

labo\_07\_schaufelberger\_yannick\_gallay\_david

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>File Index</b>	<b>1</b>
1.1	File List . . . . .	1
<b>2</b>	<b>File Documentation</b>	<b>3</b>
2.1	constants.h File Reference . . . . .	3
2.1.1	Enumeration Type Documentation . . . . .	3
2.1.1.1	Months . . . . .	3
2.1.2	Variable Documentation . . . . .	4
2.1.2.1	DATE_SEPARATOR . . . . .	4
2.1.2.2	MAX_DAY . . . . .	4
2.1.2.3	MAX_MONTH . . . . .	4
2.1.2.4	MAX_YEAR . . . . .	4
2.1.2.5	MIN_DAY . . . . .	4
2.1.2.6	MIN_MONTH . . . . .	5
2.1.2.7	MIN_YEAR . . . . .	5
2.2	delta_date.h File Reference . . . . .	5
2.2.1	Function Documentation . . . . .	5
2.2.1.1	ask_and_compute_delta_day_between_two_dates() . . . . .	6
2.2.1.2	ask_date() . . . . .	6
2.2.1.3	ask_for_valid_date() . . . . .	6
2.2.1.4	ignore_date_separator() . . . . .	7
2.3	interface.h File Reference . . . . .	7
2.3.1	Function Documentation . . . . .	7
2.3.1.1	ask_for_restart() . . . . .	7

---

2.3.2	Variable Documentation . . . . .	7
2.3.2.1	RESTART_CHAR . . . . .	8
2.3.2.2	STOP_CHAR . . . . .	8
2.4	utilities.h File Reference . . . . .	8
2.4.1	Macro Definition Documentation . . . . .	8
2.4.1.1	CLEAR_BUFFER . . . . .	9
2.4.2	Function Documentation . . . . .	9
2.4.2.1	check_date_order() . . . . .	9
2.4.2.2	days_between_dates() . . . . .	9
2.4.2.3	is_date_valid() . . . . .	10
2.4.2.4	is_leap_year() . . . . .	10
	<b>Index</b>	<b>11</b>

# Chapter 1

## File Index

### 1.1 File List

Here is a list of all files with brief descriptions:

<a href="#">constants.h</a>	3
<a href="#">delta_date.h</a>	5
<a href="#">interface.h</a>	7
<a href="#">utilities.h</a>	8



# Chapter 2

## File Documentation

### 2.1 constants.h File Reference

#### Enumerations

- enum Months {  
JANUAR = 1, FEBRUAR, MARCH, APRIL,  
MAY, JUNE, JULY, AUGUST,  
SEPTEMBER, OCTOBER, NOVEMBER, DECEMBER }

#### Variables

- const int MIN\_DAY = 1
- const int MIN\_MONTH = JANUAR
- const int MIN\_YEAR = 1900
- const int MAX\_DAY = 31
- const int MAX\_MONTH = DECEMBER
- const int MAX\_YEAR = 2300
- const char DATE\_SEPARATOR = '-'

#### 2.1.1 Enumeration Type Documentation

##### 2.1.1.1 Months

enum Months

#### Enumerator

JANUAR	
FEBRUAR	
MARCH	
APRIL	
MAY	

**Enumerator**

JUNE	
JULY	
AUGUST	
SEPTEMBER	
OCTOBER	
NOVEMBER	
DECEMBER	

**2.1.2 Variable Documentation****2.1.2.1 DATE\_SEPARATOR**

```
const char DATE_SEPARATOR = '-'
```

**2.1.2.2 MAX\_DAY**

```
const int MAX_DAY = 31
```

**2.1.2.3 MAX\_MONTH**

```
const int MAX_MONTH = DECEMBER
```

**2.1.2.4 MAX\_YEAR**

```
const int MAX_YEAR = 2300
```

**2.1.2.5 MIN\_DAY**

```
const int MIN_DAY = 1
```



### 2.1.2.6 MIN\_MONTH

```
const int MIN_MONTH = JANUAR
```

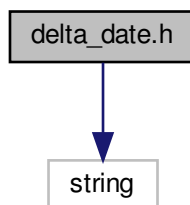
### 2.1.2.7 MIN\_YEAR

```
const int MIN_YEAR = 1900
```

## 2.2 delta\_date.h File Reference

```
#include <string>
```

Include dependency graph for delta\_date.h:



## Functions

- bool `ignore_date_separator ()`  
*This function clears the next character in the buffer, and returns true if it isn't DATE\_SEPARATOR.*
- bool `ask_date` (const std::string &date, int &day, int &month, int &year)  
*This function modifies the three referenced arguments to values inputted by the user.*
- void `ask_for_valid_date` (const std::string &date, int &day, int &month, int &year)  
*This function asks for a valid date with the format DD-MM-YYYY using ask\_date function. As long as the date is invalid, or the format is wrong, it asks for a new date.*
- void `ask_and_compute_delta_day_between_two_dates ()`  
*This function asks for a valid dates within a given range of dates in format DD-MM-YYYY and display the number of day between the choosen dates.*

### 2.2.1 Function Documentation

### 2.2.1.1 ask\_and\_compute\_delta\_day\_between\_two\_dates()

```
void ask_and_compute_delta_day_between_two_dates ( )
```

This function asks for a valid dates within a given range of dates in format DD-MM-YYYY and display the number of day between the choosen dates.

### 2.2.1.2 ask\_date()

```
bool ask_date (
    const std::string & date,
    int & day,
    int & month,
    int & year )
```

This function modifies the three referenced arguments to values inputted by the user.

#### Parameters

<i>date</i>	string telling the user if it's the start date or end date
<i>day</i>	input of the user
<i>month</i>	input of the user
<i>year</i>	input of the user

#### Returns

true if the date has been retrieve successfully according to DD-MM-YYYY format, else false

### 2.2.1.3 ask\_for\_valid\_date()

```
void ask_for_valid_date (
    const std::string & date,
    int & day,
    int & month,
    int & year )
```

This function asks for a valid date with the format DD-MM-YYYY using ask\_date function. As long as the date is invalid, or the format is wrong, it asks for a new date.

#### Parameters

<i>date</i>	string telling the user if it's the start date or end date
<i>day</i>	input of the user in function <a href="#">ask_date()</a>
<i>month</i>	input of the user in function <a href="#">ask_date()</a>
<i>year</i>	input of the user in function <a href="#">ask_date()</a>

#### 2.2.1.4 ignore\_date\_separator()

```
bool ignore_date_separator ( )
```

This function clears the next character in the buffer, and returns true if it isn't DATE\_SEPARATOR.

##### Returns

false if the buffer contains the char DATE\_SEPARATOR, true if it doesn't

## 2.3 interface.h File Reference

### Functions

- bool [ask\\_for\\_restart](#) ()

*This function keeps asking as long as the user enters anything else than RESTART\_CHAR or STOP\_CHAR.*

### Variables

- const char [RESTART\\_CHAR](#) = 'O'
- const char [STOP\\_CHAR](#) = 'N'

### 2.3.1 Function Documentation

#### 2.3.1.1 ask\_for\_restart()

```
bool ask_for_restart ( )
```

This function keeps asking as long as the user enters anything else than RESTART\_CHAR or STOP\_CHAR.

##### Returns

true if the user has entered RESTART\_CHAR and false if the user has entered STOP\_CHAR

### 2.3.2 Variable Documentation

### 2.3.2.1 RESTART\_CHAR

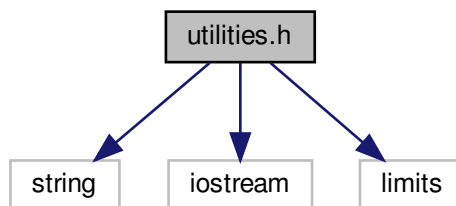
```
const char RESTART_CHAR = 'O'
```

### 2.3.2.2 STOP\_CHAR

```
const char STOP_CHAR = 'N'
```

## 2.4 utilities.h File Reference

```
#include <string>
#include <iostream>
#include <limits>
Include dependency graph for utilities.h:
```



### Macros

- `#define CLEAR_BUFFER` `std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n')`

### Functions

- `bool is_leap_year` (int year)
- `bool is_date_valid` (int day, int month, int year)
 

*This function checks that the date exists and is between a given range of dates.*
- `bool check_date_order` (int start\_day, int start\_month, int start\_year, int end\_day, int end\_month, int end\_year)
- `int days_between_dates` (int start\_day, int start\_month, int start\_year, int end\_day, int end\_month, int end\_year)
 

*This returns a negative number if the start date is after the end date.*

### 2.4.1 Macro Definition Documentation

### 2.4.1.1 CLEAR\_BUFFER

```
#define CLEAR_BUFFER std::cin.ignore(std::numeric_limits<std::streamsize>::max(),'\n')
```

## 2.4.2 Function Documentation

### 2.4.2.1 check\_date\_order()

```
bool check_date_order (
    int start_day,
    int start_month,
    int start_year,
    int end_day,
    int end_month,
    int end_year )
```

#### Parameters

<i>start_day</i>	day of the month (1 - 31 depending on the month)
<i>start_month</i>	a number between 1 and 12
<i>start_year</i>	a number between MIN_YEAR and MAX_YEAR
<i>end_day</i>	day of the month (1 - 31)
<i>end_month</i>	a number between 1 and 12
<i>end_year</i>	a number between MIN_YEAR and MAX_YEAR

#### Returns

true if the date composed of start\_day, start\_month and start\_year is before the date composed of end\_day, end\_month and end\_year

### 2.4.2.2 days\_between\_dates()

```
int days_between_dates (
    int start_day,
    int start_month,
    int start_year,
    int end_day,
    int end_month,
    int end_year )
```

This returns a negative number if the start date is after the end date.

#### Parameters

<i>startDay</i>	
-----------------	--

**Parameters**

<i>startMonth</i>	
<i>startYear</i>	
<i>endDay</i>	
<i>endMonth</i>	
<i>endYear</i>	

**Returns**

the number of days between the start date and the end date

**2.4.2.3 is\_date\_valid()**

```
bool is_date_valid (
    int day,
    int month,
    int year )
```

This function checks that the date exists and is between a given range of dates.

**Parameters**

<i>day</i>	any number
<i>month</i>	any number
<i>year</i>	any number

**Returns**

true if the date composed of day, month and year is a valid date

**2.4.2.4 is\_leap\_year()**

```
bool is_leap_year (
    int year )
```

**Parameters**

<i>year</i>	any given year
-------------	----------------

**Returns**

true if year is a leap year, else false

# Index

ask\_and\_compute\_delta\_day\_between\_two\_dates  
    delta\_date.h, 5

ask\_date  
    delta\_date.h, 6

ask\_for\_restart  
    interface.h, 7

ask\_for\_valid\_date  
    delta\_date.h, 6

CLEAR\_BUFFER  
    utilities.h, 8

check\_date\_order  
    utilities.h, 9

constants.h, 3

    DATE\_SEPARATOR, 4

    MAX\_DAY, 4

    MAX\_MONTH, 4

    MAX\_YEAR, 4

    MIN\_DAY, 4

    MIN\_MONTH, 4

    MIN\_YEAR, 5

    Months, 3

DATE\_SEPARATOR  
    constants.h, 4

days\_between\_dates  
    utilities.h, 9

delta\_date.h, 5

    ask\_and\_compute\_delta\_day\_between\_two\_↵  
        dates, 5

    ask\_date, 6

    ask\_for\_valid\_date, 6

    ignore\_date\_separator, 7

ignore\_date\_separator  
    delta\_date.h, 7

interface.h, 7

    ask\_for\_restart, 7

    RESTART\_CHAR, 7

    STOP\_CHAR, 8

is\_date\_valid  
    utilities.h, 10

is\_leap\_year  
    utilities.h, 10

MAX\_DAY  
    constants.h, 4

MAX\_MONTH  
    constants.h, 4

MAX\_YEAR  
    constants.h, 4

MIN\_DAY  
    constants.h, 4

MIN\_MONTH  
    constants.h, 4

MIN\_YEAR  
    constants.h, 5

Months  
    constants.h, 3

RESTART\_CHAR  
    interface.h, 7

STOP\_CHAR  
    interface.h, 8

utilities.h, 8

    CLEAR\_BUFFER, 8

    check\_date\_order, 9

    days\_between\_dates, 9

    is\_date\_valid, 10

    is\_leap\_year, 10