ЦЕЛЬ РАБОТЫ

Продемонстрировать работу с форматирующими методами и манипуляторами.

ПРАКТИЧЕСКОЕ ЗАДАНИЕ

Продемонстрировать работу с форматирующими методами и манипуляторами на примере перегруженных операций записи в поток из класса реализованного в 8 лабораторной работе.

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Вариант 10. Ломаная линия

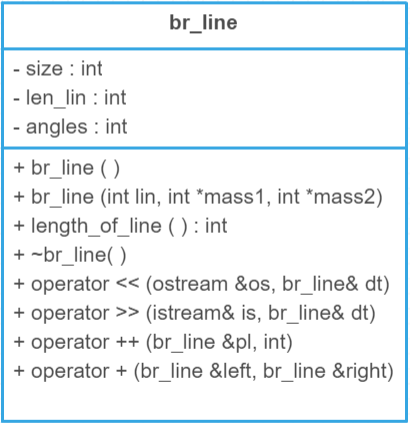
БЛОК-СХЕМА АЛГОРИТМА

Блок-схема алгоритма представлена на рисунке 1.

*Рисунок 1 Блок-схема программы*

ДИАГРАММА КЛАССОВ

Диаграмма классов представлена на рисунке 2.



*Рисунок 2 Диаграмма классов*

ЛИСТИНГ

#include <iostream>

#include <string>

#include <iomanip>

using namespace std;

class br\_line {

private:

int size;//количество звеньев

int \*len\_lin;//массив из длин звеньев

int \*angles;//массив из углов между звеньями

public:

br\_line() { //конструктор по умолчанию

size = 3;

len\_lin = new int[size];

len\_lin[0] = 10; len\_lin[1] = 10; len\_lin[2] = 10;

angles = new int[size];

angles[0] = 1; angles[1] = 1;

}

br\_line(int lin, int \*mass1, int \*mass2) { //параметризированный конструктор

size = lin;

len\_lin = new int[size];

for (int i = 0; i < size; i++) {

len\_lin[i] = mass1[i];

}

angles = new int[size - 1];

for (int i = 0; i < (size - 1); i++) {

angles[i] = mass2[i];

}

}//lin\_k-количество звеньев

~br\_line() {};

int length\_of\_line()

{

int sum = 0;

for (int n = 0; n < size; n++)

{

sum = sum + len\_lin[n];

}

return sum;

}//функция высчитывает длину ломанной

friend br\_line operator+(br\_line &left, br\_line &right);

friend ostream& operator<<(ostream &os, br\_line& dt);

friend istream& operator>>(istream& is, br\_line& dt);

friend br\_line operator++(br\_line &pl, int);

};

br\_line operator+(br\_line& left, br\_line& right)//перегрузка бинарного прератора

{

br\_line sum;

sum.size = left.size + right.size;

return sum;

}

br\_line operator++(br\_line& pl, int) //перегрузка унарного оператора

{

pl.size++;

return pl;

}

ostream& operator<<(ostream &os, br\_line& dt) //перегрузка оператора вывода

{

os << dt.length\_of\_line() << endl;

return os;

}

istream& operator>>(istream& is, br\_line &dt) //перегрузка оператора ввода

{

is >> dt.size;

return is;

}

int main(void)

{

setlocale(LC\_ALL, "Russian");

br\_line A1;

cout << A1 << endl;

cout.width(25);

cout.fill('\*');

cout << hex << A1 << endl;

cout << setbase(2) << A1 <<endl;

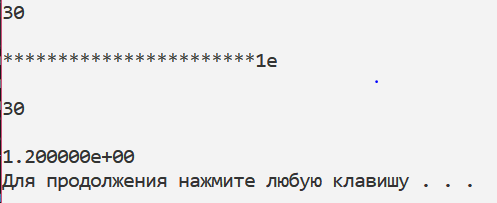
cout << scientific << 1.2 << endl;

system("pause");

}

ПРОТОКОЛ ВЫПОЛНЕНИЯ ПРОГРАММЫ

Протокол выполнения программы представлен на рис. 3.



*Рисунок 3 Протокол программы*