

## My Project

Generated by Doxygen 1.8.11



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
3.1	Donnee Struct Reference	5
3.1.1	Detailed Description	5
3.2	Enumeration Class Reference	6
3.2.1	Constructor & Destructor Documentation	6
3.2.1.1	Enumeration(Donnee data)	6
3.2.2	Member Function Documentation	6
3.2.2.1	addToEnum(int array[], int size)	6
3.2.2.2	capaciteSuffisante(int array[], int size)	6
3.2.2.3	get_m_d()	7
3.2.2.4	get_m_ListeTournees()	7
3.2.2.5	powerset(int n)	7
3.2.2.6	set_m_ListeTournees(Tournee tour)	7
3.2.2.7	size()	8
3.3	Tournee Class Reference	8
3.3.1	Member Function Documentation	8
3.3.1.1	add(int i, Donnee d)	8
3.3.1.2	calculeDistanceTour(std::vector< int > tour, Donnee d)	9
3.3.1.3	calculePlusPetiteDistancePerm(std::vector< int > tour, Donnee d)	9
3.3.1.4	getDistance()	9
3.3.1.5	getPermutationMin()	9
3.3.1.6	getPointsdeau()	10
3.3.1.7	setDistance(Donnee d)	10
3.3.1.8	setPermutationMin(Donnee d)	10
3.3.1.9	size()	10

<b>4 File Documentation</b>	<b>11</b>
4.1 Donnee.h File Reference . . . . .	11
4.2 Enumeration.cpp File Reference . . . . .	11
4.2.1 Detailed Description . . . . .	11
4.3 Enumeration.h File Reference . . . . .	11
4.3.1 Detailed Description . . . . .	12
4.4 main.cpp File Reference . . . . .	12
4.4.1 Detailed Description . . . . .	12
4.4.2 Function Documentation . . . . .	12
4.4.2.1 lecture_data(char *file, Donnee *d) . . . . .	12
4.5 partitionEnsemble.cpp File Reference . . . . .	13
4.5.1 Detailed Description . . . . .	13
4.5.2 Function Documentation . . . . .	13
4.5.2.1 partitionEnsemble(Enumeration enum, int nbLieux) . . . . .	13
4.6 Tournee.cpp File Reference . . . . .	13
4.6.1 Detailed Description . . . . .	13
4.7 Tournee.h File Reference . . . . .	14
4.7.1 Detailed Description . . . . .	14
<b>Index</b>	<b>15</b>

# Chapter 1

## Class Index

### 1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">Donnee</a>	Structure définissant les données du problème . . . . .	<a href="#">5</a>
<a href="#">Enumeration</a>	. . . . .	<a href="#">6</a>
<a href="#">Tournee</a>	. . . . .	<a href="#">8</a>



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

<a href="#">Donnee.h</a>	Définie les données du problème à résoudre, c'est à dire le nombre de points d'eau, la capacité du réservoir du drone accompagné du distancier . . . . .	11
<a href="#">Enumeration.cpp</a>	Définie une énumération des différentes tournées que le drone peut réaliser sans vider son réservoir . . . . .	11
<a href="#">Enumeration.h</a>	Définie une énumération des différentes tournées que le drone peut réaliser sans vider son réservoir . . . . .	11
<a href="#">main.cpp</a>	Permet de résoudre le problème du projet . . . . .	12
<a href="#">partitionEnsemble.cpp</a>	Résolution d'une instance de problème de partitionnement d'ensemble . . . . .	13
<a href="#">partitionEnsemble.h</a>		??
<a href="#">Tournee.cpp</a>	Définie une tournée de points d'eau, c'est à dire la suite que le drone va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir . . . . .	13
<a href="#">Tournee.h</a>	Définie une tournée de points d'eau, c'est à dire la suite que le drone va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir . . . . .	14





## Chapter 3

# Class Documentation

### 3.1 Donnee Struct Reference

Structure définissant les données du problème.

```
#include <Donnee.h>
```

#### Public Attributes

- int [m\\_nombre\\_lieux](#)  
*Nombre de lieux (incluant le village)*
- int [m\\_capacite](#)  
*Capacité du reservoir du drone.*
- int \* [m\\_quantite\\_eau](#)  
*Demande de chaque lieu (la case 0 est inutilisée car le dépôt n'a aucune demande à voir satisfaire)*
- int \*\* [m\\_distancier](#)  
*distancier (les lignes et colonnes 0 correspondent au dépôt)*

#### 3.1.1 Detailed Description

Structure définissant les données du problème.

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

- [Donnee.h](#)

## 3.2 Enumeration Class Reference

### Public Member Functions

- [Enumeration](#) ([Donnee](#) data)  
*Constructeur.*
- [Donnee](#) [get\\_m\\_d](#) ()  
*permet d'accéder aux données du problème*
- void [set\\_m\\_ListeTournées](#) ([Tournée](#) tour)  
*permet d'ajouter une tournée à l'ensemble de tournées à visiter*
- void [addToEnum](#) (int array[], int [size](#))  
*permet de créer une tournée et de l'ajouter à l'énumération*
- std::vector< [Tournée](#) > [get\\_m\\_ListeTournées](#) ()  
*permet d'accéder à la liste des tournées*
- bool [capaciteSuffisante](#) (int array[], int [size](#))  
*permet de déterminer si le réservoir du drone à une contenance suffisante pour visiter tout les points de la tournée*
- void [powerset](#) (int n)  
*permet de déterminer l'ensemble des différentes tournées que le drone peut réaliser sans vider son réservoir*
- int [size](#) ()  
*permet d'accéder au nombre nombre de tournée qu'il est possible de faire*

### 3.2.1 Constructor & Destructor Documentation

#### 3.2.1.1 Enumeration::Enumeration ( [Donnee](#) data )

Constructeur.

Parameters

<i>data</i>	données du problème
-------------	---------------------

### 3.2.2 Member Function Documentation

#### 3.2.2.1 void Enumeration::addToEnum ( int array[], int [size](#) )

permet de créer une tournée et de l'ajouter à l'énumération

Parameters

<i>array[]</i>	tableau contenant des indices de points d'eau
<i>size</i>	intervalle sur lequel on doit considérer le tableau

#### 3.2.2.2 bool Enumeration::capaciteSuffisante ( int array[], int [size](#) )

permet de déterminer si le réservoir du drone à une contenance suffisante pour visiter tout les points de la tournée

## Parameters

<i>array[]</i>	tableau contenant des indices de points d'eau
<i>size</i>	intervalle sur lequel on doit considérer le tableau

## Returns

bool retourne oui si le drone à une capacite suffisante pour visiter tout les points d'eau dont l'indice est donné dans le tableau

**3.2.2.3** `Donnee Enumeration::get_m_d ( )`

permet d'accéder aux données du problème

## Returns

m\_d données du problème

**3.2.2.4** `vector< Tournee > Enumeration::get_m_ListeTournees ( )`

permet d'accéder à la liste des tournées

## Returns

m\_ListeTournees vecteur des tournées

**3.2.2.5** `void Enumeration::powerset ( int n )`

permet de determiner l'ensemble des différentes tournées que le drone peut réaliser sans vider son réservoir

## Parameters

<i>n</i>	taille de l'ensemble à traiter
----------	--------------------------------

**3.2.2.6** `void Enumeration::set_m_ListeTournees ( Tournee tour )`

permet d'ajouter une tournée à l'ensemble de tournées à visiter

## Parameters

<i>Tournee</i>	tournée à ajouter à l'énumération
----------------	-----------------------------------

### 3.2.2.7 int Enumeration::size ( )

permet d'accéder au nombre nombre de tournée qu'il est possible de faire

#### Returns

m\_ListeTournees.size()

The documentation for this class was generated from the following files:

- [Enumeration.h](#)
- [Enumeration.cpp](#)

## 3.3 Tournee Class Reference

### Public Member Functions

- [Tournee](#) ()  
*Constructeur de [Tournee](#).*
- int [size](#) ()  
*Retourne le nombre de points d'eau visités.*
- std::vector< int > [getPointsdeau](#) ()  
*Getter sur m\_pointsdeau.*
- void [setPermutationMin](#) ([Donnee](#) d)  
*permet de mettre à jour la valeur de la permutation offrant la plus petite distance à parcourir*
- std::vector< int > [getPermutationMin](#) ()  
*Getter sur m\_permutationMin.*
- int [getDistance](#) ()  
*Getter sur m\_distance.*
- void [setDistance](#) ([Donnee](#) d)  
*permet de mettre à jour la plus petite distance à parcourir afin de visiter tout les points d'eau de la tournée*
- void [add](#) (int i, [Donnee](#) d)  
*permet d'ajouter l'indice d'un nouveau point d'eau à visiter pendant la tournée*
- std::vector< int > [calculePlusPetiteDistancePerm](#) (std::vector< int > tour, [Donnee](#) d)  
*calcule la permutation offrant la plus petite distance à parcourir en brute force*
- int [calculeDistanceTour](#) (std::vector< int > tour, [Donnee](#) d)  
*calcule la distance parcourue pendant le tour fourni*

### 3.3.1 Member Function Documentation

#### 3.3.1.1 void Tournee::add ( int *i*, [Donnee](#) *d* )

permet d'ajouter l'indice d'un nouveau point d'eau à visiter pendant la tournée

#### Parameters

<i>i</i>	indice du point d'eau à visiter pendant la tournée
<i>d</i>	données du problème

### 3.3.1.2 int Tournée::calculeDistanceTour ( std::vector< int > *tour*, Donnée *d* )

calcule la distance parcourue pendant le tour fourni

#### Parameters

<i>tour</i>	indices des points d'eau visités
<i>d</i>	données du problème

#### Returns

res la distance entre deux points

### 3.3.1.3 vector< int > Tournée::calculePlusPetiteDistancePerm ( std::vector< int > *tour*, Donnée *d* )

calcule la permutation offrant la plus petite distance à parcourir en brute force

#### Parameters

<i>tour</i>	indices des points d'eau visités
<i>d</i>	données du problème

#### Returns

meilleur\_tour permutation minimale

### 3.3.1.4 int Tournée::getDistance ( )

Getter sur m\_distance.

#### Returns

m\_distance

### 3.3.1.5 vector< int > Tournée::getPermutationMin ( )

Getter sur m\_permutationMin.

#### Returns

m\_permutationMin

**3.3.1.6** `vector< int > Tournee::getPointsdeau ( )`

Getter sur m\_pointsdeau.

**Returns**

m\_pointsdeau

**3.3.1.7** `void Tournee::setDistance ( Donnee d )`

permet de mettre à jour la plus petite distance à parcourir afin de visiter tout les points d'eau de la tournée

**Parameters**

<i>d</i>	données du problème
----------	---------------------

**3.3.1.8** `void Tournee::setPermutationMin ( Donnee d )`

permet de mettre à jour la valeur de la permutation offrant la plus petite distance à parcourir

**Parameters**

<i>d</i>	donnees du problème
----------	---------------------

**3.3.1.9** `int Tournee::size ( )`

Retourne le nombre de points d'eau visités.

**Returns**

m\_pointsdeau.size()

The documentation for this class was generated from the following files:

- [Tournee.h](#)
- [Tournee.cpp](#)

## Chapter 4

# File Documentation

### 4.1 Donnee.h File Reference

Définie les données du problème à résoudre, c'est à dire le nombre de points d'eau, la capacité du réservoir du drone accompagné du distancier.

```
#include <vector>
#include <string>
Include dependency graph for Donnee.h:
```

### 4.2 Enumeration.cpp File Reference

Définie une énumération des différentes tournées que le drone peut réaliser sans vider son réservoir.

```
#include <vector>
#include "Enumeration.h"
Include dependency graph for Enumeration.cpp:
```

#### 4.2.1 Detailed Description

Définie une énumération des différentes tournées que le drone peut réaliser sans vider son réservoir.

### 4.3 Enumeration.h File Reference

Définie une énumération des différentes tournées que le drone peut réaliser sans vider son réservoir.

```
#include <vector>
#include "Donnee.h"
#include "Tournee.h"
Include dependency graph for Enumeration.h: This graph shows which files directly or indirectly include this file:
```

## Classes

- class [Enumeration](#)

### 4.3.1 Detailed Description

Définie une énumération des différentes tournées que le drone peut réaliser sans vider son réservoir.

## 4.4 main.cpp File Reference

Permet de résoudre le problème du projet.

```
#include <iostream>
#include "stdio.h"
#include <stdlib.h>
#include <string.h>
#include <string>
#include <fstream>
#include <glpk.h>
#include <math.h>
#include "Donnee.h"
#include "Tournée.h"
#include "Enumeration.h"
#include "partitionEnsemble.h"
#include <ctime>
```

Include dependency graph for main.cpp:

## Functions

- void [lecture\\_data](#) (char \*file, [Donnee](#) \*d)  
*Lecture des données.*
- int **main** (int argc, char \*argv[])

### 4.4.1 Detailed Description

Permet de résoudre le problème du projet.

### 4.4.2 Function Documentation

#### 4.4.2.1 void [lecture\\_data](#) ( char \* *file*, [Donnee](#) \* *d* )

Lecture des données.

#### Parameters

* <i>file</i>	Le chemin du fichier de données
* <i>d</i>	Les données à remplir



## 4.5 partitionEnsemble.cpp File Reference

Résolution d'une instance de problème de partitionnement d'ensemble.

```
#include "partitionEnsemble.h"
```

Include dependency graph for partitionEnsemble.cpp:

### Functions

- void [partitionEnsemble](#) ([Enumeration](#) enume, int nbLieux)

*//fonction permettant de résoudre le problème posé en utilisant GLPK et de déterminer les tournées que le drone doit choisir afin d'avoir un itinéraire optimal*

#### 4.5.1 Detailed Description

Résolution d'une instance de problème de partitionnement d'ensemble.

#### 4.5.2 Function Documentation

##### 4.5.2.1 void partitionEnsemble ( [Enumeration](#) enume, int nbLieux )

*//fonction permettant de résoudre le problème posé en utilisant GLPK et de déterminer les tournées que le drone doit choisir afin d'avoir un itinéraire optimal*

##### Parameters

<i>enum</i>	<a href="#">Enumeration</a> de l'ensemble des tournées possible
<i>nbLieux</i>	Le nombre de lieux du problème

## 4.6 Tournee.cpp File Reference

Définie une tournée de points d'eau, c'est à dire la suite que le drone va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir.

```
#include "Tournee.h"
```

Include dependency graph for Tournee.cpp:

#### 4.6.1 Detailed Description

Définie une tournée de points d'eau, c'est à dire la suite que le drone va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir.

## 4.7 Tournee.h File Reference

Définie une tournée de points d'eau, c'est à dire la suite que le drone va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir.

```
#include <iostream>
#include <vector>
#include <algorithm>
#include "Donnee.h"
```

Include dependency graph for Tournee.h: This graph shows which files directly or indirectly include this file:

### Classes

- class [Tournee](#)

### 4.7.1 Detailed Description

Définie une tournée de points d'eau, c'est à dire la suite que le drone va pouvoir visiter en une seule fois sans avoir à aller vider son réservoir.

# Index

- add
  - Tournee, [8](#)
- addToEnum
  - Enumeration, [6](#)
- calculeDistanceTour
  - Tournee, [9](#)
- calculePlusPetiteDistancePerm
  - Tournee, [9](#)
- capaciteSuffisante
  - Enumeration, [6](#)
- Donnee, [5](#)
- Donnee.h, [11](#)
- Enumeration, [6](#)
  - addToEnum, [6](#)
  - capaciteSuffisante, [6](#)
  - Enumeration, [6](#)
  - get\_m\_ListeTournees, [7](#)
  - get\_m\_d, [7](#)
  - powerset, [7](#)
  - set\_m\_ListeTournees, [7](#)
  - size, [7](#)
- Enumeration.cpp, [11](#)
- Enumeration.h, [11](#)
- get\_m\_ListeTournees
  - Enumeration, [7](#)
- get\_m\_d
  - Enumeration, [7](#)
- getDistance
  - Tournee, [9](#)
- getPermutationMin
  - Tournee, [9](#)
- getPointsdeau
  - Tournee, [9](#)
- lecture\_data
  - main.cpp, [12](#)
- main.cpp, [12](#)
  - lecture\_data, [12](#)
- partitionEnsemble
  - partitionEnsemble.cpp, [13](#)
- partitionEnsemble.cpp, [13](#)
  - partitionEnsemble, [13](#)
- powerset
  - Enumeration, [7](#)
- set\_m\_ListeTournees
  - Enumeration, [7](#)
- setDistance
  - Tournee, [10](#)
- setPermutationMin
  - Tournee, [10](#)
- size
  - Enumeration, [7](#)
  - Tournee, [10](#)
- Tournee, [8](#)
  - add, [8](#)
  - calculeDistanceTour, [9](#)
  - calculePlusPetiteDistancePerm, [9](#)
  - getDistance, [9](#)
  - getPermutationMin, [9](#)
  - getPointsdeau, [9](#)
  - setDistance, [10](#)
  - setPermutationMin, [10](#)
  - size, [10](#)
- Tournee.cpp, [13](#)
- Tournee.h, [14](#)