МИНОБРНАУКИ РОССИИ

ФГБОУ ВО «Тульский государственный университет»

Институт прикладной математики и компьютерных наук

Кафедра вычислительной техники

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

*Направление подготовки*: 09.03.01 “Информатика и вычислительная техника”

*Формы обучения*: очная

Билет № 1 промежуточной аттестации

Разработать классы Point и Array, позволяющие использовать их в следующей программе:

Point p1(10,20), p2(2,7), p3=p1; p2.assign(p1-p3); cout<<(p2);

Array a1(10), a2(10); a2=a1; a2.assign(a2-a1); cout<<a2;

Написать тексты h-файла и cpp-файла для классов Point и Array.

Билет рассмотрен и утвержден на заседании кафедры ВТ "\_27\_" \_апреля\_2020 г.,

протокол № 10.

Заведующий кафедрой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.Н. Ивутин

Файл Array.h:

#pragma once

#include <iostream>

using namespace std;

#include "Point.h"

class Array

{

private:

Point\* point\_array;

int array\_size;

public:

Array();

Array(int array\_size);

Array(const Array& new\_array);

Array& operator=(const Array& new\_array);

Array operator-(const Array& array\_for\_subtraction) const;

friend ostream& operator<<(ostream&, Array& array);

Point& operator[](const int& index) const;

Array& assign(const Array& new\_array);

};

Файл Point.h:

#pragma once

#include <iostream>

using namespace std;

class Point

{

private:

double x, y;

public:

Point();

Point(double x, double y);

Point(const Point& point);

Point& operator=(const Point& new\_point);

bool operator==(const Point& point\_for\_comparison) const;

friend ostream& operator<<(ostream& out, const Point& point);

friend Point operator-(const Point& point\_1, const Point& point\_2);

Point& assign(const Point& point);

ostream& print(ostream& out) const;

};

Файл Array.cpp:

#include "Array.h"

Array::Array()

{

array\_size = 3;

point\_array = new Point[array\_size];

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

{

point\_array[i].assign(Point(0, 0));

}

}

Array::Array(int array\_size)

{

if (array\_size > 0)

{

this->array\_size = array\_size;

point\_array = new Point[array\_size];

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

{

point\_array[i].assign(Point(0, 0));

}

}

else

{

cerr << "\n Неправильная размерность: " << array\_size << endl;

this->array\_size = 0;

point\_array = nullptr;

}

}

Array::Array(const Array& new\_array)

{

array\_size = new\_array.array\_size;

point\_array = new Point[array\_size];

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

{

point\_array[i].assign(new\_array.point\_array[i]);

}

}

Array& Array::operator=(const Array& new\_array)

{

if (&new\_array != this)

{

delete[] point\_array;

point\_array = new Point[new\_array.array\_size];

array\_size = new\_array.array\_size;

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

{

point\_array[i].assign(new\_array.point\_array[i]);

}

}

return \*this;

}

Array Array::operator-(const Array& array) const

{

int size\_for\_temp\_array = array\_size < array.array\_size ? array\_size : array.array\_size;

Array temp\_array(size\_for\_temp\_array);

for (int i = 0; i < size\_for\_temp\_array; ++i)

{

temp\_array[i] = point\_array[i] - array.point\_array[i];

}

return temp\_array;

}

ostream& operator<<(ostream& out, Array& array)

{

for (int i = 0; i < array.array\_size; ++i)

{

array.point\_array[i].print(out);

}

return out;

}

Point& Array::operator[](const int& index) const

{

if (index < 0 || index >= array\_size)

{

cerr << "\n Ошибка индекса: " << index << endl;

exit(1);

}

return point\_array[index];

}

Array& Array::assign(const Array& new\_array)

{

if (point\_array == new\_array.point\_array)

{

return \*this;

}

delete[] point\_array;

point\_array = new Point[new\_array.array\_size];

memcpy(point\_array, new\_array.point\_array, sizeof(Point) \* new\_array.array\_size);

array\_size = new\_array.array\_size;

return \*this;

}

Файл Point.cpp:

#include <iostream>

#include "Point.h"

Point::Point() : x(0), y(0) {}

Point::Point(double new\_x, double new\_y) : x(new\_x), y(new\_y) {}

Point::Point(const Point& new\_point) : x(new\_point.x), y(new\_point.y) {}

Point& Point::operator = (const Point& new\_point)

{

if (\*this == new\_point)

{

return \*this;

}

x = new\_point.x, y = new\_point.y;

return \*this;

}

bool Point::operator == (const Point& point\_for\_comparison) const

{

return x == point\_for\_comparison.x && y == point\_for\_comparison.y;

}

ostream& operator << (ostream& out, const Point& point)

{

return out << "x = " << point.x << "\ty = " << point.y << endl;

}

Point operator - (const Point& point\_1, const Point& point\_2)

{

return Point(point\_1.x - point\_2.x, point\_1.y - point\_2.y);

}

Point& Point::assign(const Point& new\_point)

{

x = new\_point.x, y = new\_point.y;

return \*this;

}

ostream& Point::print(ostream& out) const

{

return out << "x = " << x << " " << "y = " << y << endl;

}

Файл Prog.cpp:

#include <iostream>

#include "Array.h"

using namespace std;

int main()

{

Point p1(10, 20), p2(2, 7), p3 = p1;

p2.assign(p1 - p3);

cout << (p2);

Array a1(10), a2(10);

a2 = a1;

a2.assign(a2 - a1);

cout << a2;

return 0;

}