3. Текст програми на мові програмування С++

```
#include <iostream>
#include <conio.h>
#include <math.h>

using namespace std;

class Matrix {
public:
      int* data;
      int size;

      //конструктор з ініціалізацією початкових данних
      //першим числом повиинно бути розмір матриці зі знаком мінус
      //в протилежному випадку заповнення масиву початковими значеннями
      //не відбудеться
    Matrix(int matSize, ...) {
        size = matSize;
            if(size<0) size = -size;
        data = new int[size*size];

            if(matSize < 0) {
                  int *p = &matSize;
                  for(int i = 0; i < size*size; i++) {
                        p++;
                        *(data+i) = *(p);
                  }
            }
    }

      //конструктор без параметрів
      //створюється масив 2*2
    Matrix() {
        size = 2;
        data = new int[4];
      }

      //конструктор копій
      Matrix(const Matrix &p) {
            size = p.size;
        data = new int[size*size];

            int* k =    p.data;
            for(int i = 0; i < size*size; i++)
                  *(data+i) = *(k+i);
      }

      //деструктор
    ~Matrix() {
        delete[] data;
    }

      //перезавантаження оператора []
      int* operator[](int row) {
        return data+row*size;
```

```
    }

      //перезавантаження оператора =
      Matrix& operator=(const Matrix& right) {
        //перевірка на самоприсвоєння
        if (this == &right) {
            return *this;
        }
        size = right.size;
        int* k =  right.data;
            for(int i = 0; i < size*size; i++)
                  *(data+i) = *(k+i);

            return *this;
    }

      //Дружні функції

      //унарний "-"
      friend const Matrix operator-(const Matrix& right);

      //бінарний "-" (матриця3 = матриця1 - матриця2)
      friend const Matrix operator-(const Matrix& left, const Matrix& right);

      //бінарний "-" (матриця3 = матриця1 - число)
      friend const Matrix operator-(const Matrix& left, int right);

      //матриця1 += матриця2;
      friend Matrix& operator+=(Matrix& left, const Matrix& right);

      //матриця1 += число;
      friend Matrix& operator+=(Matrix& left, int right);

      //"*" (матриця3 = матриця1 * матриця2)
      friend const Matrix operator*(const Matrix& left, const Matrix& right);

      //"*" (матриця2 = матриця1 * число)
      friend const Matrix operator*(const Matrix& left, int right);

      //перезавантаження виводу масиву на екран
      friend ostream& operator<<(ostream&, const Matrix&);

      //перезавантаження вводу масиву з клавіатури
      friend istream& operator>>(istream&, const Matrix&);
};


//унарний "-"
const Matrix operator-(const Matrix& right) {

      Matrix left(right.size);
      int* k =    right.data;
      for(int i = 0; i < right.size*right.size; i++)
            *(left.data+i) = -*(k+i);

      return left;
```

```
}

//бінарний "-" (матриця3 = матриця1 - матриця2)
const Matrix operator-(const Matrix& left, const Matrix& right) {
    Matrix result(left.size);
      int* k =    left.data;
      int* m =    right.data;
      for(int i = 0; i < left.size*left.size; i++)
            *(result.data+i) = *(k+i) - *(m+i);

      return result;
}

//бінарний "-" (матриця3 = матриця1 - число)
const Matrix operator-(const Matrix& left, int right) {
    Matrix result(left.size);
      int* k =    left.data;
      for(int i = 0; i < left.size*left.size; i++)
            *(result.data+i) = *(k+i) - right;

      return result;
}

//"*" (матриця3 = матриця1 * матриця2)
const Matrix operator*(const Matrix& left, const Matrix& right) {
    Matrix result(left.size);
      int* k =    left.data;
      int* m =    right.data;
      for(int i = 0; i < left.size*left.size; i++)
            *(result.data+i) = (*(k+i)) * (*(m+i));

      return result;
}

//"*" (матриця2 = матриця1 * число)
const Matrix operator*(const Matrix& left, int right) {
    Matrix result(left.size);
      int* k =    left.data;
      for(int i = 0; i < left.size*left.size; i++)
            *(result.data+i) = (*(k+i)) * right;

      return result;
}


//матриця1 += матриця2;
Matrix& operator+=(Matrix& left, const Matrix& right) {
    int* m =      right.data;
      for(int i = 0; i < left.size*left.size; i++)
            *(left.data+i) +=  *(m+i);

      return left;
}

//матриця1 += число;
Matrix& operator+=(Matrix& left, int right) {
```

```
	for(int i = 0; i < left.size*left.size; i++)
		*(left.data+i) +=  right;

	return left;
}

//перезавантаження виводу масиву на екран
ostream& operator<<(ostream& output, const Matrix& matr) {
	int* m =    matr.data;

	for(int i = 0, k = 0; i < matr.size; i++) {
		for(int j = 0; j < matr.size; j++, k++) {
			output<<(*(m+k))<<"\t";
		}
		output<<"\n";
	}
	return output;
}

//перезавантаження вводу масиву з клавіатури
istream& operator>>(istream& input, const Matrix& matr) {
	int* m =    matr.data;

	for(int i = 0, k = 0; i < matr.size; i++)
		for(int j = 0; j < matr.size; j++, k++) {
			cout<<"["<<i+1<<"]["<<j+1<<"]: ";
			input>>(*(m+k));
		}
	return input;
}

void main() {
	const int N = 2;

	//з початковою ініціалізацією
	Matrix a(-2,1,2,3,4);
	cout<<"a = {1,2,3,4}:\n"<<a;

	//без параметрів - по замовчуванню 2*2
	Matrix b;
	cout<<"\nInput b:\n";
	cin>>b;
	cout<<"\nb:\n"<<b;

	//з вказаним розміром, без ініціалізації
	Matrix c(N);
	c = -b;
	cout<<"\nc = -b\n"<<c;

	//копіюванням
	Matrix d(c);
	cout<<"\nMatrix d(c)\n"<<d;

	d = b - 5;
	cout<<"\nd = b - 5\n"<<d;
```

```
        b = d - a;
        cout<<"\nb = d - a\n"<<b;

        c = a * 2;
        cout<<"\nc = a * 2\n"<<c;

        a = b * c;
        cout<<"\na = b * c\n"<<a;

        a += 3;
        cout<<"\na += 3\n"<<a;

        a += c;
        cout<<"\na += c\n"<<a;

        _getch();
}
```