Laboratorium 7 – przeciążanie operatorów cd.

Zadanie 12 (cd. zadania z laboratorium 4.)

Stwórz klasę *Tablica2D*, która implementuje dynamiczną tablicę dwuwymiarową. Klasa ma posiadać:

- prywatne pola przechowujące: szerokość, wysokość i nazwę tablicy;
- wskaźnik do elementu typu *int* (w nim zostanie zapisany wskaźnik do początku jednowymiarowej tablicy dynamicznej).
- ...

W klasie tej zdefiniować operatory:

- **operator wyjścia** << do wyświetlania macierzy (w faktycznej postaci macierzy zawierającej wiersze i kolumny);
- **operator wejścia** >> do wczytywania macierzy;
- +, - ,* realizujące sumę, różnicę i mnożenie dwóch macierzy odpowiednio. Wynikiem każdego z działań ma być macierz.

W programie dodać menu, które pozwoli użytkownikowi:

- pobrać macierz;
- wykonać działanie:
  - suma;
  - różnica;
  - mnożenie;
- wyświetlić macierz o podanej nazwie.

Operatory przeciążone wyjścia i wejścia powinny być funkcjami zaprzyjaźnionymi z klasą:

- w operatorze << parametrem powinna być stała referencja na obiekt danej klasy

```
friend ostream& operator<< (ostream&,Tablica2D const&);
```

- w operatorze >> nie można używać stałej referencji, ponieważ zmieniane są wartości elementów obiektu

```
friend istream& operator>> (istream&, Tablica2D &);
```

Przykład

```
class L_zesp{
private:
      double re, im;
public:
      L_zesp(int a=0,b=0) {re=a; im=b;}
      friend ostream& operator<< (ostream &wyjscie, L_zesp const& liczba);
      //inne metody zadeklarowane w klasie
}

ostream& operator<< (ostream &wyjscie, L_zesp const& liczba)
{
  wyjscie << "Liczba zespolona: " <<endl;
  wyjście << liczba.re;
```

```
  if (liczba.im>0) wyjscie<< " + i*" << liczba.im << endl;
  else if (liczba.im<0) wyjscie<< " - i*" << fabs(liczba.im) << endl;
  else wyjscie<<endl;
  return wyjscie;
}
```

Wywołanie w programie operatora wyjścia ma postać:

```
      L_zesp liczba1(5,-7);
      cout<<liczba1;
```