

Voyageur du commerce

Généré par Doxygen 1.8.6

Mardi 15 Avril 2014 11 :01 :14



# Table des matières

<b>1</b>	<b>Page principale</b>	<b>1</b>
<b>2</b>	<b>Index des structures de données</b>	<b>3</b>
2.1	Structures de données . . . . .	3
<b>3</b>	<b>Index des fichiers</b>	<b>5</b>
3.1	Liste des fichiers . . . . .	5
<b>4</b>	<b>Documentation des structures de données</b>	<b>7</b>
4.1	Référence de la structure Matrice . . . . .	7
4.1.1	Description détaillée . . . . .	7
4.1.2	Documentation des champs . . . . .	7
4.1.2.1	dimension . . . . .	7
4.1.2.2	ref . . . . .	7
4.1.2.3	tab . . . . .	7
4.2	Référence de la structure Point . . . . .	7
4.2.1	Description détaillée . . . . .	8
4.2.2	Documentation des champs . . . . .	8
4.2.2.1	visited . . . . .	8
4.2.2.2	x . . . . .	8
4.2.2.3	y . . . . .	8
<b>5</b>	<b>Documentation des fichiers</b>	<b>9</b>
5.1	Référence du fichier main.c . . . . .	9
5.1.1	Description détaillée . . . . .	9
5.1.2	Documentation des fonctions . . . . .	9
5.1.2.1	erreurArguments . . . . .	9
5.1.2.2	main . . . . .	9
5.2	Référence du fichier matrice.c . . . . .	10
5.2.1	Documentation des fonctions . . . . .	10
5.2.1.1	afficherMatrice . . . . .	10
5.2.1.2	cloneMatrice . . . . .	10
5.2.1.3	creerMatriceDesPoints . . . . .	10

5.2.1.4	creerMatriceVide . . . . .	11
5.2.1.5	destruireMatrice . . . . .	11
5.2.1.6	findMin . . . . .	11
5.2.1.7	getDimensionMatrice . . . . .	11
5.2.1.8	getDistanceIndice . . . . .	11
5.2.1.9	getDistancePoint . . . . .	11
5.2.1.10	getIndicePoint . . . . .	11
5.2.1.11	getPointIndice . . . . .	11
5.2.1.12	getTableauPointsMatrice . . . . .	11
5.2.1.13	lowerBound . . . . .	11
5.2.1.14	markAsInfinite . . . . .	11
5.2.1.15	setDistanceIndice . . . . .	11
5.2.1.16	setPointIndice . . . . .	12
5.3	Référence du fichier matrice.h . . . . .	12
5.3.1	Documentation des définitions de type . . . . .	12
5.3.1.1	matrice . . . . .	12
5.3.2	Documentation des fonctions . . . . .	12
5.3.2.1	afficherMatrice . . . . .	12
5.3.2.2	cloneMatrice . . . . .	12
5.3.2.3	creerMatriceDesPoints . . . . .	12
5.3.2.4	creerMatriceTSP . . . . .	13
5.3.2.5	creerMatriceVide . . . . .	13
5.3.2.6	destruireMatrice . . . . .	13
5.3.2.7	getDimensionMatrice . . . . .	13
5.3.2.8	getDistanceIndice . . . . .	13
5.3.2.9	getDistancePoint . . . . .	13
5.3.2.10	getIndicePoint . . . . .	13
5.3.2.11	getPointIndice . . . . .	13
5.3.2.12	getTableauPointsMatrice . . . . .	13
5.3.2.13	lowerBound . . . . .	13
5.3.2.14	markAsInfinite . . . . .	13
5.3.2.15	setDistanceIndice . . . . .	13
5.3.2.16	setPointIndice . . . . .	14
5.4	Référence du fichier point.c . . . . .	14
5.4.1	Documentation des fonctions . . . . .	14
5.4.1.1	afficherListeDesPoints . . . . .	14
5.4.1.2	afficherPoint . . . . .	14
5.4.1.3	clone . . . . .	14
5.4.1.4	creerPoint . . . . .	14
5.4.1.5	destruirePoint . . . . .	15

5.4.1.6	distanceEntreDeuxPoints . . . . .	15
5.4.1.7	distanceManhattan . . . . .	15
5.4.1.8	equals . . . . .	15
5.4.1.9	getX . . . . .	15
5.4.1.10	getY . . . . .	15
5.4.1.11	isVisited . . . . .	15
5.4.1.12	markNoVisited . . . . .	15
5.4.1.13	markVisited . . . . .	15
5.4.1.14	setX . . . . .	15
5.4.1.15	setY . . . . .	15
5.5	Référence du fichier point.h . . . . .	15
5.5.1	Documentation des définitions de type . . . . .	16
5.5.1.1	point . . . . .	16
5.5.2	Documentation des fonctions . . . . .	16
5.5.2.1	afficherListeDesPoints . . . . .	16
5.5.2.2	afficherPoint . . . . .	16
5.5.2.3	clone . . . . .	16
5.5.2.4	creerPoint . . . . .	16
5.5.2.5	destruirePoint . . . . .	16
5.5.2.6	distanceEntreDeuxPoints . . . . .	16
5.5.2.7	distanceManhattan . . . . .	17
5.5.2.8	equals . . . . .	17
5.5.2.9	getX . . . . .	17
5.5.2.10	getY . . . . .	17
5.5.2.11	isVisited . . . . .	17
5.5.2.12	markNoVisited . . . . .	17
5.5.2.13	markVisited . . . . .	17
5.5.2.14	setX . . . . .	17
5.5.2.15	setY . . . . .	17
5.6	Référence du fichier projetAlgo.c . . . . .	17
5.6.1	Documentation des fonctions . . . . .	18
5.6.1.1	branchBound . . . . .	18
5.6.1.2	bruteForce . . . . .	18
5.6.1.3	bruteForceRough . . . . .	18
5.6.1.4	copyList . . . . .	18
5.6.1.5	copyListIndice . . . . .	18
5.6.1.6	deleteFromList . . . . .	18
5.6.1.7	nearestNeighbour . . . . .	18
5.6.1.8	overallDistance . . . . .	19
5.6.1.9	overallDistanceVerbose . . . . .	19

5.6.1.10	PointLePlusProche . . . . .	19
5.6.1.11	prim . . . . .	19
5.6.1.12	swap . . . . .	19
5.7	Référence du fichier projetAlgo.h . . . . .	19
5.7.1	Documentation des fonctions . . . . .	20
5.7.1.1	branchBound . . . . .	20
5.7.1.2	bruteForce . . . . .	20
5.7.1.3	copyList . . . . .	20
5.7.1.4	copyListIndice . . . . .	20
5.7.1.5	nearestNeighbour . . . . .	20
5.7.1.6	overallDistance . . . . .	20
5.7.1.7	overallDistanceVerbose . . . . .	20
5.7.1.8	PointLePlusProche . . . . .	20
5.7.1.9	prim . . . . .	21
5.8	Référence du fichier test_BF.c . . . . .	21
5.8.1	Documentation des macros . . . . .	21
5.8.1.1	nombreDePoints . . . . .	21
5.8.2	Documentation des fonctions . . . . .	21
5.8.2.1	main . . . . .	21
5.9	Référence du fichier test_NN.c . . . . .	21
5.9.1	Documentation des fonctions . . . . .	22
5.9.1.1	main . . . . .	22
5.10	Référence du fichier test_Prim.c . . . . .	22
5.10.1	Documentation des macros . . . . .	22
5.10.1.1	nombreDePoints . . . . .	22
5.10.2	Documentation des fonctions . . . . .	22
5.10.2.1	main . . . . .	22
5.11	Référence du fichier tspIOtourO.c . . . . .	22
5.11.1	Documentation des macros . . . . .	23
5.11.1.1	_GNU_SOURCE . . . . .	23
5.11.2	Documentation des fonctions . . . . .	23
5.11.2.1	creerMatriceTSP . . . . .	23
5.11.2.2	creerTOUR . . . . .	23
5.11.2.3	creerTSPMatrice . . . . .	23
5.12	Référence du fichier tspIOtourO.h . . . . .	23
5.12.1	Documentation des fonctions . . . . .	23
5.12.1.1	creerMatriceTSP . . . . .	23
5.12.1.2	creerTOUR . . . . .	23
5.12.1.3	creerTSPMatrice . . . . .	24







## **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 :

<b>Matrice</b>	7
<b>Point</b>	7



## Chapitre 3

# Index des fichiers

### 3.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

<b>main.c</b>	
<b>Point</b> (p. 7) d'entrée de l'application . . . . .	9
<b>matrice.c</b> . . . . .	10
<b>matrice.h</b> . . . . .	12
<b>point.c</b> . . . . .	14
<b>point.h</b> . . . . .	15
<b>projetAlgo.c</b> . . . . .	17
<b>projetAlgo.h</b> . . . . .	19
<b>test_BF.c</b> . . . . .	21
<b>test_NN.c</b> . . . . .	21
<b>test_Prim.c</b> . . . . .	22
<b>tsplOtourO.c</b> . . . . .	22
<b>tsplOtourO.h</b> . . . . .	23



## Chapitre 4

# Documentation des structures de données

### 4.1 Référence de la structure Matrice

#### Champs de données

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

#### 4.1.1 Description détaillée

Définition à la ligne 11 du fichier matrice.c.

#### 4.1.2 Documentation des champs

##### 4.1.2.1 int dimension

Définition à la ligne 13 du fichier matrice.c.

##### 4.1.2.2 point\* ref

Définition à la ligne 14 du fichier matrice.c.

##### 4.1.2.3 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**

### 4.2 Référence de la structure Point

#### Champs de données

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

#### 4.2.1 Description détaillée

Définition à la ligne 10 du fichier point.c.

#### 4.2.2 Documentation des champs

##### 4.2.2.1 bool visited

Définition à la ligne 14 du fichier point.c.

##### 4.2.2.2 signed int x

Définition à la ligne 12 du fichier point.c.

##### 4.2.2.3 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 main.c

**Point** (p. 7) 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.*

#### 5.1.1 Description détaillée

**Point** (p. 7) d'entrée de l'application.

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

#### 5.1.2 Documentation des fonctions

##### 5.1.2.1 void erreurArguments ( )

Définition à la ligne 137 du fichier main.c.

##### 5.1.2.2 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.

## 5.2 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 **destruireMatrice** (matrice m)  
 — **matrice cloneMatrice** (matrice m)  
 — int **getIndicePoint** (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)

### 5.2.1 Documentation des fonctions

#### 5.2.1.1 void afficherMatrice ( matrice m )

Définition à la ligne 125 du fichier matrice.c.

#### 5.2.1.2 matrice cloneMatrice ( matrice m )

Définition à la ligne 65 du fichier matrice.c.

#### 5.2.1.3 matrice creerMatriceDesPoints ( point liste[], int dimension )

Définition à la ligne 39 du fichier matrice.c.

**5.2.1.4** `matrice creerMatriceVide ( int dimension )`

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

**5.2.1.5** `void detruireMatrice ( matrice m )`

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

**5.2.1.6** `int findMin ( float * list, int len, int k )`

Définition à la ligne 146 du fichier `matrice.c`.

**5.2.1.7** `int getDimensionMatrice ( matrice m )`

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

**5.2.1.8** `float getDistanceIndice ( matrice m, int ref1, int ref2 )`

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

**5.2.1.9** `float getDistancePoint ( matrice m, point p1, point p2 )`

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

**5.2.1.10** `int getIndicePoint ( matrice m, point p )`

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

**5.2.1.11** `point getPointIndice ( matrice m, int indice )`

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

**5.2.1.12** `point* getTableauPointsMatrice ( matrice m )`

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

**5.2.1.13** `int lowerBound ( matrice m )`

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

**5.2.1.14** `void markAsInfinite ( matrice m, int i, int j )`

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

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

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

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

Définition à la ligne 112 du fichier matrice.c.

### 5.3 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)

#### 5.3.1 Documentation des définitions de type

##### 5.3.1.1 typedef struct Matrice\* matrice

Définition à la ligne 7 du fichier matrice.h.

#### 5.3.2 Documentation des fonctions

##### 5.3.2.1 void afficherMatrice ( matrice *m* )

Définition à la ligne 125 du fichier matrice.c.

##### 5.3.2.2 matrice cloneMatrice ( matrice *m* )

Définition à la ligne 65 du fichier matrice.c.

##### 5.3.2.3 matrice creerMatriceDesPoints ( point liste[], int dimension )

Définition à la ligne 39 du fichier matrice.c.

**5.3.2.4 `matrice creerMatriceTSP ( char * fnom )`**

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

**5.3.2.5 `matrice creerMatriceVide ( int dimension )`**

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

**5.3.2.6 `void detruireMatrice ( matrice m )`**

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

**5.3.2.7 `int getDimensionMatrice ( matrice m )`**

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

**5.3.2.8 `float getDistanceIndice ( matrice m, int ref1, int ref2 )`**

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

**5.3.2.9 `float getDistancePoint ( matrice m, point p1, point p2 )`**

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

**5.3.2.10 `int getIndicePoint ( matrice m, point p )`**

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

**5.3.2.11 `point getPointIndice ( matrice m, int indice )`**

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

**5.3.2.12 `point* getTableauPointsMatrice ( matrice m )`**

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

**5.3.2.13 `int lowerBound ( matrice m )`**

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

**5.3.2.14 `void markAsInfinite ( matrice m, int i, int j )`**

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

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

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

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

Définition à la ligne 112 du fichier matrice.c.

## 5.4 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*)

#### 5.4.1 Documentation des fonctions

##### 5.4.1.1 void afficherListeDesPoints ( point \* *p*, int *len* )

Définition à la ligne 97 du fichier point.c.

##### 5.4.1.2 void afficherPoint ( point *p* )

Définition à la ligne 90 du fichier point.c.

##### 5.4.1.3 point clone ( point *p* )

Définition à la ligne 45 du fichier point.c.

##### 5.4.1.4 point creerPoint ( signed int *x*, signed int *y* )

Définition à la ligne 22 du fichier point.c.

**5.4.1.5 void detruirePoint ( point *p* )**

Définition à la ligne 33 du fichier point.c.

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

Définition à la ligne 110 du fichier point.c.

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

Définition à la ligne 104 du fichier point.c.

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

Définition à la ligne 40 du fichier point.c.

**5.4.1.9 signed int getX ( point *p* )**

Définition à la ligne 52 du fichier point.c.

**5.4.1.10 signed int getY ( point *p* )**

Définition à la ligne 57 du fichier point.c.

**5.4.1.11 bool isVisited ( point *p* )**

Définition à la ligne 62 du fichier point.c.

**5.4.1.12 void markNoVisited ( point *p* )**

Définition à la ligne 85 du fichier point.c.

**5.4.1.13 void markVisited ( point *p* )**

Définition à la ligne 79 du fichier point.c.

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

Définition à la ligne 67 du fichier point.c.

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

Définition à la ligne 73 du fichier point.c.

**5.5 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*)

### 5.5.1 Documentation des définitions de type

#### 5.5.1.1 typedef struct **Point**\* **point**

Définition à la ligne 6 du fichier point.h.

### 5.5.2 Documentation des fonctions

#### 5.5.2.1 void **afficherListeDesPoints** ( **point** \* *p*, int *len* )

Définition à la ligne 97 du fichier point.c.

#### 5.5.2.2 void **afficherPoint** ( **point** *p* )

Définition à la ligne 90 du fichier point.c.

#### 5.5.2.3 **point clone** ( **point** *p* )

Définition à la ligne 45 du fichier point.c.

#### 5.5.2.4 **point creerPoint** ( signed int *x*, signed int *y* )

Définition à la ligne 22 du fichier point.c.

#### 5.5.2.5 void **detruirePoint** ( **point** *p* )

Définition à la ligne 33 du fichier point.c.

#### 5.5.2.6 float **distanceEntreDeuxPoints** ( **point** *p1*, **point** *p2* )

Définition à la ligne 110 du fichier point.c.



#### 5.5.2.7 float distanceManhattan ( point *p1*, point *p2* )

Définition à la ligne 104 du fichier point.c.

#### 5.5.2.8 bool equals ( point *p1*, point *p2* )

Définition à la ligne 40 du fichier point.c.

#### 5.5.2.9 signed int getX ( point *p* )

Définition à la ligne 52 du fichier point.c.

#### 5.5.2.10 signed int getY ( point *p* )

Définition à la ligne 57 du fichier point.c.

#### 5.5.2.11 bool isVisited ( point *p* )

Définition à la ligne 62 du fichier point.c.

#### 5.5.2.12 void markNoVisited ( point *p* )

Définition à la ligne 85 du fichier point.c.

#### 5.5.2.13 void markVisited ( point *p* )

Définition à la ligne 79 du fichier point.c.

#### 5.5.2.14 void setX ( point *p*, signed int *x* )

Définition à la ligne 67 du fichier point.c.

#### 5.5.2.15 void setY ( point *p*, signed int *y* )

Définition à la ligne 73 du fichier point.c.

## 5.6 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. 11) 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)
- int **overallDistanceVerbose** (matrice m, point \*points)
- void **swap** (point \*plist, int i, int j)
- void **copyList** (point \*pln, point \*pOut, int len)
- void **copyListIndice** (point \*pln, point \*pOut, int start, int end)
- void **deleteFromList** (point \*in, int length, point p)
- void **bruteForceRough** (matrice m, point \*pln, int i, int n, int \*min, point \*pOut)
- point \* **bruteForce** (matrice m)
- int **prim** (matrice mln, point \*TabVisite)
- point \* **branchBound** (matrice m)

## 5.6.1 Documentation des fonctions

### 5.6.1.1 point\* branchBound ( matrice m )

Définition à la ligne 282 du fichier projetAlgo.c.

### 5.6.1.2 point\* bruteForce ( matrice m )

Définition à la ligne 209 du fichier projetAlgo.c.

### 5.6.1.3 void bruteForceRough ( matrice m, point \* pln, int i, int n, int \* min, point \* pOut )

Définition à la ligne 182 du fichier projetAlgo.c.

### 5.6.1.4 void copyList ( point \* pln, point \* pOut, int len )

Définition à la ligne 150 du fichier projetAlgo.c.

### 5.6.1.5 void copyListIndice ( point \* pln, point \* pOut, int start, int end )

Définition à la ligne 157 du fichier projetAlgo.c.

### 5.6.1.6 void deleteFromList ( point \* in, int length, point p )

Définition à la ligne 166 du fichier projetAlgo.c.

### 5.6.1.7 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.
----------	--

Renvois

ordreDePassage[] tableau de sortie qui contiendra le nouveau parcours

Définition à la ligne 67 du fichier projetAlgo.c.

5.6.1.8 int overallDistance ( matrice *m*, point \* *points* )

Définition à la ligne 104 du fichier projetAlgo.c.

5.6.1.9 int overallDistanceVerbose ( matrice *m*, point \* *points* )

Définition à la ligne 116 du fichier projetAlgo.c.

5.6.1.10 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. 11) 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.

5.6.1.11 int prim ( matrice *mIn*, point \* *TabVisite* )

Définition à la ligne 231 du fichier projetAlgo.c.

5.6.1.12 void swap ( point \* *plist*, int *i*, int *j* )

Définition à la ligne 138 du fichier projetAlgo.c.

## 5.7 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. 11) 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*)
- int **overallDistanceVerbose** (matrice *m*, point \**points*)
- point \* **bruteForce** (matrice *m*)
- void **copyList** (point \**pIn*, point \**pOut*, int *len*)
- void **copyListIndice** (point \**pIn*, point \**pOut*, int *start*, int *end*)
- point \* **branchBound** (matrice *m*)
- int **prim** (matrice *m*, point \**TabVisite*)

### 5.7.1 Documentation des fonctions

#### 5.7.1.1 `point* branchBound ( matrice m )`

Définition à la ligne 282 du fichier projetAlgo.c.

#### 5.7.1.2 `point* bruteForce ( matrice m )`

Définition à la ligne 209 du fichier projetAlgo.c.

#### 5.7.1.3 `void copyList ( point * pln, point * pOut, int len )`

Définition à la ligne 150 du fichier projetAlgo.c.

#### 5.7.1.4 `void copyListIndice ( point * pln, point * pOut, int start, int end )`

Définition à la ligne 157 du fichier projetAlgo.c.

#### 5.7.1.5 `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.

#### 5.7.1.6 `int overallDistance ( matrice m, point * points )`

Définition à la ligne 104 du fichier projetAlgo.c.

#### 5.7.1.7 `int overallDistanceVerbose ( matrice m, point * points )`

Définition à la ligne 116 du fichier projetAlgo.c.

#### 5.7.1.8 `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. 11) 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.

**5.7.1.9 int prim ( matrice *m*, point \* *TabVisite* )**

Définition à la ligne 231 du fichier projetAlgo.c.

**5.8 Référence du fichier test\_BF.c**

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

**Macros**

— #define **nombreDePoints** 4

**Fonctions**

— int **main** ()

**5.8.1 Documentation des macros****5.8.1.1 #define nombreDePoints 4**

Définition à la ligne 8 du fichier test\_BF.c.

**5.8.2 Documentation des fonctions****5.8.2.1 int main ( )**

Définition à la ligne 9 du fichier test\_BF.c.

**5.9 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** ()

### 5.9.1 Documentation des fonctions

#### 5.9.1.1 int main ( )

Définition à la ligne 8 du fichier test\_NN.c.

## 5.10 Référence du fichier test\_Prim.c

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

## Macros

— #define **nombreDePoints** 4

## Fonctions

— int **main** ()

### 5.10.1 Documentation des macros

#### 5.10.1.1 #define nombreDePoints 4

Définition à la ligne 8 du fichier test\_Prim.c.

### 5.10.2 Documentation des fonctions

#### 5.10.2.1 int main ( )

Définition à la ligne 9 du fichier test\_Prim.c.

## 5.11 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[])

### 5.11.1 Documentation des macros

#### 5.11.1.1 `#define _GNU_SOURCE`

Définition à la ligne 1 du fichier tsplOtourO.c.

### 5.11.2 Documentation des fonctions

#### 5.11.2.1 **matrice** **creerMatriceTSP** ( char \* *fileName* )

Définition à la ligne 11 du fichier tsplOtourO.c.

#### 5.11.2.2 void **creerTOUR** ( char \* *fileName*, **matrice** *m*, **point** *liste*[] )

Définition à la ligne 90 du fichier tsplOtourO.c.

#### 5.11.2.3 void **creerTSPMatrice** ( char \* *fileName*, **matrice** *m* )

Définition à la ligne 61 du fichier tsplOtourO.c.

## 5.12 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[])

### 5.12.1 Documentation des fonctions

#### 5.12.1.1 **matrice** **creerMatriceTSP** ( char \* *fnom* )

Définition à la ligne 11 du fichier tsplOtourO.c.

#### 5.12.1.2 void **creerTOUR** ( char \* *fnom*, **matrice** *m*, **point** *liste*[] )

Définition à la ligne 90 du fichier tsplOtourO.c.

5.12.1.3 void creerTSPMatrice ( char \* *fnom*, matrice *m* )

Définition à la ligne 61 du fichier tsplOtourO.c.



# Index

- `_GNU_SOURCE`
  - `tsplOtourO.c`, 23
- `afficherListeDesPoints`
  - `point.c`, 14
  - `point.h`, 16
- `afficherMatrice`
  - `matrice.c`, 10
  - `matrice.h`, 12
- `afficherPoint`
  - `point.c`, 14
  - `point.h`, 16
- `branchBound`
  - `projetAlgo.c`, 18
  - `projetAlgo.h`, 20
- `bruteForce`
  - `projetAlgo.c`, 18
  - `projetAlgo.h`, 20
- `bruteForceRough`
  - `projetAlgo.c`, 18
- `clone`
  - `point.c`, 14
  - `point.h`, 16
- `cloneMatrice`
  - `matrice.c`, 10
  - `matrice.h`, 12
- `copyList`
  - `projetAlgo.c`, 18
  - `projetAlgo.h`, 20
- `copyListIndice`
  - `projetAlgo.c`, 18
  - `projetAlgo.h`, 20
- `creerMatriceDesPoints`
  - `matrice.c`, 10
  - `matrice.h`, 12
- `creerMatriceTSP`
  - `matrice.h`, 12
  - `tsplOtourO.c`, 23
  - `tsplOtourO.h`, 23
- `creerMatriceVide`
  - `matrice.c`, 10
  - `matrice.h`, 13
- `creerPoint`
  - `point.c`, 14
  - `point.h`, 16
- `creerTOUR`
  - `tsplOtourO.c`, 23
  - `tsplOtourO.h`, 23
- `creerTSPMatrice`
  - `tsplOtourO.c`, 23
  - `tsplOtourO.h`, 23
- `deleteFromList`
  - `projetAlgo.c`, 18
- `detruireMatrice`
  - `matrice.c`, 11
  - `matrice.h`, 13
- `detruirePoint`
  - `point.c`, 14
  - `point.h`, 16
- `dimension`
  - `Matrice`, 7
- `distanceEntreDeuxPoints`
  - `point.c`, 15
  - `point.h`, 16
- `distanceManhattan`
  - `point.c`, 15
  - `point.h`, 16
- `equals`
  - `point.c`, 15
  - `point.h`, 17
- `erreurArguments`
  - `main.c`, 9
- `findMin`
  - `matrice.c`, 11
- `getDimensionMatrice`
  - `matrice.c`, 11
  - `matrice.h`, 13
- `getDistanceIndice`
  - `matrice.c`, 11
  - `matrice.h`, 13
- `getDistancePoint`
  - `matrice.c`, 11
  - `matrice.h`, 13
- `getIndicePoint`
  - `matrice.c`, 11
  - `matrice.h`, 13
- `getPointIndice`
  - `matrice.c`, 11
  - `matrice.h`, 13
- `getTableauPointsMatrice`
  - `matrice.c`, 11
  - `matrice.h`, 13
- `getX`
  - `point.c`, 15

- point.h, 17
- getY
  - point.c, 15
  - point.h, 17
- isVisited
  - point.c, 15
  - point.h, 17
- lowerBound
  - matrice.c, 11
  - matrice.h, 13
- main
  - main.c, 9
  - test\_BF.c, 21
  - test\_NN.c, 22
  - test\_Prim.c, 22
- main.c, 9
  - erreurArguments, 9
  - main, 9
- markAsInfinite
  - matrice.c, 11
  - matrice.h, 13
- markNoVisited
  - point.c, 15
  - point.h, 17
- markVisited
  - point.c, 15
  - point.h, 17
- Matrice, 7
  - dimension, 7
  - ref, 7
  - tab, 7
- matrice
  - matrice.h, 12
- matrice.c, 10
  - afficherMatrice, 10
  - cloneMatrice, 10
  - creerMatriceDesPoints, 10
  - creerMatriceVide, 10
  - detruiMatrice, 11
  - findMin, 11
  - getDimensionMatrice, 11
  - getDistanceIndice, 11
  - getDistancePoint, 11
  - getIndicePoint, 11
  - getPointIndice, 11
  - getTableauPointsMatrice, 11
  - lowerBound, 11
  - markAsInfinite, 11
  - setDistanceIndice, 11
  - setPointIndice, 11
- matrice.h, 12
  - afficherMatrice, 12
  - cloneMatrice, 12
  - creerMatriceDesPoints, 12
  - creerMatriceTSP, 12
  - creerMatriceVide, 13
  - detruiMatrice, 13
  - getDimensionMatrice, 13
  - getDistanceIndice, 13
  - getDistancePoint, 13
  - getIndicePoint, 13
  - getPointIndice, 13
  - getTableauPointsMatrice, 13
  - lowerBound, 13
  - markAsInfinite, 13
  - matrice, 12
  - setDistanceIndice, 13
  - setPointIndice, 13
- nearestNeighbour
  - projetAlgo.c, 18
  - projetAlgo.h, 20
- nombreDePoints
  - test\_BF.c, 21
  - test\_Prim.c, 22
- overallDistance
  - projetAlgo.c, 18
  - projetAlgo.h, 20
- overallDistanceVerbose
  - projetAlgo.c, 19
  - projetAlgo.h, 20
- Point, 7
  - visited, 8
  - x, 8
  - y, 8
- point
  - point.h, 16
- point.c, 14
  - afficherListeDesPoints, 14
  - afficherPoint, 14
  - clone, 14
  - creerPoint, 14
  - detruiPoint, 14
  - distanceEntreDeuxPoints, 15
  - distanceManhattan, 15
  - equals, 15
  - getX, 15
  - getY, 15
  - isVisited, 15
  - markNoVisited, 15
  - markVisited, 15
  - setX, 15
  - setY, 15
- point.h, 15
  - afficherListeDesPoints, 16
  - afficherPoint, 16
  - clone, 16
  - creerPoint, 16
  - detruiPoint, 16
  - distanceEntreDeuxPoints, 16
  - distanceManhattan, 16
  - equals, 17
  - getX, 17

- getY, 17
- isVisited, 17
- markNoVisited, 17
- markVisited, 17
- point, 16
- setX, 17
- setY, 17
- PointLePlusProche
  - projetAlgo.c, 19
  - projetAlgo.h, 20
- prim
  - projetAlgo.c, 19
  - projetAlgo.h, 21
- projetAlgo.c, 17
  - branchBound, 18
  - bruteForce, 18
  - bruteForceRough, 18
  - copyList, 18
  - copyListIndice, 18
  - deleteFromList, 18
  - nearestNeighbour, 18
  - overallDistance, 18
  - overallDistanceVerbose, 19
  - PointLePlusProche, 19
  - prim, 19
  - swap, 19
- projetAlgo.h, 19
  - branchBound, 20
  - bruteForce, 20
  - copyList, 20
  - copyListIndice, 20
  - nearestNeighbour, 20
  - overallDistance, 20
  - overallDistanceVerbose, 20
  - PointLePlusProche, 20
  - prim, 21
- ref
  - Matrice, 7
- setDistanceIndice
  - matrice.c, 11
  - matrice.h, 13
- setPointIndice
  - matrice.c, 11
  - matrice.h, 13
- setX
  - point.c, 15
  - point.h, 17
- setY
  - point.c, 15
  - point.h, 17
- swap
  - projetAlgo.c, 19
- tab
  - Matrice, 7
- test\_BF.c, 21
  - main, 21
- nombreDePoints, 21
- test\_NN.c, 21
  - main, 22
- test\_Prim.c, 22
  - main, 22
  - nombreDePoints, 22
- tsplOtourO.c, 22
  - \_GNU\_SOURCE, 23
  - creerMatriceTSP, 23
  - creerTOUR, 23
  - creerTSPMatrice, 23
- tsplOtourO.h, 23
  - creerMatriceTSP, 23
  - creerTOUR, 23
  - creerTSPMatrice, 23
- visited
  - Point, 8
- x
  - Point, 8
- y
  - Point, 8