

IJC - DU1

Generated by Doxygen 1.8.17

1 Class Index	1
1.1 Class List	1
2 File Index	3
2.1 File List	3
3 Class Documentation	5
3.1 ppm Struct Reference	5
3.1.1 Detailed Description	5
3.1.2 Member Data Documentation	5
3.1.2.1 data	5
3.1.2.2 xsize	6
3.1.2.3 ysize	6
4 File Documentation	7
4.1 bitset.c File Reference	7
4.1.1 Detailed Description	7
4.2 bitset.h File Reference	7
4.2.1 Detailed Description	8
4.2.2 Macro Definition Documentation	8
4.2.2.1 bit_get	9
4.2.2.2 bit_set	9
4.2.2.3 bitset_alloc	10
4.2.2.4 bitset_create	11
4.2.2.5 bitset_free	11
4.2.2.6 bitset_getbit	12
4.2.2.7 bitset_setbit	13
4.2.2.8 bitset_size	13
4.2.3 Typedef Documentation	14
4.2.3.1 bitset_index_t	14
4.2.3.2 bitset_t	14
4.3 eratosthenes.c File Reference	14
4.3.1 Detailed Description	15
4.3.2 Function Documentation	15
4.3.2.1 Eratosthenes()	15
4.4 eratosthenes.h File Reference	15
4.4.1 Detailed Description	16
4.4.2 Function Documentation	16
4.4.2.1 Eratosthenes()	16
4.5 error.c File Reference	17
4.5.1 Detailed Description	17
4.5.2 Function Documentation	17
4.5.2.1 error_exit()	17

4.5.2.2 warning_msg()	18
4.6 error.h File Reference	18
4.6.1 Detailed Description	18
4.6.2 Function Documentation	19
4.6.2.1 error_exit()	19
4.6.2.2 warning_msg()	19
4.7 ppm.c File Reference	20
4.7.1 Detailed Description	20
4.7.2 Macro Definition Documentation	20
4.7.2.1 MAX_PPM_SIZE	21
4.7.2.2 MAX_PPM_SIZE_X	21
4.7.2.3 MAX_PPM_SIZE_Y	21
4.7.3 Function Documentation	21
4.7.3.1 ppm_free()	21
4.7.3.2 ppm_read()	21
4.8 ppm.h File Reference	22
4.8.1 Detailed Description	22
4.8.2 Function Documentation	23
4.8.2.1 ppm_free()	23
4.8.2.2 ppm_read()	23
4.9 primes.c File Reference	23
4.9.1 Detailed Description	24
4.9.2 Macro Definition Documentation	24
4.9.2.1 BITSET_SIZE	24
4.9.2.2 PRINT_LAST_NUMBERS	24
4.9.3 Function Documentation	25
4.9.3.1 main()	25
4.10 steg-decode.c File Reference	25
4.10.1 Detailed Description	25
4.10.2 Macro Definition Documentation	26
4.10.2.1 START_PRIME	26
4.10.3 Function Documentation	26
4.10.3.1 countPrimes()	26
4.10.3.2 main()	26

Index	29
--------------	-----------

Chapter 1

Class Index

1.1 Class List

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

ppm	Struktura pro PPM soubor	5
---------------------	------------------------------------	-------------------

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

bitset.c	Modul umoznujici externi pouziti inline funkci z ' bitset.h '	7
bitset.h	Definice maker a inline funkci pouzivanych v celem projektu	7
eratosthenes.c	Modul obsahujici implementaci Eratosthenova sita v C nad bitset polem	14
eratosthenes.h	Hlavickovy soubor k ' eratosthenes.c '	15
error.c	Modul s definicemi funkci pouzivanyymi k vypisu chybovych hlaseni	17
error.h	Hlavickovy soubor k modulu error.c	18
ppm.c	Modul definujici funkce pro zpracovani PPM souboru	20
ppm.h	Hlavickovy soubor modulu ppm.c	22
primes.c	'Hlavni' modul, který se stara o vytvoreni bitset pole, aplikace Eratosthenova sita na pole a nasledny vystup poslednich 10 prvocisel	23
steg-decode.c	'Hlavni' modul pro dekodovani zpravy z PPM souboru	25

Chapter 3

Class Documentation

3.1 ppm Struct Reference

Struktura pro PPM soubor.

```
#include <ppm.h>
```

Public Attributes

- unsigned [xsize](#)
- unsigned [ysize](#)
- char [data](#) []

3.1.1 Detailed Description

Struktura pro PPM soubor.

Spravny (akceptovany) format PPM souboru:

```
"P6" <ws>+  
<xsize<txt> <ws>+ <ysize<txt> <ws>+  
"255" <ws>  
<binární data, 3*xsize*ysize bajtů RGB>  
<EOF>
```

3.1.2 Member Data Documentation

3.1.2.1 data

```
char ppm::data[]
```

pole pro uložení binárních dat - obrazku

3.1.2.2 xsize

`unsigned ppm::xsize`

vyska obrazku (dat)

3.1.2.3 ysize

`unsigned ppm::ysize`

sirka obrazku (dat)

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

- [ppm.h](#)

Chapter 4

File Documentation

4.1 `bitset.c` File Reference

Modul umoznujici externi pouziti inline funkci z '[bitset.h](#)'.

```
#include "bitset.h"
```

4.1.1 Detailed Description

Modul umoznujici externi pouziti inline funkci z '[bitset.h](#)'.

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, priklad a) b)

Prelozeno na gcc 9.2.1 (20200130, Manjaro Linux)

4.2 `bitset.h` File Reference

Definice maker a inline funkci pouzivanych v celem projektu.

```
#include <assert.h>
#include <limits.h>
#include <stdlib.h>
#include "error.h"
```

Macros

- `#define bitset_create(jmeno_pole, velikost)`
Makro, ktore vytvori bitset pole o pozadovane velikosti v bitech.
- `#define bitset_alloc(jmeno_pole, velikost)`
Vytvori a alokuje bitset pole s pozadovanim nazvom a velikosti v bitech.
- `#define bit_set(number, index, value)`
Nastavi zadany bit cisla na pozadovanou hodnotu 0 nebo 1.
- `#define bit_get(number, index) (((bitset_index_t) index < sizeof(number)*CHAR_BIT) ? (number & (1 << (index%(sizeof(number)*CHAR_BIT)))) : (error_exit("bit_get: Index %lu mimo rozsah 0..%lu\n", (unsigned long) index, (unsigned long) sizeof(number)*CHAR_BIT), 0))`
Zjisti a "vrati" hodnotu bitu v cisle na pozadovane pozici.
- `#define bitset_free(jmeno_pole) free(jmeno_pole);`
Uvolni alokovane pole z pameti.
- `#define bitset_size(jmeno_pole) jmeno_pole[0]`
Zjisti velikost bitset pole.
- `#define bitset_getbit(jmeno_pole, index) (((bitset_index_t) index < (unsigned long) bitset_size(jmeno_pole)) ? jmeno_pole[1 + (index/(sizeof(*jmeno_pole)*CHAR_BIT))] & ((bitset_t) 1 << (index%(sizeof(*jmeno_pole)*CHAR_BIT))) : (error_exit("bitset_getbit: Index %lu mimo rozsah 0..%lu\n", (unsigned long) index, (unsigned long) bitset_size(jmeno_pole)), (unsigned long) 0))`
Zjisti hodnotu bitu na zadanem indexu.
- `#define bitset_setbit(jmeno_pole, index, vyraz)`
Nastavi hodnotu bitu na zadanem indexu.

Typedefs

- `typedef unsigned long bitset_index_t`
- `typedef unsigned long bitset_t`

4.2.1 Detailed Description

Definice maker a inline funkci pouzivanych v celem projektu.

Author

Dominik Horky, FIT

Date

14.03.2020

Note

Reseni IJC-DU1, priklad a) b)

Prelozeno na gcc 9.2.1 (20200130, Manjaro Linux)

4.2.2 Macro Definition Documentation

4.2.2.1 bit_get

```
#define bit_get(
    number,
    index ) (((bitset_index_t) index < sizeof(number)*CHAR_BIT) ? (number & (1 <<
(index%(sizeof(number)*CHAR_BIT))) ) : (error_exit("bit_get: Index %lu mimo rozsah 0..%lu\n",
(unsigned long) index, (unsigned long) sizeof(number)*CHAR_BIT), 0))
```

Zjistí a "vrátí" hodnotu bitu v čísle na požadované pozici.

Používane v modulu steg-decode.

Parameters

<i>number</i>	číslo
<i>index</i>	bit, jehož hodnotu chci zjistit

Postcondition

"Vrátí" hodnotu bitu na pozici 'index' čísla 'number'.

Note

Pokud je index mimo rozsah čísla, vypíše chybové hlášení.

See also

[steg-decode.c](#)

4.2.2.2 bit_set

```
#define bit_set(
    number,
    index,
    value )
```

Value:

```
if ((bitset_index_t) index < sizeof(number)*CHAR_BIT) { \
    if (value) \
        number[1 + (index/(sizeof(number[0])*CHAR_BIT))] |= (unsigned long) (1UL <<
(index%(sizeof(number[0])*CHAR_BIT))); \
    else \
        number[1 + (index/(sizeof(number[0])*CHAR_BIT))] &= (unsigned long) ~(1UL <<
(index%(sizeof(number[0])*CHAR_BIT))); } \
else \
    error_exit("bit_set: Index %lu mimo rozsah 0..%lu\n", (unsigned long) index, (unsigned long)
(sizeof(number)*CHAR_BIT));
```

Nastaví zadany bit čísla na požadovanou hodnotu 0 nebo 1.

Použito při dekodování PPM obrázku v modulu steg-decode.

Parameters

<i>number</i>	cislo, kteremu se nastavi/zmeni pozadovany bit
<i>index</i>	cozice bitu, který bude nastaven
<i>value</i>	nova hodnota bitu na indexu

Postcondition

Cislo 'number' bude mit na pozici 'index' nastavenou hodnotu 'value' - 0 nebo 1.

Note

Pokud bude index mimo rozsah, vypise chybovou hlasku.

See also

[steg-decode.c](#)

4.2.2.3 bitset_alloc

```
#define bitset_alloc(  
    jmeno_pole,  
    velikost )
```

Value:

```
    assert(velikost > 0); \  
    bitset_t *jmeno_pole = (bitset_t *)calloc(1 + (velikost/(sizeof(bitset_t)*CHAR_BIT)) +  
    ((velikost%(sizeof(bitset_t)*CHAR_BIT)) > 0 ? 1 : 0), CHAR_BIT); \  
    if (jmeno_pole == NULL) \  
        error_exit("bitset_alloc: Chyba alokace paměti\n", velikost); \  
    else \  
        jmeno_pole[0] = velikost;
```

Vytvori a alokuje bitset pole s pozadovany nazvem a velikosti v bitech.

Dale take je pole vynulovano a na prvni indexu je udaj o velikosti pole. (pole bude dynamicke)

Parameters

<i>jmeno_pole</i>	nazev pole, které bude vytvořeno
<i>velikost</i>	velikost pole v bitech

Precondition

Ocekava dostatek volne pameti (RAM) pro alokaci pole.

Postcondition

Bude vytvořeno (a alokováno) pole s požadovaným názvem a velikostí.

Note

Pri spatne velikosti pole nebo chybne alokaci vypise chybovou hlasku a ukonci program.

4.2.2.4 bitset_create

```
#define bitset_create(  
    jmeno_pole,  
    velikost )
```

Value:

```
static_assert((velikost > 0), "Pole nesmi mit nulovou nebo zapornou velikost!"); \  
bitset_t jmeno_pole[1 + (velikost/(sizeof(bitset_t)*CHAR_BIT)) +  
((velikost%(sizeof(bitset_t)*CHAR_BIT)) > 0 ? 1 : 0)] = { velikost };
```

Makro, které vytvoří bitset pole o požadované velikosti v bitech.

Pole je automaticky nulované a na prvním indexu je uložen údaj o jeho velikosti. (pole bude automaticke/lokalni)

Parameters

<i>jmeno_pole</i>	nazev pole, které bude vytvořeno
<i>velikost</i>	požadovaná velikost pole v bitech

Postcondition

Bude vytvořeno pole s požadovaným názvem a velikostí.

Note

Pri spatne velikosti pole vypise static_assert - chybove hlaseni pri prekladu.

Pokud bude pole prilis velke (jako např. pro tento projekt), je nutné navýsit kapacitu zásobníku!

4.2.2.5 bitset_free

```
#define bitset_free(  
    jmeno_pole ) free(jmeno_pole);
```

Uvolní alokované pole z paměti.

Parameters

<i>jmeno_pole</i>	nazev pole, které má být uvolněno
-------------------	-----------------------------------

Precondition

Ocekava existujici a alokovane pole.

Postcondition

Uvolnene pole z pameti.

Note

Pri vytvareni 'primes-i' se pouzije inline funkce shodneho nazvu a parametru (vraci void).

4.2.2.6 bitset_getbit

```
#define bitset_getbit(  
    jmeno_pole,  
    index ) (((bitset_index_t) index < (unsigned long) bitset_size(jmeno_pole)) ?  
jmeno_pole[1 + (index/(sizeof(*jmeno_pole)*CHAR_BIT))] & ((bitset_t) 1 << (index%(sizeof(*jmeno_pole)*CHAR_BIT))) : (error_exit("bitset_getbit: Index %lu mimo rozsah 0..%lu\n", (unsigned long) index, (unsigned long) bitset_size(jmeno_pole)), (unsigned long) 0))
```

Zjisti hodnotu bitu na zadanem indexu.

Parameters

<i>jmeno_pole</i>	nazev pole
<i>index</i>	index v bitset poli

Precondition

Ocekava existujici pole a index v rozsahu velikosti pole.

Postcondition

"Vrati" hodnotu bitu.

Note

Pokud je index mimo rozsah pole, vypise chybovou hlasku.

Pri vytvareni 'primes-i' se pouzije inline funkce shodneho nazvu a parametru (vraci bitset_t).

4.2.2.7 bitset_setbit

```
#define bitset_setbit(  
    jmeno_pole,  
    index,  
    vyraz )
```

Value:

```
if ((bitset_index_t) index < bitset_size(jmeno_pole)) { \
    if (vyraz) \
        jmeno_pole[1 + (index/(sizeof(*jmeno_pole)*CHAR_BIT))] |= ((bitset_t) 1 << \
        (index%(sizeof(*jmeno_pole)*CHAR_BIT))); \
    else \
        jmeno_pole[1 + (index/(sizeof(*jmeno_pole)*CHAR_BIT))] &= ~((bitset_t) 1 << \
        (index%(sizeof(*jmeno_pole)*CHAR_BIT))); \
    } \
    else \
        error_exit("bitset_setbit: Index %lu mimo rozsah 0..%lu\n", (unsigned long) index, (unsigned \
        long) bitset_size(jmeno_pole));
```

Nastavi hodnoti bitu na zadanem indexu.

Parameters

<i>jmeno_pole</i>	nazev pole
<i>index</i>	pozice-index v poli
<i>vyraz</i>	hodnota, která ma byt nastavena

Precondition

Ocekava existujici pole a index v rozsahu pole.

Postcondition

Nastavi hodnotu bitu na pozadovanou hodnotu (0 nebo 1).

Note

Pokud je index mimo rozsah pole, vypise chybove hlaseni.

Pri vytvoreni 'primes-i' se pouzije inline funkce shodneho nazvu a parametru (vraci void).

4.2.2.8 bitset_size

```
#define bitset_size(  
    jmeno_pole ) jmeno_pole[0]
```

Zjisti velikost bitset pole.

Parameters

<i>jmeno_pole</i>	nazev pole
-------------------	------------

Precondition

Ocekava existující pole

Postcondition

"Vrátí" velikost pole v bitech (je uložena na prvním indexu).

Note

Při vytváření 'primes-i' se použije inline funkce shodného názvu a parametru (vrací `bitset_t`).

4.2.3 Typedef Documentation

4.2.3.1 `bitset_index_t`

```
typedef unsigned long bitset_index_t
```

typ používaný pro indexy v `bitset` poli

4.2.3.2 `bitset_t`

```
typedef unsigned long bitset_t
```

datový typ `bitset` pole

4.3 `eratosthenes.c` File Reference

Modul obsahující implementaci Eratosthenova síta v C nad `bitset` polem.

```
#include "eratosthenes.h"
```

Functions

- void `Eratosthenes` (`bitset_t` *pole)

Implementace Eratosthenova síta nad `bitset` polem.

4.3.1 Detailed Description

Modul obsahující implementaci Eratosthenova síta v C nad bitset polem.

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, příklad a)

Preloženo na gcc 9.2.1 (20200130, Manjaro Linux)

4.3.2 Function Documentation

4.3.2.1 Eratosthenes()

```
void Eratosthenes (
    bitset_t * pole )
```

Implementace Eratosthenova síta nad bitset polem.

Hodnoty na indexech, které nejsou prvočíslem budou nastaveny na 1, ostatní na 0.

Parameters

<i>pole</i>	Pole, nad kterým se vykoná algoritmus
-------------	---------------------------------------

Precondition

Očekává existující a vynulované bitset pole.

Postcondition

Pole na indexech, které jsou prvočíslo mají hodnotu 0, ostatní 1.

4.4 eratosthenes.h File Reference

Hlavičkový soubor k '[eratosthenes.c](#)'.

```
#include <math.h>
#include "bitset.h"
```

Functions

- void `Eratosthenes` (`bitset_t` *pole)
Implementace Eratostenova sita nad bitset polem.

4.4.1 Detailed Description

Hlavickovy soubor k '`eratosthenes.c`'.

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, priklad a)

Preloženo na gcc 9.2.1 (20200130, Manjaro Linux)

4.4.2 Function Documentation

4.4.2.1 Eratosthenes()

```
void Eratosthenes (  
    bitset_t * pole )
```

Implementace Eratostenova sita nad bitset polem.

Hodnoty na indexech, které nejsou prvočíslem budou nastaveny na 1, ostatní na 0.

Parameters

<code>pole</code>	Pole, nad kterým se vykona algoritmus
-------------------	---------------------------------------

Precondition

Očekává existující a vynulované bitset pole.

Postcondition

Pole na indexech, které jsou prvočíslo mají hodnotu 0, ostatní 1.

4.5 error.c File Reference

Modul s definicemi funkci pouzivanymi k vypisu chybovych hlaseni.

```
#include <stdio.h>
#include <stdarg.h>
#include <stdlib.h>
#include "error.h"
```

Functions

- void [warning_msg](#) (const char *fmt,...)
Vypise chybove hlaseni na stderr.
- void [error_exit](#) (const char *fmt,...)
Vypise chybove hlaseni na stderr a ukonci program.

4.5.1 Detailed Description

Modul s definicemi funkci pouzivanymi k vypisu chybovych hlaseni.

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, priklad b)

Prelozeno na gcc 9.2.1 (20200130, Manjaro Linux)

4.5.2 Function Documentation

4.5.2.1 error_exit()

```
void error_exit (
    const char * fmt,
    ... )
```

Vypise chybove hlaseni na stderr a ukonci program.

Chova se jako printf.

Parameters

<i>fmt</i>	fixni zacatek kazdeho chyboveho hlaseni
...	dalsi argumenty - dodatecny text, int, char aj. hodnoty, ..

Postcondition

Chybove hlaseni se vypise na stderr a program bude ukoncen s hodnotou 1.

4.5.2.2 warning_msg()

```
void warning_msg (
    const char * fmt,
    ... )
```

Vypise chybove hlaseni na stderr.

Chova se jako printf.

Parameters

<i>fmt</i>	fixni zacatek kazdeho chyboveho hlaseni
...	dalsi argumenty - dodatecny text, int, char aj. hodnoty, ..

4.6 error.h File Reference

Hlavickovy soubor k modulu [error.c](#).

Functions

- void [warning_msg](#) (const char **fmt*,...)
Vypise chybove hlaseni na stderr.
- void [error_exit](#) (const char **fmt*,...)
Vypise chybove hlaseni na stderr a ukonci program.

4.6.1 Detailed Description

Hlavickovy soubor k modulu [error.c](#).

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, priklad b)

Prelozeno na gcc 9.2.1 (20200130, Manjaro Linux)

4.6.2 Function Documentation

4.6.2.1 error_exit()

```
void error_exit (
    const char * fmt,
    ... )
```

Vypise chybove hlasi na stderr a ukonci program.

Chova se jako printf.

Parameters

<i>fmt</i>	fixni zacatek kazdeho chyboveho hlasi
...	dalsi argumenty - dodatecny text, int, char aj. hodnoty, ..

Postcondition

Chybove hlasi se vypise na stderr a program bude ukoncen s hodnotou 1.

4.6.2.2 warning_msg()

```
void warning_msg (
    const char * fmt,
    ... )
```

Vypise chybove hlasi na stderr.

Chova se jako printf.

Parameters

<i>fmt</i>	fixni zacatek kazdeho chyboveho hlasi
...	dalsi argumenty - dodatecny text, int, char aj. hodnoty, ..

4.7 ppm.c File Reference

Modul definující funkce pro zpracování PPM souboru.

```
#include "ppm.h"
#include <stdio.h>
#include <errno.h>
```

Macros

- `#define MAX_PPM_SIZE_X 8000`
- `#define MAX_PPM_SIZE_Y 8000`
- `#define MAX_PPM_SIZE MAX_PPM_SIZE_X*MAX_PPM_SIZE_Y*3`

Functions

- `struct ppm * ppm_read (const char *filename)`
Přečte PPM soubor, zkontroluje správnost dat a nahraje data do struktury.
- `void ppm_free (struct ppm *p)`
Uvolní strukturu z paměti.

4.7.1 Detailed Description

Modul definující funkce pro zpracování PPM souboru.

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, příklad b)

Preloženo na gcc 9.2.1 (20200130, Manjaro Linux)

4.7.2 Macro Definition Documentation

4.7.2.1 MAX_PPM_SIZE

```
#define MAX_PPM_SIZE MAX_PPM_SIZE_X*MAX_PPM_SIZE_Y*3
```

implementacni limit - maximalni velikost obrazku (objem dat)

4.7.2.2 MAX_PPM_SIZE_X

```
#define MAX_PPM_SIZE_X 8000
```

maximalni vyska obrazku

4.7.2.3 MAX_PPM_SIZE_Y

```
#define MAX_PPM_SIZE_Y 8000
```

maximalni sirka obrazku

4.7.3 Function Documentation

4.7.3.1 ppm_free()

```
void ppm_free (
    struct ppm * p )
```

Uvolni strukturu z pameti.

Parameters

<i>p</i>	nazev struktury
----------	-----------------

Postcondition

Struktura bude uvolnena z pameti.

4.7.3.2 ppm_read()

```
struct ppm* ppm_read (
    const char * filename )
```

Precte PPM soubor, zkontroluje spravnost dat a nahraje data do struktury.

Parameters

<i>filename</i>	nazev souboru (*.ppm)
-----------------	-----------------------

Postcondition

Bude pripravena struktura s validnimi daty pro dekodovani.

Returns

Vrati strukturu s nahranyimi daty a udaji o velikosti nebo NULL v pripade chyby pri otevreni souboru.

4.8 ppm.h File Reference

Hlavickovy soubor modulu [ppm.c](#).

```
#include "error.h"
#include <string.h>
#include <ctype.h>
#include <stdlib.h>
```

Classes

- struct [ppm](#)
Struktura pro PPM soubor.

Functions

- struct [ppm](#) * [ppm_read](#) (const char *filename)
Precte PPM soubor, zkontroluje spravnost dat a nahraje data do struktury.
- void [ppm_free](#) (struct [ppm](#) *p)
Uvolni strukturu z pameti.

4.8.1 Detailed Description

Hlavickovy soubor modulu [ppm.c](#).

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, priklad b)

Preloženo na gcc 9.2.1 (20200130, Manjaro Linux)

4.8.2 Function Documentation

4.8.2.1 ppm_free()

```
void ppm_free (
    struct ppm * p )
```

Uvolni strukturu z pameti.

Parameters

<i>p</i>	nazev struktury
----------	-----------------

Postcondition

Struktura bude uvolнена z pameti.

4.8.2.2 ppm_read()

```
struct ppm* ppm_read (
    const char * filename )
```

Precte PPM soubor, zkontroluje spravnost dat a nahraje data do struktury.

Parameters

<i>filename</i>	nazev souboru (*.ppm)
-----------------	-----------------------

Postcondition

Bude pripravena struktura s validnimi daty pro dekodovani.

Returns

Vrati strukturu s nahranyimi daty a udaji o velikosti nebo NULL v pripade chyby pri otevreni souboru.

4.9 primes.c File Reference

'Hlavni' modul, ktery se stara o vytvoreni bitset pole, aplikace Eratosthenova sita na pole a nasledny vystup poslednich 10 prvocisel.

```
#include <time.h>
#include <stdio.h>
#include "eratosthenes.h"
```

Macros

- `#define BITSET_SIZE 5*100*1000*1000`
- `#define PRINT_LAST_NUMBERS 10`

Functions

- `int main (void)`

Main zajistuje vytvoreni pole a vystup pozadovaneho poctu prvocisel na stdout.

4.9.1 Detailed Description

'Hlavni' modul, který se stara o vytvoreni bitset pole, aplikace Eratosthenova sita na pole a nasledny vystup poslednich 10 prvocisel.

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, priklad a)

Preloženo na gcc 9.2.1 (20200130, Manjaro Linux)

4.9.2 Macro Definition Documentation

4.9.2.1 BITSET_SIZE

```
#define BITSET_SIZE 5*100*1000*1000
```

maximalni velikost bitset pole a zaroven hranice, pokud se pocitaji prvocisla

4.9.2.2 PRINT_LAST_NUMBERS

```
#define PRINT_LAST_NUMBERS 10
```

pocet poslednich prvocisel, které se vytisknou na stdout (od konce)

4.9.3 Function Documentation

4.9.3.1 main()

```
int main (
    void )
```

Main zajistuje vytvoreni pole a vystup pozadovaneho poctu prvocisel na stdout.

pokud bude definovano, vypise se pred ukoncenim programu cas algoritmu pokud bude definovano, vytvori a pouzije se dynamicke (bitset) pole misto lokalniho

Returns

Vraci 0 pokud vse probehlo v poradku nebo nenulovou hodnotu pokud nastala chyba.

4.10 steg-decode.c File Reference

'Hlavni' modul pro dekodovani zpravy z PPM souboru.

```
#include "ppm.h"
#include "eratosthenes.h"
#include <stdio.h>
```

Macros

- #define `START_PRIME` 23

Functions

- int `countPrimes` (`bitset_t` *array)
Vypocita pocet vseh prvocisel v bitset poli.
- int `main` (int argc, char **argv)
Main zde zajistuje vytvoreni pole, dekodovani a vystup zpravy na stdout.

4.10.1 Detailed Description

'Hlavni' modul pro dekodovani zpravy z PPM souboru.

Author

Dominik Horky, FIT

Date

13.03.2020

Note

Reseni IJC-DU1, priklad b)

Prelozeno na gcc 9.2.1 (20200130, Manjaro Linux)

4.10.2 Macro Definition Documentation

4.10.2.1 START_PRIME

```
#define START_PRIME 23
```

pocatecni prvocislo, od ktereho zacina zprava (viz. zadani)

4.10.3 Function Documentation

4.10.3.1 countPrimes()

```
int countPrimes (  
    bitset_t * array )
```

Vypocita pocet vseh prvocisel v bitset poli.

Parameters

<i>array</i>	bitset pole
--------------	-------------

Precondition

Ocekava bitset pole, na kterem bylo provedeno Eratosthenovo sito.

Returns

Pocet prvocisel v poli.

4.10.3.2 main()

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

Main zde zajistuje vytvoreni pole, dekodovani a vystup zpravy na stdout.

Parameters

<i>argc</i>	pocet argumentu
<i>argv</i>	argumenty programu - program akceptuje jediny argument -> nazev *.ppm souboru pro dekodovani

Returns

Vraci 0 pokud vse probehlo v poradku nebo nenulovou hodnotu pokud nastala chyba.

Index

- bit_get
 - bitset.h, 8
- bit_set
 - bitset.h, 9
- bitset.c, 7
- bitset.h, 7
 - bit_get, 8
 - bit_set, 9
 - bitset_alloc, 10
 - bitset_create, 11
 - bitset_free, 11
 - bitset_getbit, 12
 - bitset_index_t, 14
 - bitset_setbit, 12
 - bitset_size, 13
 - bitset_t, 14
- bitset_alloc
 - bitset.h, 10
- bitset_create
 - bitset.h, 11
- bitset_free
 - bitset.h, 11
- bitset_getbit
 - bitset.h, 12
- bitset_index_t
 - bitset.h, 14
- bitset_setbit
 - bitset.h, 12
- BITSET_SIZE
 - primes.c, 24
- bitset_size
 - bitset.h, 13
- bitset_t
 - bitset.h, 14
- countPrimes
 - steg-decode.c, 26
- data
 - ppm, 5
- Eratosthenes
 - eratosthenes.c, 15
 - eratosthenes.h, 16
- eratosthenes.c, 14
 - Eratosthenes, 15
- eratosthenes.h, 15
 - Eratosthenes, 16
- error.c, 17
 - error_exit, 17
 - warning_msg, 18
- error.h, 18
 - error_exit, 19
 - warning_msg, 19
- error_exit
 - error.c, 17
 - error.h, 19
- main
 - primes.c, 25
 - steg-decode.c, 26
- MAX_PPM_SIZE
 - ppm.c, 20
- MAX_PPM_SIZE_X
 - ppm.c, 21
- MAX_PPM_SIZE_Y
 - ppm.c, 21
- ppm, 5
 - data, 5
 - xsize, 5
 - ysize, 6
- ppm.c, 20
 - MAX_PPM_SIZE, 20
 - MAX_PPM_SIZE_X, 21
 - MAX_PPM_SIZE_Y, 21
 - ppm_free, 21
 - ppm_read, 21
- ppm.h, 22
 - ppm_free, 23
 - ppm_read, 23
- ppm_free
 - ppm.c, 21
 - ppm.h, 23
- ppm_read
 - ppm.c, 21
 - ppm.h, 23
- primes.c, 23
 - BITSET_SIZE, 24
 - main, 25
 - PRINT_LAST_NUMBERS, 24
- PRINT_LAST_NUMBERS
 - primes.c, 24
- START_PRIME
 - steg-decode.c, 26
- steg-decode.c, 25
 - countPrimes, 26
 - main, 26
 - START_PRIME, 26

warning_msg
 error.c, [18](#)
 error.h, [19](#)

xsize
 ppm, [5](#)

ysize
 ppm, [6](#)