

Calculatrice C++

2

Généré par Doxygen 1.8.6

Dimanche 8 Février 2015 13 :47 :54

Table des matières

1	Index des fichiers	1
1.1	Liste des fichiers	1
2	Documentation des fichiers	3
2.1	Référence du fichier addition.cpp	3
2.1.1	Documentation des fonctions	3
2.1.1.1	ajouter	3
2.1.1.2	ajouter	3
2.1.1.3	ajouter	4
2.1.1.4	ajouter	4
2.2	Référence du fichier division.cpp	5
2.2.1	Documentation des fonctions	5
2.2.1.1	division	5
2.2.1.2	division	5
2.2.1.3	division	5
2.2.1.4	division	6
2.3	Référence du fichier main.cpp	6
2.3.1	Documentation des fonctions	6
2.3.1.1	main	6
2.4	Référence du fichier modulo.cpp	7
2.4.1	Documentation des fonctions	7
2.4.1.1	modulo	7
2.4.1.2	modulo	7
2.5	Référence du fichier multiplication.cpp	8
2.5.1	Documentation des fonctions	8
2.5.1.1	multiplication	8
2.5.1.2	multiplication	8
2.5.1.3	multiplication	9
2.5.1.4	multiplication	9
2.6	Référence du fichier soustraction.cpp	10
2.6.1	Documentation des fonctions	10

2.6.1.1	soustraction	10
2.6.1.2	soustraction	10
2.6.1.3	soustraction	10
2.6.1.4	soustraction	11

Chapitre 1

Documentation Calculatrice Cpp

1.1 Introduction

Documentation d'un programme écrit en C++ qui doit permettre de réaliser les calculs de type +, -, x, ÷, modulo. Utilisant les types de variables int32, int64, float et double.

Chapitre 2

Index des fichiers

2.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

addition.cpp	3
division.cpp	5
main.cpp	6
modulo.cpp	7
multiplication.cpp	8
soustraction.cpp	10

Chapitre 3

Documentation des fichiers

3.1 Référence du fichier addition.cpp

Fonctions

- `int32_t ajouter (int32_t operandeA, int32_t operandeB)`
Fonction d'addition de deux nombre de type `int32_t`.
- `int64_t ajouter (int64_t operandeA, int64_t operandeB)`
Fonction d'addition de deux nombre de type `int64_t`.
- `float ajouter (float operandeA, float operandeB)`
Fonction d'addition de deux nombre de type `Float`.
- `double ajouter (double operandeA, double operandeB)`
Fonction d'addition de deux nombre de type `Double`.

3.1.1 Documentation des fonctions

3.1.1.1 `int32_t ajouter (int32_t operandeA, int32_t operandeB)`

Fonction d'addition de deux nombre de type `int32_t`.

Paramètres

<code>operandeA</code>	est un entier sur 32 bits
<code>operandeB</code>	est un entier sur 32 bits

Renvoie

La somme des deux entiers 32 bits saisie par l'utilisateur.

Voir également

`ajouter(int64_t, int64_t)`, `ajouter(float, float)`, `ajouter(double, double)`

```
11 {  
12     return operandeA + operandeB;  
13 }
```

3.1.1.2 `int64_t ajouter (int64_t operandeA, int64_t operandeB)`

Fonction d'addition de deux nombre de type `int64_t`.

Paramètres

<i>operandeA</i>	est un entier sur 64 bits
<i>operandeB</i>	est un entier sur 64 bits

Renvoie

La somme des deux entiers 64 bits saisie par l'utilisateur.

Voir également

[ajouter\(float, float\)](#), [ajouter\(double, double\)](#), [ajouter\(int32_t, int32_t\)](#)

```
24 {  
25     return operandeA + operandeB;  
26 }
```

3.1.1.3 float ajouter (float *operandeA*, float *operandeB*)

Fonction d'addition de deux nombre de type Float.

Paramètres

<i>operandeA</i>	est un Float
<i>operandeB</i>	est un Float

Renvoie

La somme des deux Float saisie par l'utilisateur.

Voir également

[ajouter\(double, double\)](#), [ajouter\(int32_t, int32_t\)](#), [ajouter\(int64_t, int64_t\)](#)

```
37 {  
38     return operandeA + operandeB;  
39 }
```

3.1.1.4 double ajouter (double *operandeA*, double *operandeB*)

Fonction d'addition de deux nombre de type Double.

Paramètres

<i>operandeA</i>	est un Double
<i>operandeB</i>	est un Double

Renvoie

La somme des deux Double saisie par l'utilisateur.

Voir également

[ajouter\(int32_t, int32_t\)](#), [ajouter\(int64_t, int64_t\)](#), [ajouter\(float, float\)](#)

```
50 {  
51     return operandeA + operandeB;  
52 }
```

3.2 Référence du fichier division.cpp

Fonctions

- `int32_t division (int32_t operandeA, int32_t operandeB)`
Fonction de division de deux nombre de type int32_t.
- `int64_t division (int64_t operandeA, int64_t operandeB)`
Fonction de division de deux nombre de type int64_t.
- `float division (float operandeA, float operandeB)`
Fonction de division de deux nombre de type Float.
- `double division (double operandeA, double operandeB)`
Fonction de division de deux nombre de type Double.

3.2.1 Documentation des fonctions

3.2.1.1 `int32_t division (int32_t operandeA, int32_t operandeB)`

Fonction de division de deux nombre de type `int32_t`.

Paramètres

<code>operandeA</code>	est un entier sur 32 bits
<code>operandeB</code>	est un entier sur 32 bits

Renvoie

La division des deux entiers 32 bits saisie par l'utilisateur.

Voir également

`division(int64_t, int64_t)`, `division(float, float)`, `division(double, double)`

```
11 {  
12     return operandeA / operandeB;  
13 }
```

3.2.1.2 `int64_t division (int64_t operandeA, int64_t operandeB)`

Fonction de division de deux nombre de type `int64_t`.

Paramètres

<code>operandeA</code>	est un entier sur 64 bits
<code>operandeB</code>	est un entier sur 64 bits

Renvoie

La division des deux entiers 64 bits saisie par l'utilisateur.

Voir également

`division(float, float)`, `division(double, double)`, `division(int32_t, int32_t)`

```
24 {  
25     return operandeA / operandeB;  
26 }
```

3.2.1.3 `float division (float operandeA, float operandeB)`

Fonction de division de deux nombre de type `Float`.

Paramètres

<i>operandeA</i>	est un Float
<i>operandeB</i>	est un Float

Renvoie

La division des deux Float saisie par l'utilisateur.

Voir également

[division\(double, double\)](#), [division\(int32_t, int32_t\)](#), [division\(int64_t, int64_t\)](#)

```
37 {
38     return operandeA / operandeB;
39 }
```

3.2.1.4 double division (double *operandeA*, double *operandeB*)

Fonction de division de deux nombre de type Double.

Paramètres

<i>operandeA</i>	est un Double
<i>operandeB</i>	est un Double

Renvoie

La division des deux Double saisie par l'utilisateur.

Voir également

[division\(int32_t, int32_t\)](#), [division\(int64_t, int64_t\)](#), [division\(float, float\)](#)

```
50 {
51     return operandeA * operandeB;
52 }
```

3.3 Référence du fichier main.cpp

```
#include <iostream>
#include <stdint.h>
#include "../addition.cpp"
#include "../soustraction.cpp"
#include "../multiplication.cpp"
#include "../division.cpp"
#include "../modulo.cpp"
```

Fonctions

— `int main ()`

3.3.1 Documentation des fonctions**3.3.1.1 int main ()**

```
26 {
```

```

27 // Affichage titre Calculatrice C++ :
28 cout << "---\nCalculatrice C++\n---\n\n";
29 int16_t a=321, b=23;
30
31 int32_t c=321, d=23;
32
33 float e=2324, f=25;
34
35 double g=67863, h=432;
36
37 cout << "Valeur pour calcul : " << endl;
38 cout << "Valeur de a : " << a << endl;
39 cout << "Valeur de b : " << b << endl;
40 cout << "Valeur de c : " << c << endl;
41 cout << "Valeur de d : " << d << endl;
42 cout << "Valeur de e : " << e << endl;
43 cout << "Valeur de f : " << f << endl;
44 cout << "Valeur de g : " << g << endl;
45 cout << "Valeur de h : " << h << endl << endl;
46
47
48 cout << "Addition : " << ajouter(a,b) << endl;
49 cout << "Soustraction : " << soustraction(a,b) << endl;
50 cout << "Multiplication : " << multiplication(a,b) << endl;
51 cout << "Division : " << division(a,b) << endl;
52 cout << "Modulo : " << modulo(a,b) << endl;
53
54 return 0;
55 }

```

3.4 Référence du fichier modulo.cpp

Fonctions

- `int32_t modulo (int32_t operandeA, int32_t operandeB)`
Fonction de modulo de deux nombre de type int32_t.
- `int64_t modulo (int64_t operandeA, int64_t operandeB)`
Fonction de modulo de deux nombre de type int64_t.

3.4.1 Documentation des fonctions

3.4.1.1 `int32_t modulo (int32_t operandeA, int32_t operandeB)`

Fonction de modulo de deux nombre de type `int32_t`.

Paramètres

<i>operandeA</i>	est un entier sur 32 bits
<i>operandeB</i>	est un entier sur 32 bits

Renvoie

Le modulo des deux entiers 32 bits saisie par l'utilisateur.

Voir également

`modulo(int64_t, int64_t)`, `modulo(float, float)`, `modulo(double, double)`

```

11 {
12     return operandeA % operandeB;
13 }

```

3.4.1.2 `int64_t modulo (int64_t operandeA, int64_t operandeB)`

Fonction de modulo de deux nombre de type `int64_t`.

Paramètres

<i>operandeA</i>	est un entier sur 64 bits
<i>operandeB</i>	est un entier sur 64 bits

Renvoie

Le modulo des deux entiers 64 bits saisie par l'utilisateur.

Voir également

`modulo(float, float)`, `modulo(double, double)`, [modulo\(int32_t, int32_t\)](#)

```
24 {
25     return operandeA % operandeB;
26 }
```

3.5 Référence du fichier multiplication.cpp

Fonctions

- `int32_t multiplication` (`int32_t operandeA`, `int32_t operandeB`)
Fonction de soustraction de deux nombre de type int32_t.
- `int64_t multiplication` (`int64_t operandeA`, `int64_t operandeB`)
Fonction de multiplication de deux nombre de type int64_t.
- `float multiplication` (`float operandeA`, `float operandeB`)
Fonction de multiplication de deux nombre de type Float.
- `double multiplication` (`double operandeA`, `double operandeB`)
Fonction de multiplication de deux nombre de type Double.

3.5.1 Documentation des fonctions

3.5.1.1 `int32_t multiplication (int32_t operandeA, int32_t operandeB)`

Fonction de soustraction de deux nombre de type `int32_t`.

Paramètres

<i>operandeA</i>	est un entier sur 32 bits
<i>operandeB</i>	est un entier sur 32 bits

Renvoie

La multiplication des deux entiers 32 bits saisie par l'utilisateur.

Voir également

[multiplication\(int64_t, int64_t\)](#), [multiplication\(float, float\)](#), [multiplication\(double, double\)](#)

```
11 {
12     return operandeA * operandeB;
13 }
```

3.5.1.2 `int64_t multiplication (int64_t operandeA, int64_t operandeB)`

Fonction de multiplication de deux nombre de type `int64_t`.

Paramètres

<i>operandeA</i>	est un entier sur 64 bits
<i>operandeB</i>	est un entier sur 64 bits

Renvoie

La multiplication des deux entiers 64 bits saisie par l'utilisateur.

Voir également

[multiplication\(float, float\)](#), [multiplication\(double, double\)](#), [multiplication\(int32_t, int32_t\)](#)

```
24 {  
25     return operandeA * operandeB;  
26 }
```

3.5.1.3 float multiplication (float *operandeA*, float *operandeB*)

Fonction de multiplication de deux nombre de type Float.

Paramètres

<i>operandeA</i>	est un Float
<i>operandeB</i>	est un Float

Renvoie

La multiplication des deux Float saisie par l'utilisateur.

Voir également

[multiplication\(double, double\)](#), [multiplication\(int32_t, int32_t\)](#), [multiplication\(int64_t, int64_t\)](#)

```
37 {  
38     return operandeA * operandeB;  
39 }
```

3.5.1.4 double multiplication (double *operandeA*, double *operandeB*)

Fonction de multiplication de deux nombre de type Double.

Paramètres

<i>operandeA</i>	est un Double
<i>operandeB</i>	est un Double

Renvoie

La multiplication des deux Double saisie par l'utilisateur.

Voir également

[multiplication\(int32_t, int32_t\)](#), [multiplication\(int64_t, int64_t\)](#), [multiplication\(float, float\)](#)

```
50 {  
51     return operandeA * operandeB;  
52 }
```

3.6 Référence du fichier soustraction.cpp

Fonctions

- `int32_t soustraction (int32_t operandeA, int32_t operandeB)`
Fonction de soustraction de deux nombre de type int32_t.
- `int64_t soustraction (int64_t operandeA, int64_t operandeB)`
Fonction de soustraction de deux nombre de type int64_t.
- `float soustraction (float operandeA, float operandeB)`
Fonction de soustraction de deux nombre de type Float.
- `double soustraction (double operandeA, double operandeB)`
Fonction de soustraction de deux nombre de type Double.

3.6.1 Documentation des fonctions

3.6.1.1 `int32_t soustraction (int32_t operandeA, int32_t operandeB)`

Fonction de soustraction de deux nombre de type `int32_t`.

Paramètres

<code>operandeA</code>	est un entier sur 32 bits
<code>operandeB</code>	est un entier sur 32 bits

Renvoie

La soustraction des deux entiers 32 bits saisie par l'utilisateur.

Voir également

[soustraction\(int64_t, int64_t\)](#), [soustraction\(float, float\)](#), [soustraction\(double, double\)](#)

```
11 {
12     return operandeA - operandeB;
13 }
```

3.6.1.2 `int64_t soustraction (int64_t operandeA, int64_t operandeB)`

Fonction de soustraction de deux nombre de type `int64_t`.

Paramètres

<code>operandeA</code>	est un entier sur 64 bits
<code>operandeB</code>	est un entier sur 64 bits

Renvoie

La soustraction des deux entiers 64 bits saisie par l'utilisateur.

Voir également

[soustraction\(float, float\)](#), [soustraction\(double, double\)](#), [soustraction\(int32_t, int32_t\)](#)

```
24 {
25     return operandeA - operandeB;
26 }
```

3.6.1.3 `float soustraction (float operandeA, float operandeB)`

Fonction de soustraction de deux nombre de type `Float`.

Paramètres

<i>operandeA</i>	est un Float
<i>operandeB</i>	est un Float

Renvoie

La soustraction des deux Float saisie par l'utilisateur.

Voir également

[soustraction\(double, double\)](#), [soustraction\(int32_t, int32_t\)](#), [soustraction\(int64_t, int64_t\)](#)

```
37 {  
38     return operandeA - operandeB;  
39 }
```

3.6.1.4 double soustraction (double *operandeA*, double *operandeB*)

Fonction de soustraction de deux nombre de type Double.

Paramètres

<i>operandeA</i>	est un Double
<i>operandeB</i>	est un Double

Renvoie

La soustraction des deux Double saisie par l'utilisateur.

Voir également

[soustraction\(int32_t, int32_t\)](#), [soustraction\(int64_t, int64_t\)](#), [soustraction\(float, float\)](#)

```
50 {  
51     return operandeA - operandeB;  
52 }
```