```cpp
#include <iostream>
#include <ostream>
#include <cmath>
using namespace std;

//*****************************************************************************
//Klasa liczb zespolonych
class zespolona
  {
  public:
  //dane
  double real;
  double imag;
  //konstruktor
  zespolona();
  zespolona(double re, double im);
  //metody
  double mod();
  void print();
  //operatory
  zespolona operator=(zespolona z);
  zespolona operator+(zespolona z);
  //operator << dla strumieniowego wyjscia
  friend ostream& operator<<(ostream& wy, const zespolona& z);
  };


//*****************************************************************************
//konstruktor inicjujący dane zerami
zespolona::zespolona()
  {
  real = 0;
  imag = 0;
  }

//konstruktor inicjujący dane podanymi wartosciami
zespolona::zespolona(double re, double im)
  {
  real = re;
  imag = im;
  }

//modul liczby zespolonej
double zespolona::mod()
  {
  return sqrt(real*real+imag*imag);
  }

//wypisanie liczby na cout
void zespolona::print()
  {
  cout << "(" << real << (imag<0?"-j":"+j") << abs(imag) << ")";
  }

//w tym przykladzie nie ma potrzeby przeciazania operatora = (przyklad)
zespolona zespolona::operator=(zespolona z)
  {
  real = z.real;
  imag = z.imag;
  return *this;
  }

//przeciazony operator dodawania
zespolona zespolona::operator+(zespolona z)
  {
  zespolona tmp;
  tmp.real = real + z.real;
  tmp.imag = imag + z.imag;
  return tmp;
  }
```

```
 71:
 72: //przeciazony operator << dla strumieniowego wyjscia
 73: ostream& operator << (ostream &wy, const zespolona& z)
 74:   {
 75:   wy << "(" << z.real << (z.imag<0?"-j":"+j") << abs(z.imag) << ")";
 76:   return wy;
 77:   }
 78:
 79:
 80: //*****************************************************************************
 81: int main()
 82:   {
 83:   //deklaracja obiektow zainicjowanych
 84:   zespolona a(2,3), b(3,-4);
 85:   zespolona c;
 86:
 87:   //wypisanie metoda print
 88:   cout << "a = ";
 89:   a.print();
 90:   cout << endl;
 91:
 92:   //dodawanie
 93:   c=a+b;
 94:   //wypisanie strumieniowe za pomoca cout
 95:   cout << a << " + " << b << " = " << c << endl;
 96:
 97:   system("pause");
 98:   return 0;
 99:   }
100:
```