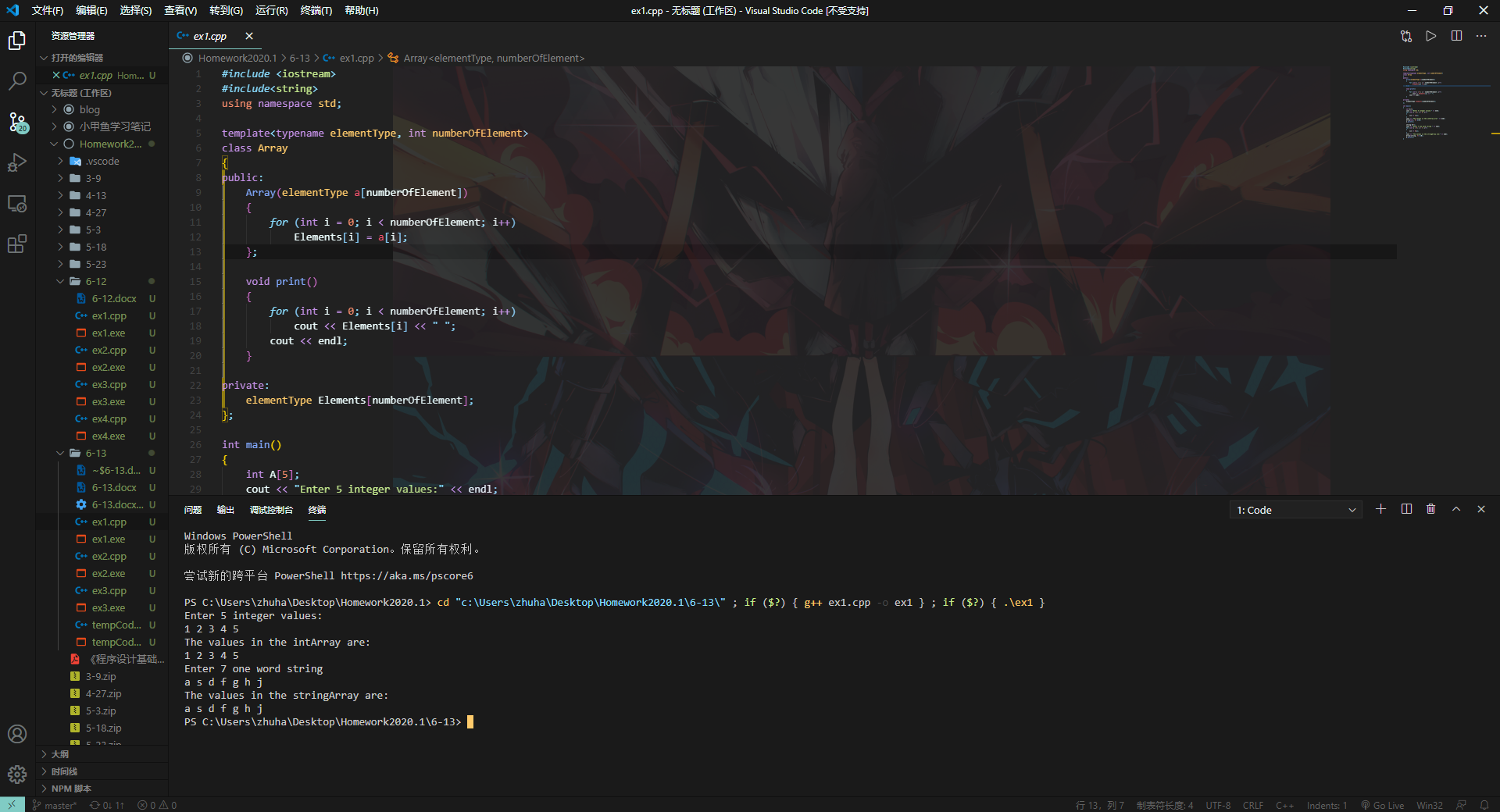
Ex1



*#include* <iostream>

*#include*<string>

using namespace std;

template<typename elementType, int numberOfElement>

class Array

{

public:

    Array(elementType a[numberOfElement])

    {

*for* (int i = 0; i < numberOfElement; i++)

            Elements[i] = a[i];

    };

    void print()

    {

*for* (int i = 0; i < numberOfElement; i++)

            cout << Elements[i] << " ";

        cout << endl;

    }

private:

    elementType Elements[numberOfElement];

};

int main()

{

    int A[5];

    cout << "Enter 5 integer values:" << endl;

*for* (int i = 0; i < 5; i++)

    {

        cin >> A[i];

    }

    cout << "The values in the intArray are:" << endl;

    Array<int, 5> a(A);

    a.print();

    string B[7];

    cout << "Enter 7 one word string " << endl;

*for* (int i = 0; i < 7; i++)

    {

        cin >> B[i];

    }

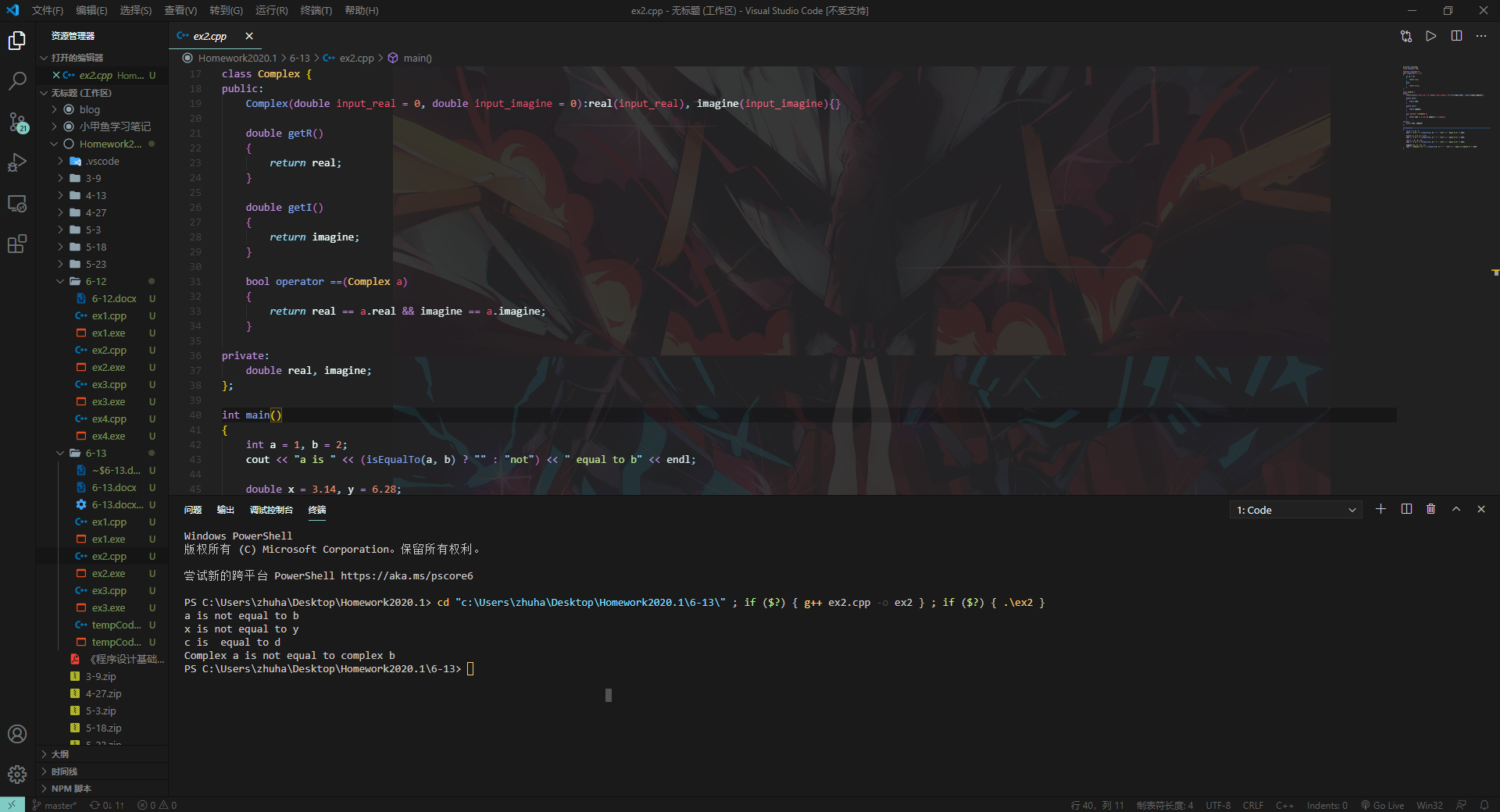
    cout << "The values in the stringArray are:" << endl;

    Array<string, 7> b(B);

    b.print();

}

Ex2



*#include* <iostream>

using namespace std;

template <typename T>

bool isEqualTo(T a, T b)

{

*if* (a == b)

    {

*return* true;

    }

*else*

    {

*return* false;

    }

}

class Complex {

public:

    Complex(double input\_real = 0, double input\_imagine = 0):real(input\_real), imagine(input\_imagine){}

    double getR()

    {

*return* real;

    }

    double getI()

    {

*return* imagine;

    }

    bool operator ==(Complex a)

    {

*return* real == a.real && imagine == a.imagine;

    }

private:

    double real, imagine;

};

int main()

{

    int a = 1, b = 2;

    cout << "a is " << (isEqualTo(a, b) ? "" : "not") << " equal to b" << endl;

    double x = 3.14, y = 6.28;

    cout << "x is " << (isEqualTo(x, y) ? "" : "not") << " equal to y" << endl;

    char c = 'A', d = 65;

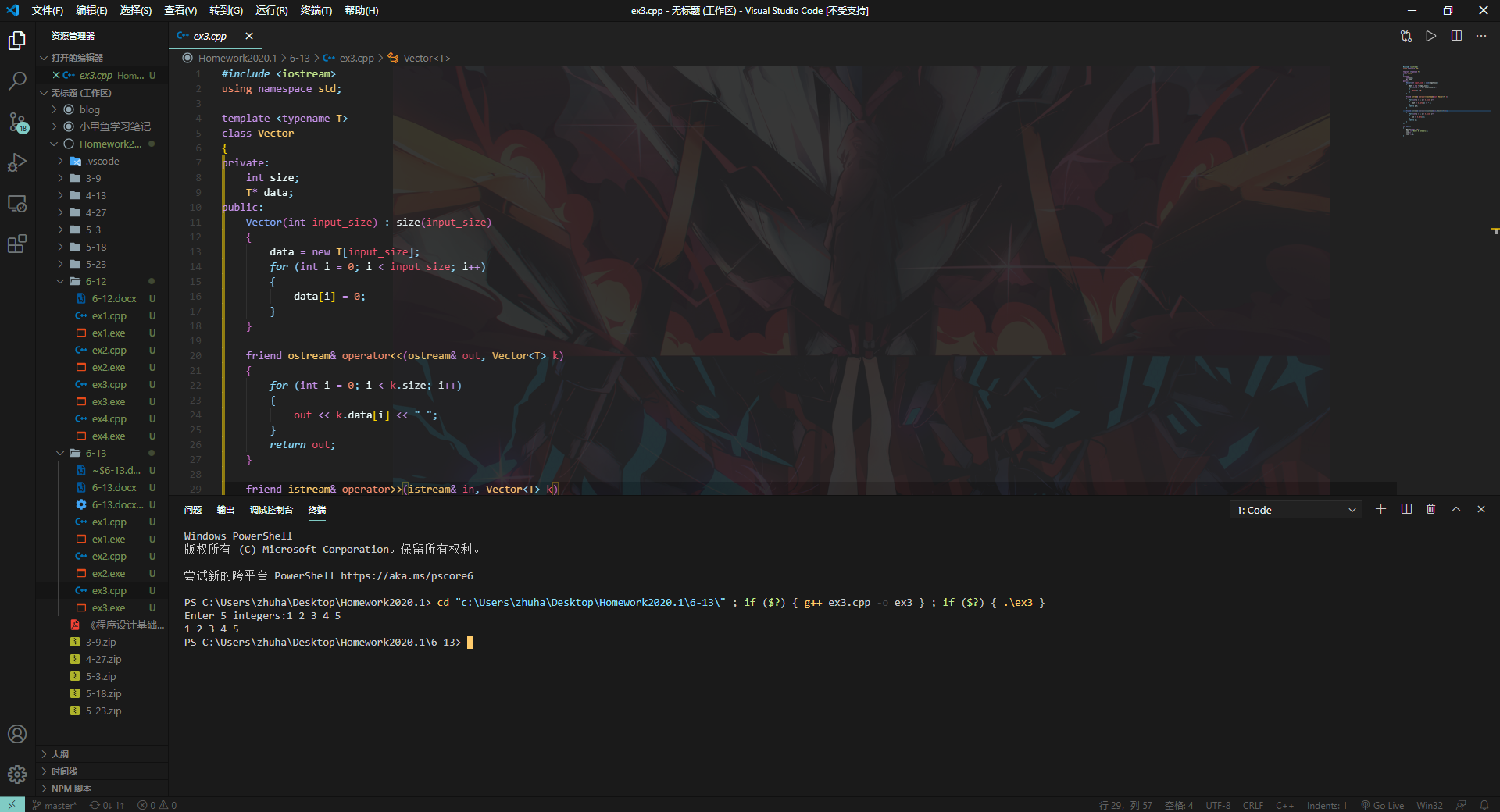
    cout << "c is " << (isEqualTo(c, d) ? "" : "not") << " equal to d" << endl;

    Complex e(2, 3), f(3, 3);

    cout << "Complex a is " << (isEqualTo(e, f) ? "" : "not") << " equal to complex b" << endl;

}

Ex3



*#include* <iostream>

using namespace std;

template <typename T>

class Vector

{

private:

    int size;

    T\* data;

public:

    Vector(int input\_size) : size(input\_size)

    {

        data = new T[input\_size];

*for* (int i = 0; i < input\_size; i++)

        {

            data[i] = 0;

        }

    }

    friend ostream& operator<<(ostream& out, Vector<T> k)

    {

*for* (int i = 0; i < k.size; i++)

        {

            out << k.data[i] << " ";

        }

*return* out;

    }

    friend istream& operator>>(istream& in, Vector<T> k)

    {

*for* (int i = 0; i < k.size; i++)

        {

            in >> k.data[i];

        }

*return* in;

    }

};

int main()

{

    Vector<int> a(5);

    cout << "Enter 5 integers:";

    cin >> a;

    cout << a;

}