**Лабораторная работа №4**

**Задание №1**

**Постановка задачи:**

Создать класс Int.

Перегрузить четыре бинарных целочисленных арифметических операции (+,-,\*,/) и унарные операции постфиксной и префиксной форм инкремента.

Использовать концепцию исключения если результат выходит за границы типа int.

При описании унарных операций использовать указатель this.

**Список используемых переменных:**

Начальные данные:

Int\_max – max of integer

Int\_min – min of integer

number – целочисленная переменная

Выходные данные:

result – переменная хранящий результат выполнения операции

**Описание программы:**

Программа была реализована с помощью класса Int, а также был использован конструктор c аргументами. При реализации использовались функции из стандартных библиотек.

main() – управляет последовательностью вызова функций.

operator + () – бинарный оператор, выполняет операцию сложения, если результат операции выходит за границы типа int выводит соответствующее сообщение. (рис.1)

operator - () – бинарный оператор, выполняет операцию вычитания, если результат операции выходит за границы типа int выводит соответствующее сообщение.

operator \* () – бинарный оператор, выполняет операцию умножения, если результат операции выходит за границы типа int выводит соответствующее сообщение.

operator / () – бинарный оператор, выполняет операцию деления, если результат операции выходит за границы типа int выводит соответствующее сообщение.

operator ++ () – унарный оператор, префиксная форма инкремента, если результат операции выходит за границы типа int вывод соответствующее сообщение. (рис.2)

operator ++ (int) – унарный оператор, постфиксная форма инкремента, если результат операции выходит за границы типа int вывод соответствующее сообщение. (рис.3)

Show() – выводит на экран значения числа number.

рис.1

рис.2 рис.3

**Задание №2**

**Постановка задачи:**

Создать класс fraction, с закрытым полем chislo типа double.

Перегрузить операнды так, чтобы они могли оперировать как с объектами класса, так и с числами.

Перегрузите унарную операцию инкремента в префиксной или постфиксной форме.

**Список используемых переменных:**

Начальные данные:

chislo – переменная типа double

Выходные данные:

Результат операндов

**Описание программы:**

Программа была реализована с помощью класса fraction, а также был использован конструктор c аргументами и без. При реализации использовались функции из стандартных библиотек.

main() – управляет последовательностью вызова функций.

operator + () – бинарный оператор, выполняет операцию сложения.

operator - () – бинарный оператор, выполняет операцию вычитания.

operator \* () – бинарный оператор, выполняет операцию умножения.

operator / () – бинарный оператор, выполняет операцию деления.

operator ++ () – унарный оператор, префиксная форма инкремента.

operator ++ (int) – унарный оператор, постфиксная форма инкремента.

Show() – выводит на экран значения числа number.

**Задание №3**

**Постановка задачи:**

Создать класс, в котором перегружается метод **rect\_area()** (возвращает площадь прямоугольника).

Перегрузить метод двумя способами: в первом передать два аргумента, во втором только один.

**Список используемых переменных:**

Начальные данные:

Width – ширина фигуры

Length – длина фигуры

Выходные данные:

Площадь фигуры

**Описание программы:**

Программа была реализована с помощью класса Area, а также был использован конструктор c аргументами. При реализации использовались функции из стандартных библиотек.

main() – управляет последовательностью вызова функций.

rect\_area – перегруженный метод вычисляющая площадь прямоугольника либо квадрата, в зависимости от количества аргументов.

**ВОПРОСЫ К ЗАЩИТЕ ЛАБОРАТОРНОЙ РАБОТЫ №4**

1. Что такое полиморфизм?

Полиморфизм – способность функции обрабатывать данные разных типов. Цель полиморфизма – использование одного имени метода для задания общих для класса действий и тем самым снижение сложности программ.

1. Что такое перегрузка операторов?

Перегрузка операторов – это возможность заставить компилятор осуществлять стандартные операции над нестандартными (пользовательскими) типами данных.

1. Что такое переопределение операторов?

Переопределение

1. Каков синтаксис операторной функции?

Тип\_результата **operator** символ\_ оператора (аргументы: тип и название)

{

программный код

}

1. Какие операторы можно перегружать, а какие нельзя?

Можно перегружать следующие операторы:

+ - \* / % ^ & | ~ ! = < >  
+= -= \*= /= %= ^= &= |=  
<< >> >>= <<= == != <= >=  
&& || ++ -- ->\* , -> [] ()  
new new[] delete delete[]

Нельзя перегружать операторы:

1. :: - (разрешение области видимости)
2. . - (доступ к членам класса)
3. - (выбор члена через указатель на член)
4. - ? тернарный оператор
5. Какова зависимость между количеством аргументов в операторной функции и количеством операндов? Объясните эту зависимость.

Перегруженной операции всегда требуется количество аргументов на один меньше, чем количество операндов, так как один из операндов является объектом, вызывающим операторную функцию. Поэтому для унарных операторов аргументы не нужны, а для бинарных нужен один аргумент.

1. Каким образом перегруженные операции позволяют вид программного кода сделать более читабельным?

Например, код операции сложения объектов в виде в точечном формате:

Cars.car3 = Cars.car2 + Cars.car2;

Операторная функция даёт возможность заменить этот код на более простой:

car3 = car2 + car2;