ПРЕДМЕТНОЕ ОБУЧЕНИЕ

*Шульженко Виталий Игоревич,*
*Эмман Павел Александрович*

# ИНСТРУМЕНТ ДЛЯ ИЗУЧЕНИЯ МЕТОДА РЕЗОЛЮЦИЙ

*В данной статье описывается программа, с помощью которой студент, изучающий метод резолюций, сможет рассмотреть его на примере любого множества логических формул любой интерпретации.*

## ОСОБЕННОСТИ РАЗРАБОТКИ ПРОГРАММЫ

Для разработки программы был выбран язык Java как наиболее простой для реализации и переносимый. Для возможности использования приложения на различных платформах оформление ядра разработки было сделано в виде апплета на HTML-страничке.

Для реализации выражения был спроектирован и реализован класс *LogicExpression*. Выражение хранится в виде строки *String expression*. При разборе выражения переменные записываются в *Vector variables*, а значения переменных в *Map vars*.

Для создания объекта данного класса предоставлен конструктор *public LogicExpression (String expression)*, параметром которого может быть любое скобочное логическое выражение с операторами: !, |, &, +, =, > (где ! – отрицание, | – дизъюнкция, & – конъюнкция, + – исключающее ИЛИ, = – эквиваленция, > – импликация). Работа конструктора заключается в разборе всего выражения и заполнении структур *variables* и *vars*.

*Для проверки выражения была спроектирована отдельная функция...*

Для проверки выражения была спроектирована отдельная функция, выбрасывающая собственное исключение при наличии ошибки. Данная проверка реализована вычислением выражения. Если выражение удалось вычислить, значит, синтаксических ошибок нет. Если при вычислении выражения встречается какая-либо синтаксическая ошибка, – выбрасывается исключение. По этому исключению можно определить, что за ошибка встретилась при разборе выражения. Ошибками могут быть неверный баланс скобок (при этом сообщается, какой скобки не хватает, – закрывающей или открывающей), отсутствие операнда после операнда и другие.

Для вычисления истинности выражения была реализована функция *getExpressionValue()*, основой работы которой является обход выражения, включая вложенные скобки, и последовательное вычис-

ление каждого операнда. Используя данную функцию, можно легко обходить самые сложные выражения с немалым количеством переменных и глубокой вложенностью скобок. Эту функцию удобно использовать при построении таблиц истинности.

Непосредственно для реализации метода резолюций были спроектированы функции построения совершенной дезъюнктивной нормальной формы (СДНФ) и совершенной конъюнктивной нормальной формы (СКНФ). Суть методов – обход всех возможных значений логического выражения. Для каждого истинного (или ложного для СКНФ) значения выражения сохраняем набор переменных, затем соответствующим образом объединяем этот набор и составляем необходимую форму исходного выражения.

Так же реализованы функции проверки выражения – находится ли оно в форме СКНФ или СДНФ.

Для наглядного представления задачи был создан класс *ResolutionTask*, который инкапсулирует логическую задачу, решаемую методом резолюций. Этот класс адекватно влияет на добавление и удаление атомов и высказываний, реализует интерфейс *Serializable*, таким образом, может быть записан в файл.

В данной реализации апплета не был предоставлен интерфейс для записи и чтения задач, что было бы более эффективно для работы на практических занятиях. Но, благодаря интуитивному интерфейсу, ввод данных для задачи занимает незначительное время.

## ВИЗУАЛИЗАЦИЯ РАБОТЫ МЕТОДА РЕЗОЛЮЦИЙ

При визуализации необходимо сохранить функционал реализованного математического аппарата и предоставить пользователю интуитивно понятный и наглядный интерфейс.

На первом этапе пользователю необходимо ввести условие задачи, используемые высказывания (вместе со словесным описанием каждого используемого в высказываниях атома) и правила вывода. Далее пользователю будет предоставлена возможность работать непосредственно только с этими данными. Уже на этом этапе проверяется условие задачи на предмет корректности условия. Если из предложенного набора высказываний нельзя определить их истинность, апплет запретит переход к работе с условиями и укажет на это пользователю.

На втором этапе работы пользователь может работать с данными высказываниями. Доступна возможность изменения конкретного выражения (кнопка «Изменить высказывание»). На основе двух высказываний добавить новое (необходимо выделить два высказывания и нажать на кнопку «Добавить резольвенту», а в предложенном диалоге набрать соответствующее выражение), преобразовать произвольное высказывание в СКНФ (необходимо выделить высказывание и нажать на кнопку «Преобразовать в СКНФ»), а также прочитать условие задачи, соответствие операндов и их словесной интерпретации, просмотреть исходные данные задачи по нажатию кнопки «Условие задачи...».

## ПРИМЕР РАБОТЫ С ПРОГРАММОЙ

Выполним преобразование для простейшего примера (см. рисунок 1).

*На основе двух высказываний добавить новое ...*

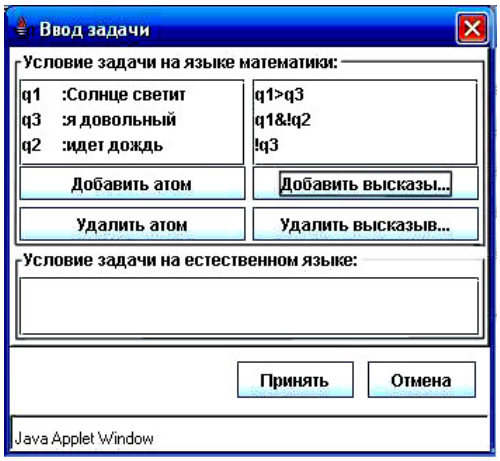



**Рисунок 1.**

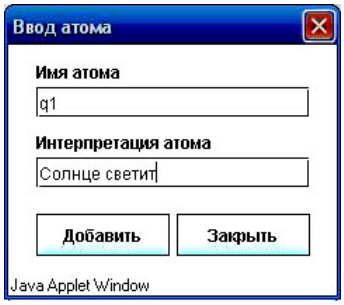



**Рисунок 2.**


**Рисунок 3.**

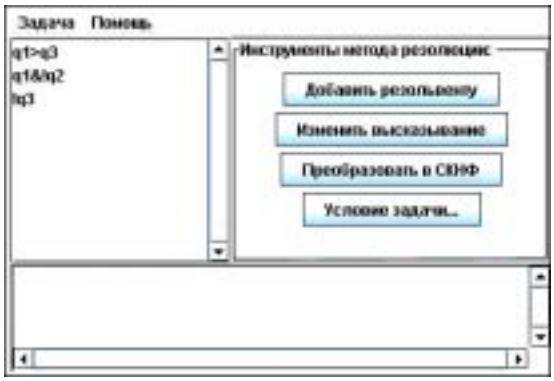



**Рисунок 4.**

Можно добавить атом (по нажатию на кнопке «Добавить атом» – см. рисунок 2) или удить атом, выделив его и нажав кнопку «Удалить атом». Введем атомы: «$Q_1$ – солнце светит», «$Q_2$ – идет дождь», «$Q_3$ – я довольный».

Для ввода высказываний по кнопке «Добавить высказывание», необходимо ввести содержательное высказывание. Для этого предусмотрены два режима: ввод с клавиатуры – тогда доступно поле для ввода, и можно заполнить предложенное поле; ввод с панели – тогда необходимо воспользоваться предоставленными клавишами для составления выражения (см. рисунок 3). Добавим выражения: «$Q_1$ => $Q_3$» – «Солнце светит, следовательно, я довольный», «$Q_1$ & $!Q_2$» – «Сейчас солнце светит и не идет дождь». Докажем, что «я довольный»» – истина.

Тогда: исходными высказываниями для метода резолюций будут «$Q_1$ => $Q_3$»; «$Q_1$ & $!Q_2$»; «$!Q_3$» (см. рисунок 4).

Преобразуем все эти высказывания в СКНФ. Для этого необходимо выделить высказывание и нажать на кнопку «Преобразовать в СКНФ». Получим: «$!Q_1|Q_3$»; «$!Q_1|Q_2$»; «$Q_1||Q_2$»; «$Q_1|Q_2$»; «$!Q_3$» (см. рисунок 5). Если не выделить высказывание, апплет сообщит об этом соответствующим сообщением (см. рисунок 6).

Если этого не сделать, программа при попытке добавления новых резольвент

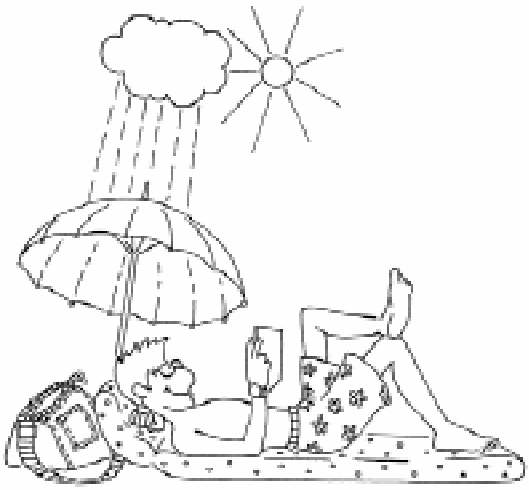
*Для вычисления истинности выражения была реализована функция getExpressionValue...*

выведет сообщение: «Подсказка: Оба выражения должны иметь форму конъюнкта в дизъюнктной нормальной форме (ДНФ). Проще говоря, должны иметь только следующие операции: НЕ[!],ИЛИ[|].»

Из «$Q_1$|!$Q_2$»; «$Q_1$|$Q_2$» следует «$Q_1$» – поэтому выбираем две эти резольвенты, нажимаем кнопку «Добавить резольвенту» и вводим «$Q_1$». Резольвента будет добавлена, а в окошке появится объяснение данного преобразования в терминах задачи: «Можно интерпретировать резольвенту так: Известно, что солнце светит ИЛИ НЕ идет дождь и вместе с этим солнце светит ИЛИ идет дождь, СЛЕДОВАТЕЛЬНО, солнце светит». Если пользователь не выберет два выражения либо выберет более двух, апплет сообщит об этом, как показано на рисунке 7.

Из «$Q_1$» и «!$Q_1$|$Q_2$» следует «$Q_3$» – добавляем ее таким же образом.

Теперь можно выделить «$Q_3$» и «!$Q_3$» – и на их основе добавить «тождественная ложь» (для этого есть кнопка «тождественная ложь» на панели кнопок, либо можно ввести «**F**» в поле ввода – см. рисунок 8). Так как тождественная ложь является следствием этого выражения, что означает окончание работы метода, программа выдаст соответствующее сообщение: «Поздравляем! Вы успешно справились с этой задачей!» (см. рисунок 9).



*Известно, что солнце светит ИЛИ НЕ идет дождь и вместе с этим солнце светит ИЛИ идет дождь...*

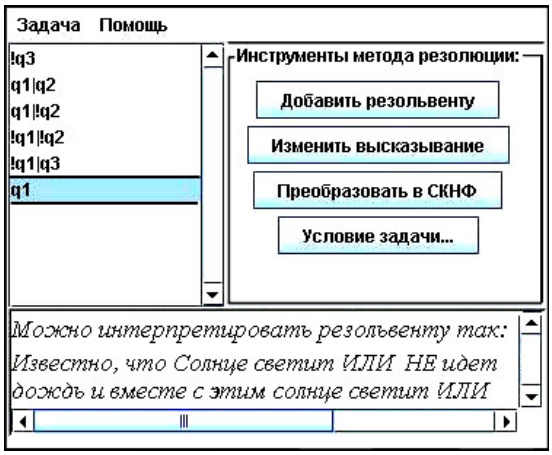



**Рисунок 5.**

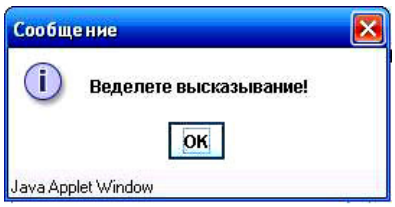



**Рисунок 6.**

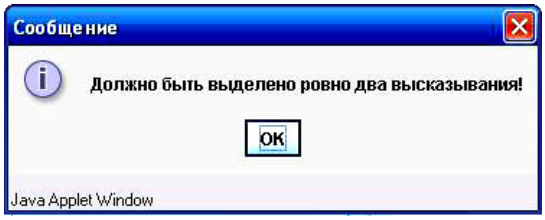



**Рисунок 7.**

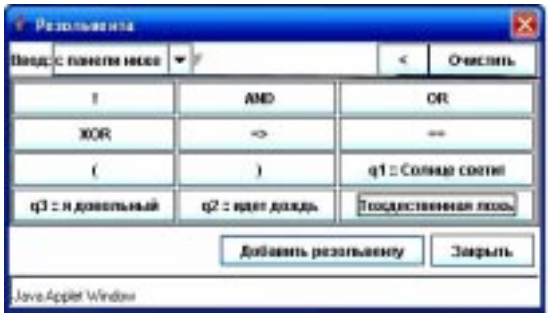



**Рисунок 8.**

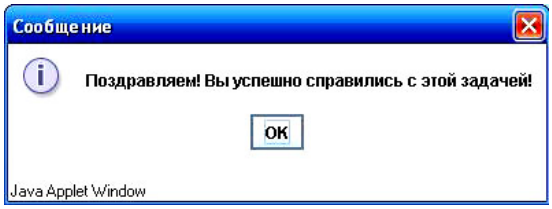



**Рисунок 9.**

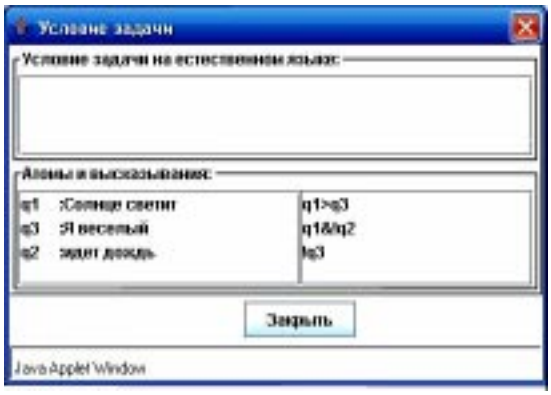



**Рисунок 10.**

**Рисунок 11.**

На любом этапе вывода выражений пользователь может просмотреть условие задачи и ее исходные данные, для этого необходимо нажать на кнопку «Условие задачи». Появится форма, похожая на представленную на рисунке 10.

## УСТАНОВКА И НАСТРОЙКА ПРОГРАММЫ. СИСТЕМНЫЕ ТРЕБОВАНИЯ

Для установки программы, необходимо разархивировать содержимое дистрибутива.

В папке *app* находится сборка для запуска программы как приложения. Для этого необходимо запустить *app\WinLunch.bat* (в Windows) либо набрать в командной строке *java gui/ResolutionFrame*. Запустится приложение, аналогичное описанному апплету (см. рисунок 11).

Для корректного запуска приложения необходимо наличие Java машины не ниже 1.5.

Для запуска апплета необходимо из папки *www* открыть файл *rm.html* в любом браузере с поддержкой java. Если Ваш браузер поддерживает Java машину, то на открытой страничке будет активен апплет, описанный в предыдущем разделе.

Браузеры: Mozilla Firefox 2.0.0.4, Opera 9.23, Internet Explorer 5.5. Для всех браузеров должны быть установлены поддерживаемые Java машины, версии не ниже 1,5.

В папке «resolution workspace» находится проект для среды разработки Eclipse и необходимые исходные тексты для компиляции и сборки приложения.

## ОБЛАСТИ ПРИМЕНЕНИЯ ПРОГРАММНОГО ПРОДУКТА. УСОВЕРШЕНСТВОВАНИЯ

Мы хотели разработать качественное математическое ядро для работы с логикой предикатов. Реализованные классы можно использовать для вычисления истинности булевских выражений, для построения таблиц истинности, для реализации задач логики высказываний, для построения СКНФ и СДНФ для любого выражения. Благодаря простоте интерфейсных методов достаточно просто расширять их функциональность. Использование одного класса для полного описания выражений позволяет сохранять их в файл и восстанавливать из файлов.



***Шульженко Виталий Игоревич, студент 4 курса факультета компьютерных технологий и информатики СПбГЭТУ «ЛЭТИ»,***

***Эмман Павел Александрович, студент 4 курса факультета компьютерных технологий и информатики СПбГЭТУ «ЛЭТИ».***