice (matrice *m*, int *i*, point *p*)

Définition à la ligne 112 du fichier matrice.c.

5.6 Référence du fichier point.c

```
#include <stdlib.h>
#include <assert.h>
#include <math.h>
#include <stdbool.h>
#include <stdio.h>
#include "point.h"
```

Structures de données

— struct **Point**

Fonctions

- point **creerPoint** (signed int *x*, signed int *y*)
- void **detruirePoint** (point *p*)
- bool **equals** (point *p1*, point *p2*)
- point **clone** (point *p*)
- signed int **getX** (point *p*)
- signed int **getY** (point *p*)
- bool **isVisited** (point *p*)
- void **setX** (point *p*, signed int *x*)
- void **setY** (point *p*, signed int *y*)
- void **markVisited** (point *p*)
- void **markNoVisited** (point *p*)
- void **afficherPoint** (point *p*)
- void **afficherListeDesPoints** (point **p*, int *len*)
- float **distanceManhattan** (point *p1*, point *p2*)
- float **distanceEntreDeuxPoints** (point *p1*, point *p2*)

5.6.1 Documentation des fonctions

5.6.1.1 void afficherListeDesPoints (point * *p*, int *len*)

Définition à la ligne 97 du fichier point.c.

5.6.1.2 void afficherPoint (point *p*)

Définition à la ligne 90 du fichier point.c.

5.6.1.3 point clone (point *p*)

Définition à la ligne 45 du fichier point.c.

5.6.1.4 point creerPoint (signed int *x*, signed int *y*)

Définition à la ligne 22 du fichier point.c.

5.6.1.5 void detruirePoint (point *p*)

Définition à la ligne 33 du fichier point.c.

5.6.1.6 float distanceEntreDeuxPoints (point *p1*, point *p2*)

Définition à la ligne 110 du fichier point.c.

5.6.1.7 float distanceManhattan (point *p1*, point *p2*)

Définition à la ligne 104 du fichier point.c.

5.6.1.8 bool equals (point *p1*, point *p2*)

Définition à la ligne 40 du fichier point.c.

5.6.1.9 signed int getX (point *p*)

Définition à la ligne 52 du fichier point.c.

5.6.1.10 signed int getY (point *p*)

Définition à la ligne 57 du fichier point.c.

5.6.1.11 bool isVisited (point *p*)

Définition à la ligne 62 du fichier point.c.

5.6.1.12 void markNoVisited (point *p*)

Définition à la ligne 85 du fichier point.c.

5.6.1.13 void markVisited (point *p*)

Définition à la ligne 79 du fichier point.c.

5.6.1.14 void setX (point *p*, signed int *x*)

Définition à la ligne 67 du fichier point.c.

5.6.1.15 void setY (point *p*, signed int *y*)

Définition à la ligne 73 du fichier point.c.

5.7 Référence du fichier point.h

```
#include <stdbool.h>
```

Définitions de type

— typedef struct **Point** * **point**

Fonctions

— **point creerPoint** (signed int *x*, signed int *y*)
— void **detruirePoint** (**point** *p*)
— bool **equals** (**point** *p1*, **point** *p2*)
— **point clone** (**point** *p*)
— signed int **getX** (**point** *p*)
— signed int **getY** (**point** *p*)
— bool **isVisited** (**point** *p*)
— void **setX** (**point** *p*, signed int *x*)
— void **setY** (**point** *p*, signed int *y*)
— void **markVisited** (**point** *p*)
— void **markNoVisited** (**point** *p*)
— void **afficherPoint** (**point** *p*)
— void **afficherListeDesPoints** (**point** **p*, int *len*)
— float **distanceEntreDeuxPoints** (**point** *p1*, **point** *p2*)
— float **distanceManhattan** (**point** *p1*, **point** *p2*)

5.7.1 Documentation des définitions de type

5.7.1.1 typedef struct **Point*** **point**

Définition à la ligne 6 du fichier point.h.

5.7.2 Documentation des fonctions

5.7.2.1 void **afficherListeDesPoints** (**point** * *p*, int *len*)

Définition à la ligne 97 du fichier point.c.

5.7.2.2 void **afficherPoint** (**point** *p*)

Définition à la ligne 90 du fichier point.c.

5.7.2.3 **point clone** (**point** *p*)

Définition à la ligne 45 du fichier point.c.

5.7.2.4 **point creerPoint** (signed int *x*, signed int *y*)

Définition à la ligne 22 du fichier point.c.

5.7.2.5 void **detruirePoint** (**point** *p*)

Définition à la ligne 33 du fichier point.c.

5.7.2.6 float **distanceEntreDeuxPoints** (**point** *p1*, **point** *p2*)

Définition à la ligne 110 du fichier point.c.

5.7.2.7 float distanceManhattan (point *p1*, point *p2*)

Définition à la ligne 104 du fichier point.c.

5.7.2.8 bool equals (point *p1*, point *p2*)

Définition à la ligne 40 du fichier point.c.

5.7.2.9 signed int getX (point *p*)

Définition à la ligne 52 du fichier point.c.

5.7.2.10 signed int getY (point *p*)

Définition à la ligne 57 du fichier point.c.

5.7.2.11 bool isVisited (point *p*)

Définition à la ligne 62 du fichier point.c.

5.7.2.12 void markNoVisited (point *p*)

Définition à la ligne 85 du fichier point.c.

5.7.2.13 void markVisited (point *p*)

Définition à la ligne 79 du fichier point.c.

5.7.2.14 void setX (point *p*, signed int *x*)

Définition à la ligne 67 du fichier point.c.

5.7.2.15 void setY (point *p*, signed int *y*)

Définition à la ligne 73 du fichier point.c.

5.8 Référence du fichier projetAlgo.c

```
#include "point.h"
#include <stdlib.h>
#include "matrice.h"
```

Fonctions

- int **PointLePlusProche** (int indicePointActuel, **matrice** m)
*Cherche dans la matrice quel est le point non visité le plus proche du point indiqué par l'indice, cette fonction utilise la fonction **getDistanceIndice()** (p. 16) permettant de connaître la distance entre deux points.*
- **point** * **nearestNeighbour** (**matrice** mIn)

Parcour de la matrice, crée un ordre de parcours permettant d'effectuer une distance moindre, recherche de point le plus proche.

- int **overallDistance** (matrice m, point *points)
Fonction qui calcul la distance totale entre tous les points d'une liste.
- int **overallDistanceVerbose** (matrice m, point *points)
- void **swap** (point *plist, int i, int j)
Fonction qui echange deux points passés en paramètres.
- void **copyList** (point *pIn, point *pOut, int len)
Fonction qui copie une liste de points.
- void **copyListIndice** (point *pIn, point *pOut, int start, int end)
Fonction qui copie une liste de points à partir d'un indice.
- void **deleteFromList** (point *in, int length, point p)
- void **bruteForceRough** (matrice m, point *pIn, int i, int n, int *min, point *pOut)
- point * **bruteForce** (matrice m)
Fonction qui permet le calcul de la distance du Voyageur avec l'algorithme Brute Force.
- point * **prim** (matrice mIn)
Fonction qui permet le calcul de la distance du Voyageur avec Prim.
- point * **branchBound** (matrice m)
Fonction qui permet le calcul de la distance du Voyageur par l'algorithme de Branch & Bound.

5.8.1 Documentation des fonctions

5.8.1.1 point* branchBound (matrice m)

Fonction qui permet le calcul de la distance du Voyageur par l'algorithme de Branch & Bound.

Paramètres

<i>m</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
----------	--

Renvoie

: la distance générale parcourue

Définition à la ligne 328 du fichier projetAlgo.c.

5.8.1.2 point* bruteForce (matrice m)

Fonction qui permet le calcul de la distance du Voyageur avec l'algorithme Brute Force.

Paramètres

<i>m</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
----------	--

Renvoie

: la distance générale parcourue

Définition à la ligne 241 du fichier projetAlgo.c.

5.8.1.3 void bruteForceRough (matrice m, point * pIn, int i, int n, int * min, point * pOut)

Définition à la ligne 208 du fichier projetAlgo.c.

5.8.1.4 void copyList (point * pIn, point * pOut, int len)

Fonction qui copie une liste de points.

Paramètres

<i>pIn</i>	tableau 1 de points
<i>pOut</i>	tableau 2 de points
<i>len</i>	taille

Définition à la ligne 168 du fichier projetAlgo.c.

5.8.1.5 void copyListIndice (point * *pIn*, point * *pOut*, int *start*, int *end*)

Fonction qui copie une liste de points à partir d'un indice.

Paramètres

<i>pIn</i>	tableau 1 de points
<i>pOut</i>	tableau 2 de points
<i>start</i>	indice de début
<i>end</i>	indice de fin

Définition à la ligne 183 du fichier projetAlgo.c.

5.8.1.6 void deleteFromList (point * *in*, int *length*, point *p*)

Définition à la ligne 192 du fichier projetAlgo.c.

5.8.1.7 point* nearestNeighbour (matrice *mIn*)

Parcour de la matrice, crée un ordre de parcours permettant d'effectuer une distance moindre, recherche de point le plus proche.

Paramètres

<i>m</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
----------	--

Renvoie

ordreDePassage[] tableau de sortie qui contiendra le nouveau parcours

Définition à la ligne 67 du fichier projetAlgo.c.

5.8.1.8 int overallDistance (matrice *m*, point * *points*)

Fonction qui calcul la distance totale entre tous les points d'une liste.

Paramètres

<i>m</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
<i>points</i>	tableau contenant les points

Renvoie

: la distance totale

Définition à la ligne 111 du fichier projetAlgo.c.

5.8.1.9 int overallDistanceVerbose (matrice *m*, point * *points*)

Définition à la ligne 123 du fichier projetAlgo.c.

5.8.1.10 int PointLePlusProche (int *indicePointActuel*, matrice *m*)

Cherche dans la matrice quel est le point non visité le plus proche du point indiqué par l'indice, cette fonction utilise la fonction **getDistanceIndice()** (p. 16) permettant de connaître la distance entre deux points.

Paramètres

<i>indicePoint-Actuel</i>	correspond à l'indice du point actuel, permettant l'accès à ce point.
<i>m</i>	matrice contenant tous les points

Renvoie

: indice du point le plus proche du point actuel

Définition à la ligne 14 du fichier projetAlgo.c.

5.8.1.11 **point*** prim (**matrice** *mln*)

Fonction qui permet le calcul de la distance du Voyageur avec Prim.

Paramètres

<i>mln</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
------------	--

Renvoie

: TabVisite tableau qui contient les points ordonnés selon le parcours

Définition à la ligne 270 du fichier projetAlgo.c.

5.8.1.12 **void** swap (**point** * *plist*, **int** *i*, **int** *j*)

Fonction qui échange deux points passés en paramètres.

Paramètres

<i>plist</i>	liste contenant les points
<i>i</i>	premier point
<i>j</i>	deuxième point

Définition à la ligne 151 du fichier projetAlgo.c.

5.9 Référence du fichier projetAlgo.h

```
#include "matrice.h"
```

Fonctions

- **int** **PointLePlusProche** (**int** *indicePointActuel*, **matrice** *m*)
*Cherche dans la matrice quel est le point non visité le plus proche du point indiqué par l'indice, cette fonction utilise la fonction **getDistanceIndice()** (p. 16) permettant de connaître la distance entre deux points.*
- **point** * **nearestNeighbour** (**matrice** *m*)
Parcours de la matrice, crée un ordre de parcours permettant d'effectuer une distance moindre, recherche de point le plus proche.
- **int** **overallDistance** (**matrice** *m*, **point** **points*)
Fonction qui calcul la distance totale entre tous les points d'une liste.
- **int** **overallDistanceVerbose** (**matrice** *m*, **point** **points*)
- **point** * **bruteForce** (**matrice** *m*)
Fonction qui permet le calcul de la distance du Voyageur avec l'algorithme Brute Force.
- **void** **copyList** (**point** **pln*, **point** **pOut*, **int** *len*)
Fonction qui copie une liste de points.
- **void** **copyListIndice** (**point** **pln*, **point** **pOut*, **int** *start*, **int** *end*)
Fonction qui copie une liste de points à partir d'un indice.

- **point * branchBound (matrice m)**
Fonction qui permet le calcul de la distance du Voyageur par l'algorithme de Branch & Bound.
- **point * prim (matrice m)**
Fonction qui permet le calcul de la distance du Voyageur avec Prim.

5.9.1 Documentation des fonctions

5.9.1.1 point* branchBound (matrice m)

Fonction qui permet le calcul de la distance du Voyageur par l'algorithme de Branch & Bound.

Paramètres

<i>m</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
----------	--

Renvoie

: la distance générale parcourue

Définition à la ligne 328 du fichier projetAlgo.c.

5.9.1.2 point* bruteForce (matrice m)

Fonction qui permet le calcul de la distance du Voyageur avec l'algorithme Brute Force.

Paramètres

<i>m</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
----------	--

Renvoie

: la distance générale parcourue

Définition à la ligne 241 du fichier projetAlgo.c.

5.9.1.3 void copyList (point * pIn, point * pOut, int len)

Fonction qui copie une liste de points.

Paramètres

<i>pIn</i>	tableau 1 de points
<i>pOut</i>	tableau 2 de points
<i>len</i>	taille

Définition à la ligne 168 du fichier projetAlgo.c.

5.9.1.4 void copyListIndice (point * pIn, point * pOut, int start, int end)

Fonction qui copie une liste de points à partir d'un indice.

Paramètres

<i>pIn</i>	tableau 1 de points
<i>pOut</i>	tableau 2 de points

<i>start</i>	indice de début
<i>end</i>	indice de fin

Définition à la ligne 183 du fichier projetAlgo.c.

5.9.1.5 **point*** nearestNeighbour (**matrice** *mln*)

Parcour de la matrice, crée un ordre de parcours permettant d'effectuer une distance moindre, recherche de point le plus proche.

Paramètres

<i>m</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
----------	--

Renvoie

ordreDePassage[] tableau de sortie qui contiendra le nouveau parcours

Définition à la ligne 67 du fichier projetAlgo.c.

5.9.1.6 **int** overallDistance (**matrice** *m*, **point *** *points*)

Fonction qui calcul la distance totale entre tous les points d'une liste.

Paramètres

<i>m</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
<i>points</i>	tableau contenant les points

Renvoie

: la distance totale

Définition à la ligne 111 du fichier projetAlgo.c.

5.9.1.7 **int** overallDistanceVerbose (**matrice** *m*, **point *** *points*)

Définition à la ligne 123 du fichier projetAlgo.c.

5.9.1.8 **int** PointLePlusProche (**int** *indicePointActuel*, **matrice** *m*)

Cherche dans la matrice quel est le point non visité le plus proche du point indiqué par l'indice, cette fonction utilise la fonction **getDistanceIndice()** (p. 16) permettant de connaître la distance entre deux points.

Paramètres

<i>indicePoint-Actuel</i>	correspond à l'indice du point actuel, permettant l'accès à ce point.
<i>m</i>	matrice contenant tous les points

Renvoie

: indice du point le plus proche du point actuel

Définition à la ligne 14 du fichier projetAlgo.c.

5.9.1.9 **point*** prim (**matrice** *mln*)

Fonction qui permet le calcul de la distance du Voyageur avec Prim.

Paramètres

<i>mln</i>	matrice d'entrée qui permet d'obtenir la liste des points que l'on souhaite parcourir.
------------	--

Renvoie

: TabVisite tableau qui contient les points ordonnés selon le parcours

Définition à la ligne 270 du fichier projetAlgo.c.

5.10 Référence du fichier test_BF.c

```
#include <stdio.h>
#include <stdlib.h>
#include "point.h"
#include "projetAlgo.h"
#include "matrice.h"
#include "tspIOtourO.h"
```

Fonctions

— int **main** ()
Fonction principale permettant de tester la fonction Brute force.

5.10.1 Documentation des fonctions

5.10.1.1 int main ()

Fonction principale permettant de tester la fonction Brute force.

Renvoie

: EXIT_FAILURE si il y a un probleme, EXIT_SUCESS si tout ce passe bien

Définition à la ligne 13 du fichier test_BF.c.

5.11 Référence du fichier test_NN.c

```
#include <stdio.h>
#include <stdlib.h>
#include "point.h"
#include "projetAlgo.h"
#include "matrice.h"
#include "tspIOtourO.h"
```

Fonctions

— int **main** ()
Fonction principale permettant de tester la fonction Nearest Neighbour.

5.11.1 Documentation des fonctions

5.11.1.1 int main ()

Fonction principale permettant de tester la fonction Nearest Neighbour.

Renvoie

: EXIT_FAILURE si il y a un probleme, EXIT_SUCCESS si tout ce passe bien

Définition à la ligne 14 du fichier test_NN.c.

5.12 Référence du fichier test_Prim.c

```
#include <stdio.h>
#include <stdlib.h>
#include "point.h"
#include "projetAlgo.h"
#include "matrice.h"
#include "tspIOtourO.h"
```

Fonctions

— int **main** ()
Fonction principale permettant de tester la fonction Prim.

5.12.1 Documentation des fonctions

5.12.1.1 int main ()

Fonction principale permettant de tester la fonction Prim.

Renvoie

: EXIT_FAILURE si il y a un probleme, EXIT_SUCCESS si tout ce passe bien

Définition à la ligne 13 du fichier test_Prim.c.

5.13 Référence du fichier tspIOtourO.c

```
#include <stdio.h>
#include <assert.h>
#include <stdlib.h>
#include <string.h>
#include "point.h"
#include "matrice.h"
#include "tspIOtourO.h"
```

Macros

— #define **_GNU_SOURCE**

Fonctions

- **matrice creerMatriceTSP** (char *fileName)
- void **creerTSPMatrice** (char *fileName, **matrice** m)
- void **creerTOUR** (char *fileName, **matrice** m, **point** liste[])

5.13.1 Documentation des macros

5.13.1.1 #define _GNU_SOURCE

Définition à la ligne 1 du fichier tsplOtourO.c.

5.13.2 Documentation des fonctions

5.13.2.1 **matrice** creerMatriceTSP (char * *fileName*)

Définition à la ligne 11 du fichier tsplOtourO.c.

5.13.2.2 void creerTOUR (char * *fileName*, **matrice** *m*, **point** *liste*[])

Définition à la ligne 90 du fichier tsplOtourO.c.

5.13.2.3 void creerTSPMatrice (char * *fileName*, **matrice** *m*)

Définition à la ligne 61 du fichier tsplOtourO.c.

5.14 Référence du fichier tsplOtourO.h

```
#include <stdio.h>
#include "matrice.h"
#include "point.h"
```

Fonctions

- **matrice creerMatriceTSP** (char *fnom)
- void **creerTSPMatrice** (char *fnom, **matrice** m)
- void **creerTOUR** (char *fnom, **matrice** m, **point** liste[])

5.14.1 Documentation des fonctions

5.14.1.1 **matrice** creerMatriceTSP (char * *fnom*)

Définition à la ligne 11 du fichier tsplOtourO.c.

5.14.1.2 void creerTOUR (char * *fnom*, **matrice** *m*, **point** *liste*[])

Définition à la ligne 90 du fichier tsplOtourO.c.

5.14.1.3 void creerTSPMatrice (char * *fnom*, **matrice** *m*)

Définition à la ligne 61 du fichier tsplOtourO.c.

Index

- `_GNU_SOURCE`
 - `tsplOtourO.c`, 31
- `afficherListeDesPoints`
 - `point.c`, 19
 - `point.h`, 21
- `afficherMatrice`
 - `matrice.c`, 15
 - `matrice.h`, 17
- `afficherPoint`
 - `point.c`, 19
 - `point.h`, 21
- `branchBound`
 - `projetAlgo.c`, 23
 - `projetAlgo.h`, 27
- `bruteForce`
 - `projetAlgo.c`, 23
 - `projetAlgo.h`, 27
- `bruteForceRough`
 - `projetAlgo.c`, 23
- `clone`
 - `point.c`, 19
 - `point.h`, 21
- `cloneMatrice`
 - `matrice.c`, 15
 - `matrice.h`, 17
- `compteFile`
 - `Files.c`, 11
- `copyList`
 - `projetAlgo.c`, 23
 - `projetAlgo.h`, 27
- `copyListIndice`
 - `projetAlgo.c`, 24
 - `projetAlgo.h`, 27
- `creerFile`
 - `Files.c`, 12
 - `Files.h`, 13
- `creerMatriceDesPoints`
 - `matrice.c`, 15
 - `matrice.h`, 17
- `creerMatriceTSP`
 - `matrice.h`, 17
 - `tsplOtourO.c`, 31
 - `tsplOtourO.h`, 31
- `creerMatriceVide`
 - `matrice.c`, 15
 - `matrice.h`, 18
- `creerPoint`
 - `point.c`, 19
 - `point.h`, 21
- `creerTOUR`
 - `tsplOtourO.c`, 31
 - `tsplOtourO.h`, 31
- `creerTSPMatrice`
 - `tsplOtourO.c`, 31
 - `tsplOtourO.h`, 31
- `defiler`
 - `Files.c`, 12
 - `Files.h`, 13
- `deleteFromList`
 - `projetAlgo.c`, 24
- `dernier`
 - `file`, 7
- `detruireMatrice`
 - `matrice.c`, 16
 - `matrice.h`, 18
- `detruirePoint`
 - `point.c`, 19
 - `point.h`, 21
- `dimension`
 - `Matrice`, 8
- `distanceEntreDeuxPoints`
 - `point.c`, 20
 - `point.h`, 21
- `distanceManhattan`
 - `point.c`, 20
 - `point.h`, 21
- `enfiler`
 - `Files.c`, 12
 - `Files.h`, 13
- `equals`
 - `point.c`, 20
 - `point.h`, 22
- `erreurArguments`
 - `main.c`, 13
- `file`, 7
 - `dernier`, 7
 - `Files.c`, 11
 - `Files.h`, 12
 - `plein`, 7
 - `premier`, 7
 - `tableau`, 7
 - `taille`, 7
- `fileVide`
 - `Files.c`, 12

- Files.h, 13
- Files.c, 11
 - compteFile, 11
 - creerFile, 12
 - defiler, 12
 - enfiler, 12
 - file, 11
 - fileVide, 12
 - objet, 11
 - valeur, 12
- Files.h, 12
 - creerFile, 13
 - defiler, 13
 - enfiler, 13
 - file, 12
 - fileVide, 13
 - objet, 12
 - valeur, 13
- findMin
 - matrice.c, 16
- getDimensionMatrice
 - matrice.c, 16
 - matrice.h, 18
- getDistanceIndice
 - matrice.c, 16
 - matrice.h, 18
- getDistancePoint
 - matrice.c, 16
 - matrice.h, 18
- getIndicePoint
 - matrice.c, 16
 - matrice.h, 18
- getPointIndice
 - matrice.c, 16
 - matrice.h, 18
- getTableauPointsMatrice
 - matrice.c, 16
 - matrice.h, 18
- getX
 - point.c, 20
 - point.h, 22
- getY
 - point.c, 20
 - point.h, 22
- isVisited
 - point.c, 20
 - point.h, 22
- lowerBound
 - matrice.c, 16
 - matrice.h, 18
- main
 - main.c, 13
 - test_BF.c, 29
 - test_NN.c, 30
 - test_Prim.c, 30
- main.c, 13
 - erreurArguments, 13
 - main, 13
- markAsInfinite
 - matrice.c, 16
 - matrice.h, 18
- markNoVisited
 - point.c, 20
 - point.h, 22
- markVisited
 - point.c, 20
 - point.h, 22
- Matrice, 8
 - dimension, 8
 - ref, 8
 - tab, 8
- matrice
 - matrice.h, 17
- matrice.c, 15
 - afficherMatrice, 15
 - cloneMatrice, 15
 - creerMatriceDesPoints, 15
 - creerMatriceVide, 15
 - detruireMatrice, 16
 - findMin, 16
 - getDimensionMatrice, 16
 - getDistanceIndice, 16
 - getDistancePoint, 16
 - getIndicePoint, 16
 - getPointIndice, 16
 - getTableauPointsMatrice, 16
 - lowerBound, 16
 - markAsInfinite, 16
 - setDistanceIndice, 16
 - setPointIndice, 16
- matrice.h, 17
 - afficherMatrice, 17
 - cloneMatrice, 17
 - creerMatriceDesPoints, 17
 - creerMatriceTSP, 17
 - creerMatriceVide, 18
 - detruireMatrice, 18
 - getDimensionMatrice, 18
 - getDistanceIndice, 18
 - getDistancePoint, 18
 - getIndicePoint, 18
 - getPointIndice, 18
 - getTableauPointsMatrice, 18
 - lowerBound, 18
 - markAsInfinite, 18
 - matrice, 17
 - setDistanceIndice, 18
 - setPointIndice, 18
- nearestNeighbour
 - projetAlgo.c, 24
 - projetAlgo.h, 28
- objet

- Files.c, 11
- Files.h, 12
- overallDistance
 - projetAlgo.c, 24
 - projetAlgo.h, 28
- overallDistanceVerbose
 - projetAlgo.c, 24
 - projetAlgo.h, 28
- plein
 - file, 7
- Point, 8
 - visited, 8
 - x, 8
 - y, 8
- point
 - point.h, 21
- point.c, 19
 - afficherListeDesPoints, 19
 - afficherPoint, 19
 - clone, 19
 - creerPoint, 19
 - detruirePoint, 19
 - distanceEntreDeuxPoints, 20
 - distanceManhattan, 20
 - equals, 20
 - getX, 20
 - getY, 20
 - isVisited, 20
 - markNoVisited, 20
 - markVisited, 20
 - setX, 20
 - setY, 20
- point.h, 20
 - afficherListeDesPoints, 21
 - afficherPoint, 21
 - clone, 21
 - creerPoint, 21
 - detruirePoint, 21
 - distanceEntreDeuxPoints, 21
 - distanceManhattan, 21
 - equals, 22
 - getX, 22
 - getY, 22
 - isVisited, 22
 - markNoVisited, 22
 - markVisited, 22
 - point, 21
 - setX, 22
 - setY, 22
- PointLePlusProche
 - projetAlgo.c, 24
 - projetAlgo.h, 28
- premier
 - file, 7
- prim
 - projetAlgo.c, 26
 - projetAlgo.h, 28
- projetAlgo.c, 22
 - branchBound, 23
 - bruteForce, 23
 - bruteForceRough, 23
 - copyList, 23
 - copyListIndice, 24
 - deleteFromList, 24
 - nearestNeighbour, 24
 - overallDistance, 24
 - overallDistanceVerbose, 24
 - PointLePlusProche, 24
 - prim, 26
 - swap, 26
- projetAlgo.h, 26
 - branchBound, 27
 - bruteForce, 27
 - copyList, 27
 - copyListIndice, 27
 - nearestNeighbour, 28
 - overallDistance, 28
 - overallDistanceVerbose, 28
 - PointLePlusProche, 28
 - prim, 28
- ref
 - Matrice, 8
- setDistanceIndice
 - matrice.c, 16
 - matrice.h, 18
- setPointIndice
 - matrice.c, 16
 - matrice.h, 18
- setX
 - point.c, 20
 - point.h, 22
- setY
 - point.c, 20
 - point.h, 22
- swap
 - projetAlgo.c, 26
- tab
 - Matrice, 8
- tableau
 - file, 7
- taille
 - file, 7
- test_BF.c, 29
 - main, 29
- test_NN.c, 29
 - main, 30
- test_Prim.c, 30
 - main, 30
- tsplOtourO.c, 30
 - _GNU_SOURCE, 31
 - creerMatriceTSP, 31
 - creerTOUR, 31
 - creerTSPMatrice, 31
- tsplOtourO.h, 31

creerMatriceTSP, 31
creerTOUR, 31
creerTSPMatrice, 31

valeur

Files.c, 12
Files.h, 13

visited

Point, 8

x

Point, 8

y

Point, 8