Лабораторная работа №3

Тема: Наследование и полиморфизм. Абстрактные классы и интерфейсы. RTTI.

Цель: Научиться реализовывать на С++ наследование классов, программировать абстрактные классы и интерфейсы, виртуальные методы, а также динамически определять тип объекта во время выполнения программы.

Индивидуальные задания:

**1.** Написать в ООП-стиле код программы, позволяющей работать с арифметическими выражениями разного вида, оперирующими вещественными числами: вычислять результат выражения, выводить запись выражения на консоль и в файл лога. Например, для вычисления выражений вида (10+4+2+3+7+1) и (1+2.5) будет использоваться класс Summator, выражений вида (2\*3\*7\*1) – класс Multiplier, и т.д. В коде необходимо отразить следующее:

* Создать интерфейс ILoggable с 2 методами (функционал логирования): Запись лога всего выражения на консоль: void logToScreen() Добавление записи лога всего выражения в файл лога: void logToFile(const std::string& filename).
* Создать абстрактный класс ExpressionEvaluator, реализующий интерфейс ILoggable и предоставляющий чисто виртуальный метод double calculate() для вычисления результата произвольного выражения. Количество операндов должно храниться в отдельном члене класса. Сами операнды х1,х2,х3 и т.д. должны храниться в члене данного класса – массиве, в куче (динамической памяти).
* Класс ExpressionEvaluator должен предоставлять два конструктора и виртуальный деструктор. В конструкторе без параметров выделять память под 20 операндов и инициализировать их нулями, в конструкторе с параметром n – выделять память под n элементов и инициализировать нулями. Также необходимо реализовать 2 метода, позволяющие присвоить операндам конкретные значения: Присвоить значение value одному операнду на позиции pos: void setOperand(size\_t pos, double value) Заполнить сразу группу из n операндов массивом значений ops: void setOperands(double ops[], size\_t n)
* В деструкторе должна освобождаться память, выделенная в конструкторе.
* Создать два подкласса класса ExpressionEvaluator, работающих со стандартными выражениями, в соответствии с вариантом, из четырех возможных: Summator – сумма всех операндов (х1 + х2 + х3 + х4 + ...) Subtractor – разность всех операндов (х1 – х2 – х3 – х4 – ...) Multiplier – произведение всех операндов (х1 \* х2 \* х3 \* х4 \* ...) Divisor – частное всех операндов (х1/х2/х3/х4/...), но если хоть один операнд равен 0, то результату выражения присвоить также 0.
* Создать подкласс CustomExpressionEvaluator, работающий со специфическими выражениями, вид которых приведен в варианте
* Подклассы ExpressionEvaluator, для которых порядок следования операндов важен, должны также реализовывать интерфейс IShuffle. Данный интерфейс объявляет 2 перегруженных метода (функционал перемешивания операндов): Произвольно перемешать операнды: void shuffle() Перемешать операнды, находящиеся на позициях i и j: void shuffle(size\_t i, size\_t j) В функции main() необходимо продемонстрировать работу созданных классов:
* Создать массив из трех указателей на класс обработки арифметических выражений.
* В соответствии с вариантом, создать в куче три объекта конкретных подклассов обработки арифметических выражений и установить на них указатели; присвоить их операндам значения двумя способами (поэлементным и групповым).
* Продемонстрировать полиморфизм: организовать проход в цикле по указателям и вывести лог выражения на консоль и в файл (в консоли отобразить еще и сам результат выражения).
* Организовать цикл по указателям, в теле которого средствами С++ проверить, реализует ли текущий объект интерфейс IShuffle. Если да, то вызвать один из методов shuffle() этого объекта и отобразить на экране запись выражения после перемешивания операндов, а также вычислить и отобразить результат нового выражения.

**2.** Дополнить код задания 3 лабораторной работы № 2, написав еще два класса по предметной области, в соответствии с вариантом. Продумать и корректно реализовать схему наследования классов. В главной функции продемонстрировать применение интерфейса, полиморфизм и RTTI на примере 3-4 объектов классов, по аналогии с заданием 1. 3. Написать свой аналог стандартной функции обработки строк из файла cstring, в соответствии с вариантом. В функции main на тестовых данных продемонстрировать результат работы как стандартной функции, так и собственной версии. Ввод-вывод данных организовать средствами cstdio.

Ход работы

Вариант 4

**1.** Вид выражения CustomExpression: result = x1 + 2\*x2 + 3\*x3 + 4\*x4 + ...

Порядок создания и инициализации объектов подклассов:

CustomExpressionEvaluator: 4 операнда, присвоить поэлементно 50, 40, -10, -2.

Multiplier: 4 операнда, присвоить группой -0.5, -8, 1.5, 16.

Summator: 6 операндов, присвоить группой 2.6,-8.1, 13.2, 1.5, 3.4, -4.

Метод shuffle() – отсортировать все операнды в порядке убывания.

Метод shuffle(size\_t i, size\_t j) – отсортировать все операнды между i-ым и j-ым.

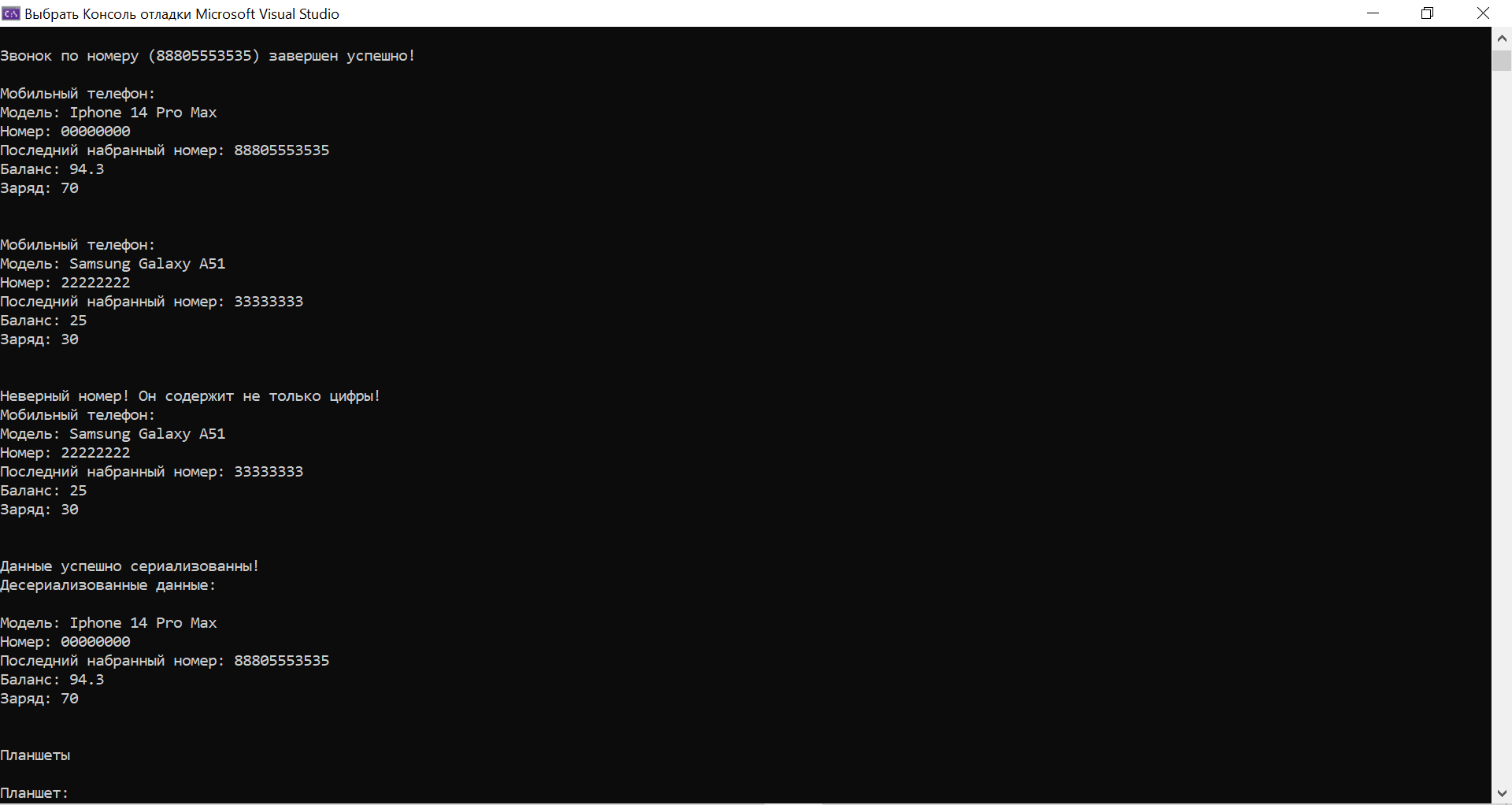
Формат вывода: 4 operands : 50 + 2\*40 + 3\*(-10) + (-2) -> 98

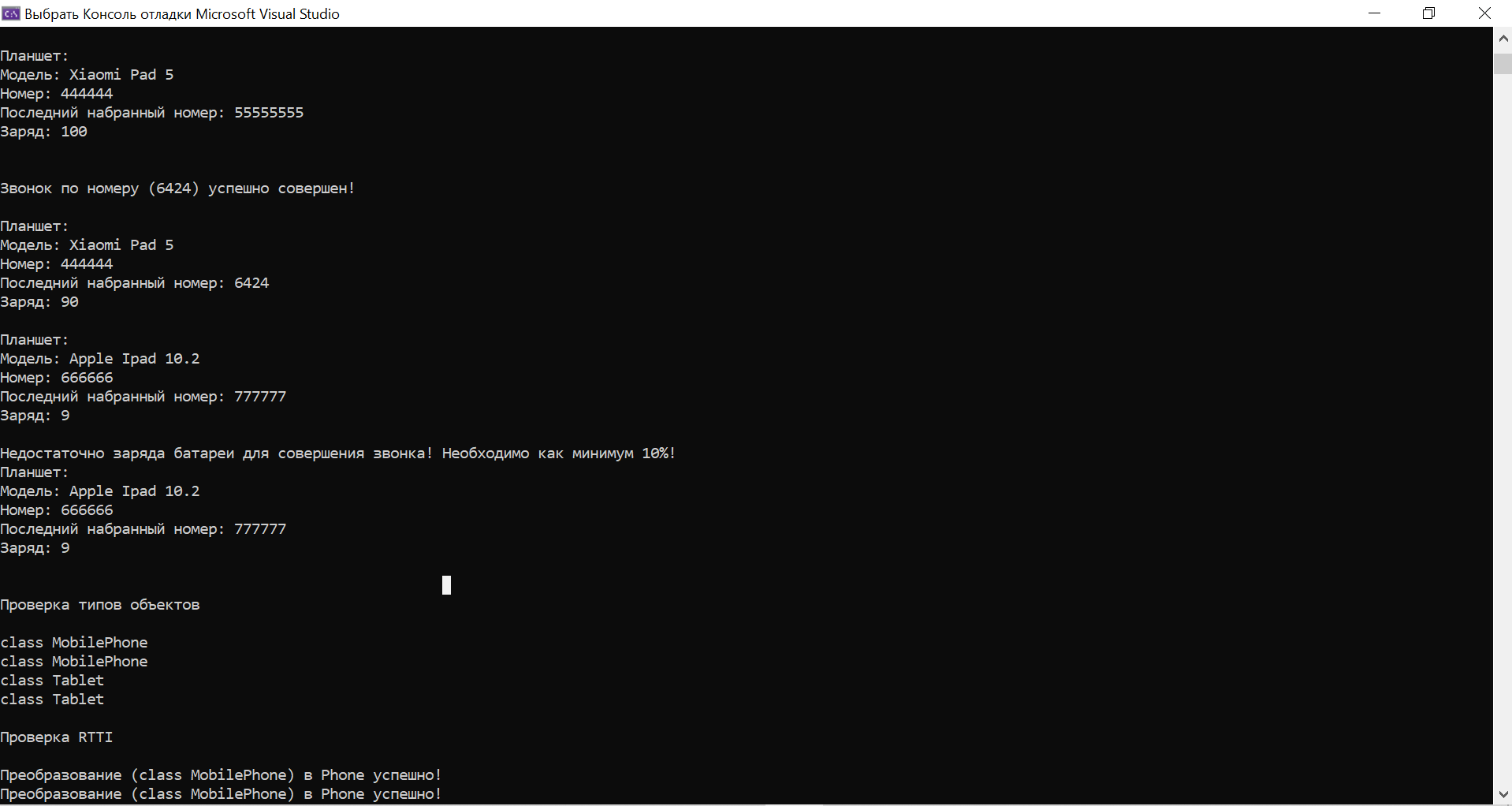
**2.** Класс ТЕЛЕФОН + классы МОБИЛЬНОЕ УСТРОЙСТВО, ПЛАНШЕТ. Реализовать схему наследования классов и корректно распределить по классам данные: фирма-производитель, модель, номер телефона, последний набранный номер, остаток на счету, вес, цвет, цена, уровень заряда.

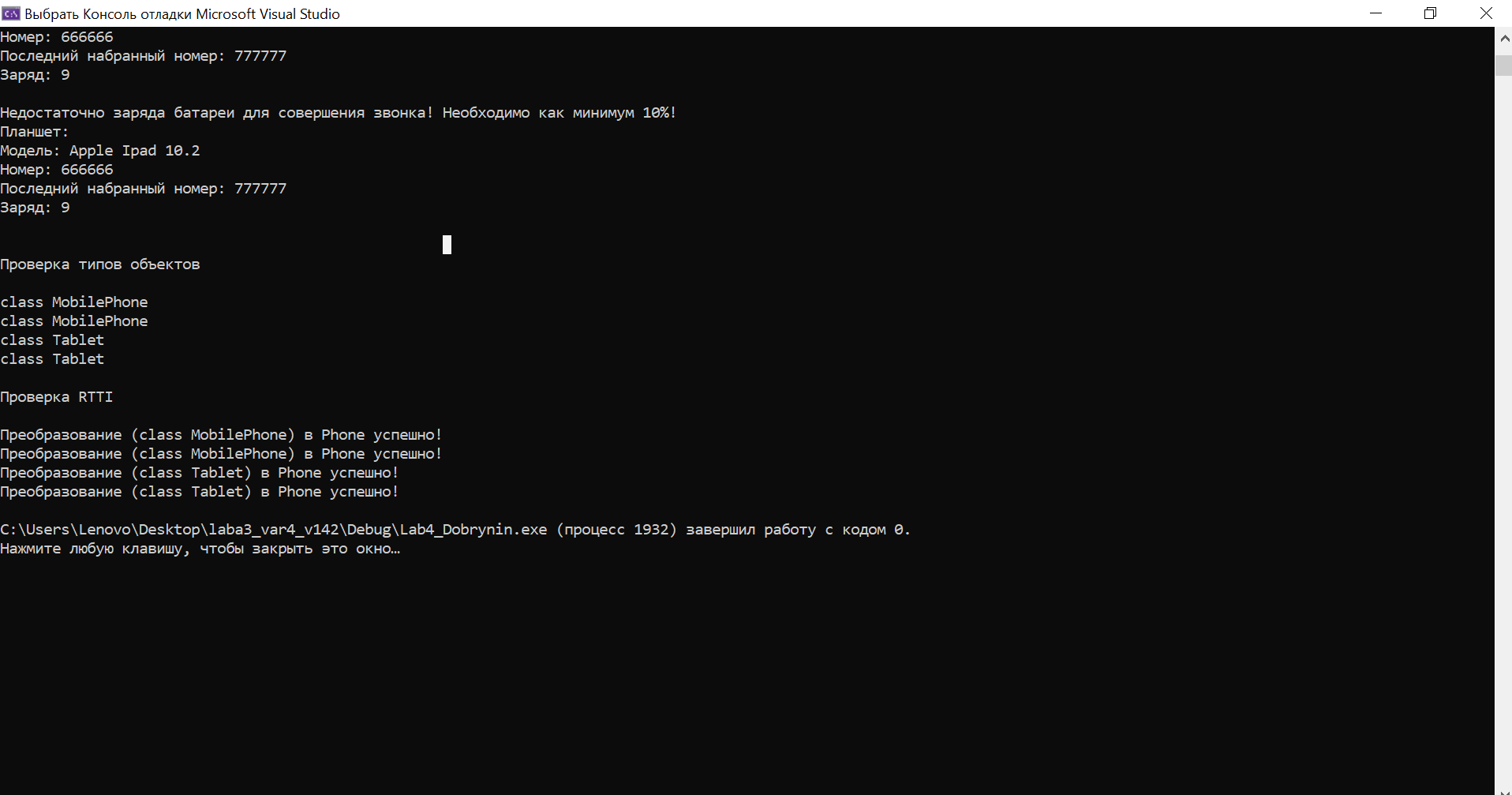
Интерфейс возможности звонка ICallable с методом void call(const std::string& recepient) – позвонить по номеру recepient. Реализация метода в классе телефона: проверка корректности номера телефона (содержит только цифры), снятие суммы со счета и расходование 3% заряда. Реализация метода в классе планшета: израсходовать 10% заряда (звонок идет по скайпу). В main() создать 2 телефона и 2 планшета, продемонстрировать полиморфизм call().

СКРИНШОТ ОТРАБОТАВШЕЙ ПРОГРАММЫ









Вывод: Я научился реализовывать на С++ наследование классов, программировать абстрактные классы и интерфейсы, виртуальные методы, а также динамически определять тип объекта во время выполнения программы.

Контрольные вопросы

1. Перечислите механизмы повторного использования кода в ООП.

2. Что такое наследование? Приведите три примера схем наследования классов.

3. Типы и виды наследования в С++.

4. В чем заключается разница между ранним и поздним связыванием?

5. Что такое полиморфизм? Виртуальные методы.

6. Что такое абстрактный класс? Чисто виртуальные методы.

7. Что из себя представляет класс-интерфейс? Для чего он используется?

8. Порядок создания и удаления подклассов. Зачем нужен виртуальный деструктор?

9. Что означает RTTI? Какие есть возможности RTTI в С++?

ФОТОГРАФИИ ОТВЕТОВ НА КОНТРОЛЬНЫЕ ВОПРОСЫ.

