Matice

Termín odevzdání:	**07.04.2013 23:59:59**
Pozdní odevzdání s penalizací:	**12.05.2013 23:59:59** (Penále za pozdní odevzdání: 100.0000 %)
Hodnocení:	**4.4000**
Max. hodnocení:	**4.0000** (bez bonusů)
Odevzdaná řešení:	4 / 20 Volné pokusy + 20 Penalizované pokusy (-2 % penalizace za každé odevzdání)
Nápovědy:	0 / 2 Volné nápovědy + 2 Penalizované nápovědy (-10 % penalizace za každou nápovědu)

Úkolem je realizovat třídu, která bude reprezentovat matici čísel.

Třída `CMatrix` reprezentuje matici desetinných čísel. S maticí předpokládáme běžné operace:

konstruktor
: vytvoří matici zadané velikosti, vyplněnou hodnotami nula,

kopírující konstruktor
: který vytvoří hlubokou kopii zadané matice (implementujte jej, pokud automaticky generovaný kopírující konstruktor nevyhovuje),

destruktor
: uvolní prostředky alokované instancí (implementujte jej, pokud automaticky generovaný destruktor nevyhovuje),

operátor =
: provede zkopírování matice (implementujte jej, pokud automaticky generovaný operátor = nevyhovuje)

operátor binární +
: sečte dvě matice. Operátor zkontroluje velikosti matic, pokud se neshodují, operátor vygeneruje výjimku `CSizeException` (více níže),

operátor unární -
: vytvoří matici s opačnými hodnotami znamének,

operátor binární -
: odečte dvě matice. Operátor zkontroluje velikosti matic, pokud se neshodují, operátor vygeneruje výjimku `CSizeException` (více níže),

operátor binární *
: vynásobí dvě matice. Operátor zkontroluje velikosti matic, pokud velikosti matic neumožňují násobení, operátor vygeneruje výjimku `CSizeException` (více níže),

operátor binární *
: vynásobí hodnoty v matici zadaným desetinným číslem,

operátor []
: umožní přístup (čtení/zápis) čísel v matici se zadanými indexy. Pokud Jsou zadané indexy mimo meze, operátor vyhodí výjimku `CIndexException`, pozor, musí pracovat i s konstantními maticemi,

operátor <<
: zobrazí matici do výstupního proudu. Formát odpovídá inicializátoru 2D pole v C/C++ syntaxi, číselné hodnoty neformátujte (ponechte implicitní formátování desetinných čísel),

operátor >>
: načte matici ze vstupního proudu. Formát vstupu je stejný jako u výstupu, načítání dále musí přeskakovat bílé znaky. Pokud se načtení nezdaří, bude operátor fungovat jako standardní operátory >>, tedy nastaví příznak fail bit ve vstupním proudu (`is . setstate (ios::failbit)`) a ponechá pravý operand beze změny.

Implementace vyžaduje nejméně dvě podpůrné třídy implementující výjimky:

- `CSizeException` výjimka bude vyhozena pokud pro dané velikosti matic (operandy) nelze operaci provést (např. sečíst, vynásobit). Třída implementující výjimku musí umět zobrazení do výstupního proudu pomocí operátoru <<. Výsledkem zobrazení je řetězec popisující okolnosti vzniku výjimky např.:

  ```
  Invalid matrix size 2x3 * 2x3
  ```

  (Uvozovky v ukázkách níže jsou nejsou součástí výpisu výjimky, jsou uvedené pouze jako jasné ohraničení zobrazovaného textu. Z toho je mj. vidět, že odřádkování není součástí výpisu.)
- `CIndexException` je výjimka házená při nesprávném použití operátoru hranaté závorky. Výjimka zobrazuje jiný text, ale jinak se chová stejně jako předešlá výjimka.
- Při házení výjimek nevytvářejte objekt výjimky dynamicky. Použijte způsob běžný v C++ - házejte dočasný objekt, např:

  ```
  throw CIndexException ( ...parametry... ); // zde neni  new
  ```

```
...

try
 {
   // code
 }
catch ( const CIndexException & e )
 {
   cout << e;
 }
```

Odevzdávejte zdrojový soubor, který obsahuje Vaší implementaci třídy `CMatrix` a podpůrných tříd (výjimky, ...). V odevzdávaném souboru nenechávejte vkládání hlavičkových souborů, Vaše testovací funkce a funkci `main`. Pokud v souboru chcete ponechat `main` nebo vkládání hlavičkových souborů, vložte je do bloku podmíněného překladu. V testovacím prostředí není k dispozici STL - pokud použijete např. `vector`, program nepůjde zkompilovat (je potřeba alokovat matici s dopředu známými rozměry, "natahovací" pole je zbytečné).

V tomto příkladu není poskytnutý předpis pro požadované rozhraní třídy. Z textového popisu, ukázky použití níže a znalostí přetěžování operátorů byste měli být schopni toto rozhraní vymyslet.

Ukázka použití třídy:

```
#ifndef __PROGTEST__
#include <iostream>
#include <sstream>
#include <iomanip>
using namespace std;
#endif /* __PROGTEST__ */

istringstream is;
ostringstream os;
double x;

CMatrix a ( 2, 3 );
a[0][0] = 1;
a[0][1] = 2;
a[0][2] = 3;
a[1][0] = 4;
a[1][1] = 5;
a[1][2] = 6;
const CMatrix b = a;
CMatrix c ( 3, 2 );
c[0][0] = 1;
c[0][1] = 1;
c[1][0] = 2;
c[1][1] = -2;
c[2][0] = 3;
c[2][1] = 3;
os . str ("");
os << a;
/*
--8<----8<----8<----8<----8<--
{
  {1, 2, 3},
  {4, 5, 6}
}
--8<----8<----8<----8<----8<--
*/
os . str ("");
os << b;
/*
--8<----8<----8<----8<----8<--
{
  {1, 2, 3},
  {4, 5, 6}
}
--8<----8<----8<----8<----8<--
*/
os . str ("");
os << c;
```

```
/*
--8<----8<----8<----8<----8<--
{
  {1, 1},
  {2, -2},
  {3, 3}
}
--8<----8<----8<----8<----8<--
*/
CMatrix d ( 1, 1 );
d[0][0] = -1;
os . str ("");
os << d;
/*
--8<----8<----8<----8<----8<--
{
  {-1}
}
--8<----8<----8<----8<----8<--
*/
d = a + b;
os . str ("");
os << d;
/*
--8<----8<----8<----8<----8<--
{
  {2, 4, 6},
  {8, 10, 12}
}
--8<----8<----8<----8<----8<--
*/
d = a - b;
os . str ("");
os << d;
/*
--8<----8<----8<----8<----8<--
{
  {0, 0, 0},
  {0, 0, 0}
}
--8<----8<----8<----8<----8<--
*/
d = - a;
os . str ("");
os << d;
/*
--8<----8<----8<----8<----8<--
{
  {-1, -2, -3},
  {-4, -5, -6}
}
--8<----8<----8<----8<----8<--
*/
d = a * 2;
os . str ("");
os << d;
/*
--8<----8<----8<----8<----8<--
{
  {2, 4, 6},
  {8, 10, 12}
}
--8<----8<----8<----8<----8<--
*/
d = b * c;
os . str ("");
os << d;
/*
--8<----8<----8<----8<----8<--
```

```
{
  {14, 6},
  {32, 12}
}
--8<----8<----8<----8<----8<--
*/
d = a * b; // CSizeException thrown, text: "Invalid matrix size 2x3 * 2x3"
d = b + c; // CSizeException thrown, text: "Invalid matrix size 2x3 + 3x2"
x = b[0][0]; // x = 1.000000
x = b[0][3]; // CIndexException thrown, text: "Invalid index [0][3]"
x = b[2][0]; // CIndexException thrown, text: "Invalid index [2][0]"
is . clear ();
is . str ( "{ { 1, 2, 3, 4 }, { 4, 3, 2, 1 } } " );
is >> a; // is . fail () = false
d = a;
os . str ("");
os << a;
/*
--8<----8<----8<----8<----8<--
{
  {1, 2, 3, 4},
  {4, 3, 2, 1}
}
--8<----8<----8<----8<----8<--
*/
is . clear ();
is . str ( "{ { 1, 2, 3 } { 3, 4, 5} } " );
is >> c; // is . fail () = true
```

Nápověda

- Testovací prostředí kontroluje hodnoty ve Vašich objektech tím, že si je zpřístupní pomocí operátoru []. Dokud Vám nebude správně fungovat tento operátor, budou všechny testy negativní.
- Operátor pro výstup implementujte správně -- neposílejte data na `cout`, posílejte je do předaného výstupního proudu. Dodržte formát výstupu, posílejte čísla matice do výstupního proudu beze změn formátování (ponechte implicitní formátování des. čísel). Přesné rozmístění bílých znaků ve výstupu nehraje roli, porovnávání přeskakuje bílé znaky. Operátor pro vstup musí být schopen načíst matici, při načítání přeskakujte bílé znaky. Kontrolujte přítomnost závorek, čárek a shodný počet čísel na řádce matice.
- Pokud Vám program nejde zkompilovat, ujistěte se, že máte správně přetížené operátory. Zejména si zkontrolujte kvalifikátory `const`.

Referenční řešení

4	**25.03.2013 21:29:21**	**Download**
Stav odevzdání:	Ohodnoceno	
Hodnocení:	4.4000	

- **Hodnotitel: automat**
  - Program zkompilován
  - Test 'Zakladni test s parametry podle ukazky': Úspěch
    - Dosaženo: 100.00 %, požadováno: 100.00 %
    - Celková doba běhu: 0.000 s (limit: 2.000 s)
    - Využití paměti: 12580 KiB (limit: 17267 KiB)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Test I/O (op >> a op <<)': Úspěch
    - Dosaženo: 100.00 %, požadováno: 50.00 %
    - Celková doba běhu: 0.000 s (limit: 2.000 s)
    - Využití paměti: 12580 KiB (limit: 17267 KiB)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Test nahodnymi vstupy': Úspěch
    - Dosaženo: 100.00 %, požadováno: 50.00 %
    - Celková doba běhu: 0.003 s (limit: 2.000 s)
    - Využití paměti: 12580 KiB (limit: 17267 KiB)
    - Úspěch v závazném testu, hodnocení: 100.00 %
  - Test 'Test nahodnymi vstupy + kontrola prace s pameti': Úspěch

        - Dosaženo: 100.00 %, požadováno: 50.00 %
        - Celková doba běhu: 0.423 s (limit: 2.000 s)
        - Využití paměti: 19632 KiB (limit: 29366 KiB)
        - Úspěch v závazném testu, hodnocení: 100.00 %
    - Všechny paměťové bloky byly uvolněné - ok.
    - Celkové hodnocení: 100.00 % (= 1.00 * 1.00 * 1.00 * 1.00)
- Celkové procentní hodnocení: 100.00 %
- Bonus za včasné odevzdání: 0.40
- Celkem bodů: 1.00 * (4.00 + 0.40) = 4.40

		Celkem	Průměr	Maximum	Jméno funkce
SW metriky:	Funkce:	**28**	--	--	--
	Řádek kódu:	**314**	**11.21 ± 15.89**	**90**	`operator >>(istream &,CMatrix &)`
	Cyklomatická složitost:	**91**	**3.25 ± 5.56**	**29**	`operator >>(istream &,CMatrix &)`

3	**25.03.2013 21:28:03**	**Download**
Stav odevzdání:	Ohodnoceno	
Hodnocení:	4.4000	

- **Hodnotitel: automat**
    - Program zkompilován
    - Test 'Zakladni test s parametry podle ukazky': Úspěch
        - Dosaženo: 100.00 %, požadováno: 100.00 %
        - Celková doba běhu: 0.000 s (limit: 2.000 s)
        - Využití paměti: 12580 KiB (limit: 17267 KiB)
        - Úspěch v závazném testu, hodnocení: 100.00 %
    - Test 'Test I/O (op >> a op <<)': Úspěch
        - Dosaženo: 100.00 %, požadováno: 50.00 %
        - Celková doba běhu: 0.000 s (limit: 2.000 s)
        - Využití paměti: 12580 KiB (limit: 17267 KiB)
        - Úspěch v závazném testu, hodnocení: 100.00 %
    - Test 'Test nahodnymi vstupy': Úspěch
        - Dosaženo: 100.00 %, požadováno: 50.00 %
        - Celková doba běhu: 0.003 s (limit: 2.000 s)
        - Využití paměti: 12580 KiB (limit: 17267 KiB)
        - Úspěch v závazném testu, hodnocení: 100.00 %
    - Test 'Test nahodnymi vstupy + kontrola prace s pameti': Úspěch
        - Dosaženo: 100.00 %, požadováno: 50.00 %
        - Celková doba běhu: 0.484 s (limit: 2.000 s)
        - Využití paměti: 19632 KiB (limit: 29366 KiB)
        - Úspěch v závazném testu, hodnocení: 100.00 %
    - Všechny paměťové bloky byly uvolněné - ok.
    - Celkové hodnocení: 100.00 % (= 1.00 * 1.00 * 1.00 * 1.00)
- Celkové procentní hodnocení: 100.00 %
- Bonus za včasné odevzdání: 0.40
- Celkem bodů: 1.00 * (4.00 + 0.40) = 4.40

		Celkem	Průměr	Maximum	Jméno funkce
SW metriky:	Funkce:	**29**	--	--	--
	Řádek kódu:	**416**	**14.34 ± 22.77**	**102**	`main(int,char **)`
	Cyklomatická složitost:	**92**	**3.17 ± 5.48**	**29**	`operator >>(istream &,CMatrix &)`

2	**25.03.2013 18:57:04**	**Download**
Stav odevzdání:	Ohodnoceno	
Hodnocení:	0.0000	

- **Hodnotitel: automat**
    - Chyba při základní kompilaci **[Zpřístupnit nápovědu (531 B)]**

- Celkové procentní hodnocení: 0.00 %
- Bonus za včasné odevzdání: 0.40
- Celkem bodů: 0.00 * (4.00 + 0.40) = 0.00

1	**25.03.2013 12:17:37**	**Download**
Stav odevzdání:	Ohodnoceno	
Hodnocení:	0.0000	

- **Hodnotitel: automat**
  - Chyba při základní kompilaci **[Zpřístupnit nápovědu (2069 B)]**
- Celkové procentní hodnocení: 0.00 %
- Bonus za včasné odevzdání: 0.40
- Celkem bodů: 0.00 * (4.00 + 0.40) = 0.00