## Лабораторная работа №3. Перегрузка операторов.

Согласно варианту описать указанные типы данных и поместить их в отдельный заголовочный файл, в нем же описать операторы, указанные в варианте. Реализацию функций поместить с отдельный cpp файл.

Написать программу, проверяющую правильность работы – для наглядности и лучшего усвоения материала использовать как явный, так и не явный метод вызова функций операторов (см. пример в конце задания).

Варианты.

|  |  |  |
| --- | --- | --- |
| № | Тип данных | Операторы |
| 1 | Комплексное число | Умножение комплексного числа на вещественное число.  Сложение двух комплексных чисел.  Умножение двух комплексных чисел.  Длина комплексного числа (используйте для этого, например, операцию «приведение к double» – operator double (Complex&). |
| 2 | Квадрат на плоскости.  Задается координатой левого верхнего угла, стороной квадрата и углом, на который квадрат повернут относительно оси OX. | Равенство площадей квадратов (перегрузите операции ==, !=, <, >)  Умножение квадрата на вещественное число (увеличивает сторону квадрата).  Прибавление к квадрату вектора (смещение квадрата на указанный вектор). |
| 3 | Треугольник на плоскости (самостоятельно выбирайте необходимые данные). | Равенство площадей треугольников (перегрузите операции ==, !=, <, >) – для вычисления площади можете использовать, например, формулу Герона (зависит от тех данных, что используются для хранения треугольника).  Прибавление вектора (смещение треугольника на указанный вектор). |
| 4 | Матрица 3x3 | Перемножение двух матриц.  Умножение матрицы на вещественное число.  Вычитание и сложение матриц.  Сравнение матриц (==, !=, >, <); |
| 5 | Стек целых чисел глубиной не более 100. | Добавление числа в стек (operator << ).  Изъятие числа из стека (operator >>).  Не забудьте написать простую функцию (не оператор) для вывода стека на экран – с ней будет удобнее производить отладку. |
| 6 | Массив целых чисел (длина не более 100). | Объединение двух массивов в один (operator+)  Сравнение длин массивов (==, >, < !=). |
| 7 | Подмножество множества целых чисел от нуля до девяти: { 0, 1, 2, … 9 }. (см. примечание ниже) | Объединение двух множеств (operator+).  Сравнение (== и !=).  Добавление числа в множество (operator+=).  Изъятие числа из множества (operator-=). |
| 8 | FIFO (очередь) целых чисел длиной не более 100. | Добавить целое число в очередь (operator<<)  Взять число из очереди (operator>>) |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |  |
| **1** |  |  | X |  |  | X |  |  | **1** |
| **2** |  |  |  | X |  |  | X |  | **2** |
| **3** | X |  |  |  | X |  |  |  | **3** |
| **4** |  | X |  |  |  | X |  |  | **4** |
| **5** |  |  | X |  |  |  | X |  | **5** |
| **6** |  |  |  | X |  |  |  | X | **6** |
| **7** | X |  |  |  |  | X |  |  | **7** |
| **8** |  | X |  |  |  |  | X |  | **8** |
| **9** |  |  | X |  |  |  |  | X | **9** |
| **10** |  |  |  | X | X |  |  |  | **10** |
| **11** | X |  |  |  |  |  | X |  | **11** |
| **12** |  | X |  |  |  |  |  | X | **12** |
| **13** |  |  | X |  | X |  |  |  | **13** |
| **14** |  |  |  | X |  | X |  |  | **14** |
| **15** | X |  |  |  |  |  |  | X | **15** |
| **16** |  | X |  |  | X |  |  |  | **16** |
| **17** |  |  | X |  |  | X |  |  | **17** |
| **18** |  |  |  | X |  |  | X |  | **18** |
| **19** | X |  |  |  | X |  |  |  | **19** |
| **20** |  | X |  |  |  | X |  |  | **20** |
| **21** |  |  | X |  |  |  | X |  | **21** |
| **22** |  |  |  | X |  |  |  | X | **22** |
| **23** | X |  |  |  |  | X |  |  | **23** |
| **24** |  | X |  |  |  |  | X |  | **24** |
|  | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** |  |

Для варианта 7.

Примеры подмножеств:

Пустое{}, {0, 9}, {3}, {2,4,7,8,9}, и т.д.

Думаю, что удобнее всего использовать массив mas[10] такой, что mas[ i ] равно 1 тогда и только тогда, когда i содержится в подмножестве.

// -------------- mystruct.h

#ifndef \_COMPLEX\_TYPES\_H\_

#define \_COMPLEX\_TYPES\_H\_

struct MyStruct { /\* some fields here \*/ };

MyStruct operator+( MyStruct, MyStruct );

MyStruct operator+( MyStruct, double );

MyStruct& operator=( const MyStruct& );

/\* etc… \*/

#endif

// --------------

// -------------- mystruct.cpp

#include “mystruct.h”

MyStruct operator+( MyStruct a, MyStruct b ) { /\*…\*/ }

MyStruct operator+( MyStruct a, double x ) { /\*…\*/ }

MyStruct& operator=( const MyStruct& a ) { /\*…\*/ }

// --------------

// -------------- main.cpp

#include “mystruct.h”

#include <cstdio>

int main() {

MyStruct a, b, c;

double x;

a = b + c;

a = operator+( b, c );

operator=( a, operator( b, c ) );

// etc…

return 0;

}

// --------------