

ПОВЕРХ (НАУЧНЫХ) БАРЬЕРОВ

*Хассельстрём К., Ослунн Ю.  
(перевод Б. В. Орехова)*

**Язык программирования Шекспир**

Как и в любой большой области, в программировании помимо мейнстрима есть свой фольклор, профессиональные развлечения и экзотика. Излишне прагматическая суть этой дисциплины порождает неизбежный карнавал, хотя и в довольно специфических формах. Одной из них является категория «эзотерических языков программирования», то есть таких, которые, в отличие от «нормальных» (например, C++, Perl, Python, Ruby, Assembler), не предназначены для решения практических задач, а интересны как эксперимент и забава.

Незаслуженную, на наш взгляд, популярность в этом ряду имеет язык Brainfuck, исходный код которого состоит из команд, обозначаемых одним символом («<», «>» или «|»»), и поэтому трудночитаем. Собственно, на этом его особенности заканчиваются. Совсем иное — язык программирования Шекспир (the Shakespeare programming language), программы на котором должны быть написаны так, чтобы напоминать «читателю» пьесы Шекспира в том виде, в котором их представляют себе создатели языка, скандинавские программисты Калле Хассельстрём и Юн Ослунн. Ниже мы помещаем наш перевод документации этого языка, взятый со страницы <http://shakespearelang.sourceforge.net/>, а также перевод программ-«пьес», написанных на этом языке для примера его разработчиками. Из перевода выпущен технический раздел, поясняющий, как нужно компилировать (то есть готовить к выполнению) программы. Перевод «пьес» снабжён комментариями, поясняющими их функционал.

Главный интерес такой публикации, на наш взгляд, в любопытном примере взгляда на устройство художественного текста с точки зрения не профессионального филолога и не его вечного антагониста — наивного читателя, а представителя другой весьма специфической области знания.

Б. В. Орехов

Язык программирования, созданный с целью, чтобы исходный код напоминал пьесы Шекспира.

Персонажи пьесы являются переменными. Если вы хотите присвоить персонажу, например, Гамлету, отрицательное значение, вы выводите его и другого персонажа на сцену и даёте другому оскорбить Гамлета.

Ввод и вывод реализованы через то, что кто-то велит персонажам прислушаться к сердцу или выразить своё мнение. Язык содержит условные конструкции, когда персонажи задают друг другу вопросы, и переходы, когда они решают перейти к соответствующему акту или сцене. Персонажи также выступают как стек, в который можно добавить значение и можно его оттуда удалить.

## Введение

Однажды поздно ночью в феврале Калле Хассельстрём и Юн Ослунн (Kalle Hasselström, Jon Åslund, это мы, авторы) сидели над программистской задачей, которую должны были представить в девять часов утра следующего дня. Это была задача номер четыре из нашего курса синтаксического анализа, и мы уже порядком устали от всего этого. С другой стороны, последняя задача была более забавной, потому что можно делать все что угодно, сколь угодно долго, коль скоро это связано с лексическим и синтаксическим анализом. Итак, вместо того, чтобы завершить четвёртое задание, мы стали придумывать отличные идеи для пятого — такое, которое начинаешь придумывать только тогда, когда пора бы уже идти спать.

Несколькими неделями ранее мы открыли для себя несколько замечательных языков программирования, таких как Java2k<sup>1</sup>, Sorted!<sup>2</sup>, Brainfuck<sup>3</sup> и Malbolge<sup>4</sup>, и нам хотелось сделать свой собственный. Неизвестно почему, но той ночью мы также думали про Шекспира вообще и про оскорбления у Шекспира в частности и тремя часами позже мы пришли к невероятной идее: the Shakespeare Programming Language, SPL.

Это — документация языка и история его создания.

## Задачи

Задачей было создать язык с прекрасным исходным кодом, который бы напоминал шекспировские пьесы. Здесь нет сложных типов данных и управляющих структур. Только арифметические действия и переходы. Можно сказать, что мы соединили выразительность BASIC и удобство ассемблера.

Этот курс был о синтаксическом анализе, а не о создании компилятора. Соответственно, мы не сделали компилятора для SPL, только конвертер SPL в C. Это доказывает, если говорить предельно просто, что SPL может быть прямо транслирован в C, по одному утверждению за раз.

## Hello World!

Поскольку нам не хотелось бы нарушать эту древнюю традицию, давайте начнём с простого примера: программа Hello World. Хотя в другом случае могло бы показаться, что единственной целью этой программы является вывод строки "Hello World!". Она находится в файле `hello.spl`, а также в приложении. Если вы хотите запустить её, обратитесь к соответствующему разделу<sup>5</sup>.

Давайте разберём программу и посмотрим, как она работает.

## Заглавие

Первая строка любой программы на SPL — это заглавие. Или, точнее, всё, что до первой точки, это заглавие, будь то одна строка, три строки или полстроки. Вы можете вставлять пробелы и переводы строк, если хотите, чтобы они были в коде, но мы *настоятельно советуем* со вкусом сделать отступ.

Заглавие служит только для эстетики. С точки зрения парсера, это комментарий.

<sup>1</sup> <http://www.p-nand-q.com/java2k.htm>

<sup>2</sup> <http://www.p-nand-q.com/sorted.htm>

<sup>3</sup> <http://www.catseye.mb.ca/esoteric/bf/>

<sup>4</sup> <http://www.mines.edu/students/b/bolmstea/malbolge/>

<sup>5</sup> В данном переводе отсутствует. Упомянутые в тексте файлы можно найти на сайте языка в Интернете: <http://shakespearelang.sourceforge.net/>. — *прим. перев.*

### Действующие лица

Следующие несколько строк — это список всех персонажей пьесы. Считайте их переменными, способными содержать целочисленное значение. Вы должны объявить каждого персонажа, которого намереваетесь использовать, иначе программа не скомпилируется.

Объявление состоит из имени, за которым следует описание персонажа (оно игнорируется парсером). Вы, однако, не можете взять любое имя, нужно использовать имя настоящего шекспировского героя, такое как Ромео, Джульетта или Призрак (преставившийся отец Гамлета).

### Акты и сцены

Задача актов и сцен — разделить пьесу на части меньшего размера. Пьеса состоит из одного или нескольких актов, каждый акт состоит из одной или более сцен, а каждая сцена состоит из строк (где персонажи что-нибудь говорят) и входных и выходных ремарок, которые заставляют персонажей выходить на сцену и покидать её.

Акты и сцены пронумерованы римскими цифрами. Они начинаются со слов «Act» или «Scene», затем идёт номер, а затем — описание того, что происходит в этом акте или в этой сцене. Так же, как и в случае с заглавием и с описанием персонажа, эти описания игнорируются парсером.

Кроме того, что они прекрасны и описательны, акты и сцены также служат метками, к которым можно переходить, используя утверждения перехода. Как бы там ни было, в программе Hello World нет переходов, так что мы поговорим о них позже.

### Входит, уходит, уходят

Чтобы произносить свои реплики, персонажи должны быть на сцене. Персонаж, к которому они обращаются «you» ‘ты’ (или «thou» или любое другое местоимение второго лица), также должно быть на сцене. Но если на сцене есть ещё один персонаж, то непонятно, какой из них имеется в виду. Парсер отнесётся к этому неодобрительно.

Enter ‘входит’, Enter <Exit? — перев.> ‘уходят’ and Exit ‘уходит’ — эти директивы, заставляющие персонажей выходить на сцену и покидать её. За Enter следует список из одного или более персонажей. За Exit следует только один персонаж. Множественное число от Exit — Exit, за которым следует список, в котором будет, как минимум, два персонажа или ни одного, это будет означать, что уходят все.

Директива Enter по отношению к персонажу, который уже находится на сцене, или наоборот, Exit по отношению к тому, кого нет на сцене, приведёт к ошибке выполнения программы.

### Строки

Строки состоят из имени персонажа, двоеточия и одного или нескольких предложений. В программе Hello World использованы только два типа предложений: вывод на экран и утверждения, которые приводят к присвоению другому персонажу определённого значения.

### Константы

Сначала мы объясним, как выражены константы (такие как 17 и 4711).

Каждое существительное — это константа со значением 1 или -1, в зависимости от того, приятное оно или нет. Например, «flower» ‘цветок’ имеет значение 1, потому что цветы милые, а вот «pig» ‘свинья’ имеет значение -1, потому что свиньи грязные (что не препятствует большинству людей их есть). Нейтральные существительные, такие как «tree» ‘дерево’ тоже означают 1.

Если мы предваряем существительное прилагательным, то это умножает его на два. Ещё одно прилагательное — ещё одно умножение на два, и так далее. Этим способом вы можете получить любую степень двух или её отрицательное соответствие. Из этого легко получить какие угодно целочисленные значения, используя базовые арифметические операции, такие как «сумма X и Y», где X и Y сами являются какими угодно целыми числами. Например, «the difference between the square of the difference between my little pony and your big hairy hound and the cube of your sorry little codpiece» ‘разница между квадратом разницы между моим маленьким пони и твоим большим косматым псом и кубом твоего жалкого маленького гольфика’. Замещая простые константы числами, мы получаем «разницу квадрата разницы 2 и 4 и куба от -4». Итак, поскольку разность 2 и 4 будет  $2 - 4 = -2$ , а куб -4 будет  $(-4)^3 = -64$ , это равняется «разнице квадрата -2 и -64». Квадрат -2 будет  $(-2)^2 = 4$ , а разность 4 и -64 будет 60. Итак, «разница между квадратом разницы между моим маленьким пони и твоим большим косматым псом и кубом твоего жалкого маленького гольфика» означает 60.

Итак, вы можете видеть, что такой способ оперировать константами даёт вам намного больше поэтической свободы, чем другие языки программирования.

#### Присвоение значения переменным

Теперь о том, как мы используем эти числа. Давайте рассмотрим два утверждения. «You lying stupid fatherless big smelly half-witted coward!» ‘Ты — лживый, глупый, бесприютный, страшно вонючий, полоумный трус!’ и «You are as stupid as the difference between a handsome rich brave hero and thyself!» ‘Ты так же глуп, как разность между прекрасным богатым храбрым героем и тобой’.

С первым всё просто: за местоимением второго лица следует число. Результат этого утверждения — присвоение значения этого числа (в данном случае -64) персонажу, к которому обращаются. Имеется в виду:  $X = -64$ .

Со вторым всё немного сложнее. Что для программы должно означать «thyself»? Это не существительное, это возвратное местоимение. Его значение — это текущее значение персонажа, к которому обращено высказывание. Таким образом, число во втором утверждении составляет  $8 - X$ , где X — это значение персонажа, к которому обращено высказывание. И, как вы могли бы ожидать благодаря вашему знанию английского, второе утверждение — это просто ещё один случай присвоения значения. Имеется в виду « $X = 8 - X$ ». Быть «таким же плохим, как», «таким же хорошим, как» или таким же [любое прилагательное], как что-то ещё, означает быть равным этому чему-то ещё.

#### Вывод

Ещё один тип предложения, используемый в Hello World — это вывод. Есть два вида предложений вывода: «Open your heart» ‘открой своё сердце’ and «Speak your mind» ‘что на уме твоём, скажи’. Первый заставляет персонажа, к которому обращено высказывание, вывести своё значение в числовом формате, а второй более символичный, выводит сопоставленную этому числу букву, цифру или иной символ, в соответствии с кодировкой, которая используется на вашем компьютере.

В этой программе мы используем только вторую форму. Вся программа — это последовательность получения чисел, записи соответствующего символа, получения следующего числа, записи соответствующего символа и так далее.

### Несколько менее банальный пример

Теперь в качестве менее банального примера — вычисление простых чисел. В файле `primes.spl` и в приложении имеется программа, которая запрашивает у пользователя число и затем печатает все простые числа, которые меньше или равны исходному.

В этой программе есть три вещи, которых мы до сих пор не видели: ввод, переход и условные конструкции.

#### Ввод

Утверждения ввода работают так же, как и утверждения вывода, за тем исключением, что они рассчитаны на чтение, а не на запись. Чтобы считать число, как в этой программе, нужно использовать предложение «Listen to your heart» ‘прислушайся ты к сердцу своему’, для считывания символа нужно использовать «Open your mind» ‘открой ты разум свой’. Значение, как обычно, будет присвоено персонажу, к которому обращено высказывание.

#### Переходы

Предложение вида «Let us return to scene III» ‘давайте вернёмся к сцене III’ означает просто «переход (goto) к сцене III». Вместо «let us» ‘давайте’ можно использовать «we shall» ‘мы’ или «we must» ‘мы должны’, а вместо «return to» ‘вернёмся к’ можно использовать «proceed to» ‘перейдём к’. Если вы указываете сцену, то это будет ссылка на сцену в текущем акте. Не предусмотрено ссылки на конкретную сцену в другом акте, вы должны смириться с возможностью осуществлять переходы только внутри акта.

#### Условные конструкции

Условные конструкции состоят из двух шагов, как это показано на примере фрагмента кода ниже:

Juliet:	Джюльетта:
Am I better than you?	И лучше ль я тебя?
Hamlet:	Гамлет:
If so, let us proceed to scene III.	А если так, да перейдём мы к сцене III.

Сначала кто-то озвучивает вопрос. Это своего рода сравнение, результатом которого будет либо истина, либо ложь. Но подробнее об этом ниже.

Далее следует условное утверждение. Оно создаётся помещением «if so» ‘если это так’ (или «if not» ‘если это не так’) и запятой перед предложением — это предложение будет исполняться только если ответ на последний вопрос будет «да» (или «нет»).

Это очень похоже на то, как вы бы осуществляли условные переходы и тому подобные вещи во многих ассемблероподобных языках.

## Сравнения

Сравнения осуществляются так, как вы бы и могли этого ожидать: «`is X as good as Y`» ‘X так же хорош, как Y’ проверяет на равенство X и Y, которые являются произвольными значениями. Можно заменить «good» ‘хорош’ на любое другое прилагательное. «`is X better than Y`» ‘X лучше Y’ проверяет на  $X > Y$ . Это справедливо для любого положительного сравнения. Если вам нужно  $X < Y$ , используйте негативное сравнение с помощью «worse» ‘хуже’.

## Несколько менее банальный пример

Можно даже сказать, что описанный таким образом язык делает всё, что может любой другой язык программирования, хотя и делает это более пышно, никак не обойти тот факт, что его ёмкость строго лимитирована. Есть только данное число персонажей Шекспира (около сотни из них распознаются парсером) и каждый из них может содержать только целое число конечной длины. Итак, ёмкость лимитирована, а это значит, что SPL может решать только проблемы конечного объёма.

Осознав это, мы добавили в язык возможности запоминания. Мы сейчас их опишем, но сначала взгляните на то, как их можно использовать. Программа в файле `reverse.spl`, которая также может быть найдена в приложении — читает любое количество символов и затем выдаёт их в обратной последовательности.

## Запоминание

Персонажи в языке программирования Шекспир не простофили, которые могут запомнить только одно число. Как и нормальные люди, они могут запоминать несколько. Основываясь на современных экспериментальных исследованиях психологии это реализовано в ячейках памяти.

Каждый персонаж может поместить целое число в свою память и в дальнейшем извлечь его. Занесение в память реализовано так:

Lady Macbeth:  
Remember me.

Леди Макбет:  
Помни обо мне.

Это заставляет, разумеется, того, к кому обращается леди Макбет, занести значение леди Макбет в свою память. Извлечение ещё проще:

Lady Macbeth:  
Recall your imminent death!

Леди Макбет:  
О неизбежной смерти вспомни ты!

Единственное значимое слово здесь — это «recall» ‘вспомни’, всё остальное — только художественная набивка. Этот кусок кода заставляет того, к кому обращена реплика леди Макбет, извлечь число из памяти и присваивает ему или ей это число.

Попытка извлечь значение из пустой ячейки — верный знак того, что автор пока не довёл до совершенства свои навыки рассказчика и жестоко разочарует систему.

<...>

## Примеры

Hello World<sup>1</sup>

Бесславная программа Hello World!

Ромео, молодой человек, отличающийся терпеливостью.  
Джульетта, также молодая девушка, отличающаяся изяществом.  
Офелия, замечательная девушка, много спорящая с Гамлетом.  
Гамлет, льстец из АО «Очернители Андерсена»

Акт I: Очернение и лесть Гамлета.

Сцена I: Очернение Ромео.

[Входят Гамлет и Ромео]

Гамлет:

Ты лживый глупый, бесприютный, вонючий страшно, полоумный трус!

Ты так же глуп, как разница между прекрасным, богатым храбрецом-героем и тобой самим! Что на уме твоём?

Ты так же храбр, как сумма твоих жирных, маленьких, набитых, грязных, дурных, старых, прогнивших гульфиков и тёплым, прекрасным, честным, солнечным и мирным летним днём. Ты так же здоров, как разница между сложеньем милейшей алой розы с моим отцом и самим тобою. Что на уме твоём?

Ты так же малодушен, как сложенье тебя же самого и разницей большого могучего и гордого престола с конём. Что на уме твоём?

Что на уме твоём?

[Ромео уходит]

Сцена II: Восхваление Джульетты.

[Входит Джульетта]

Гамлет:

Ты так мила, как съединенье сочетанья Ромео и его коня с его как смоль котом. Так вырази себя!

[Джульетта уходит]

Сцена III: Восхваление Офелии.

[Входит Офелия]

Гамлет:

Ты так прелестна, как произведенье широкого местечка на отшибе и моего невероятного, бездонного и баснословного богатства. Так вырази себя!

[Гамлет и Офелия уходят]

---

<sup>1</sup> Программы с таким названием традиционно являются первыми учебными программами для всех языков программирования. Они обычно предельно просты и выполняют только одно действие: вывод на экран сообщения «Hello World» ‘здравствуй, мир!’ — *прим. перев.*

Акт II: За спиной у Гамлета.

Сцена I: Разговор Ромео и Джульетты.

[Входят Ромео и Джульетта]

Ромео:

Откройся мне. Ты так встревожена, как если бы сложить тебя саму и разницу приглаженного маленького хомяка и носа моего. Откройся мне!

Джульетта:

Откройся ТЫ мне! Ты так же плох, как Гамлет! Ты мал, как разница между квадратом разницы между моею крохотной лошадкой и твоим большим косматым псом и кубом твоего несчастнейшего крохи-гульфика. Откройся мне!

[Ромео уходит]

Сцена II: Разговор Джульетты и Офелии.

[Входит Офелия]

Джульетта:

Ты так же хороша, как частное Ромео и сложенья небольшого мохнатого зверька и кровопийцы. Что на уме твоём?

Офелия:

Ты омерзительна, как частное Ромео и дважды разница омелы и болезненного сочащегося волдыря. Что на уме твоём?

[Уходят]

## Простые числа<sup>1</sup>

Расчёт простых чисел в Копенгагене

Ромео, юный веронец.

Джульетта, молодая девушка.

Гамлет, временная переменная из Дании.

Призрак, ограничивающий фактор (а по удивительному совпадению ещё и отец Гамлета)

Акт I: Беседа с потусторонним.

Сцена I: В последний час перед рассветом.

[Входят Джульетта и Призрак]

---

<sup>1</sup>Простым называется число, которое делится без остатка только на два числа: на единицу и на само себя. Это, например, числа 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37 и так далее. Нахождение простых чисел — одна из простейших учебных задач программиста, имеющего дело с новым для себя языком. Эта программа определяет и выводит на экран все простые числа в диапазоне от 1 до обрабатываемого. — *прим. перев.*

Призрак:

Ты небольшая тёплая красивая накидка. Ты так скоро, как разница квадрата твоего и золотых твоих волос. Говори.

Джульетта:

Прислушайся же к сердцу!

[Призрак уходит]

[Входит Ромео]

Джульетта:

Ты так любезен, словно ясный летний день!

Акт II: Определяя делимость.

Сцена I: Интимный разговор.

Джульетта:

Искусней ты, чем Призрак?

Ромео:

А если так, то перейдём мы к сцене V.

[Ромео уходит]

[Входит Гамлет]

Джульетта:

Ты столь же гнусен, как квадрат корня Ромео!

Гамлет:

Как роза алая, прекрасна ты.

Сцена II: Вопросы и их последствия.

Джульетта:

И лучше ль я тебя?

Гамлет:

А если так, то перейдём мы к сцене III.

Джульетта:

Остаток частного Ромео и меня так же хорош, как пустота?

Гамлет:

А если так, пройдем к IV сцене.

Ты так же дерзновенна, как и сумма тебя самой и римлянина.

Джульетта:

Вернёмся к сцене II.

Сцена III: Ромео должен умереть!

[Гамлет уходит]

[Входит Ромео]

Джульетта:  
Открой своё ты сердце.

[Джульетта уходит]

[Входит Гамлет]

Ромео:  
Ты так же гнил, как разница пустого места и сложенья противного, зло-  
вонного, придурка-борова и жабы небольшой. Что на уме твоём?

[Ромео уходит]

[Входит Джульетта]

Сцена IV: По одному пёсику за раз.

[Гамлет уходит]

[Входит Ромео]

Джульетта:  
Ты благороден так же, как сложенье тебя же самого и моего чихуахуа!  
Вернёмся мы к сцене I.

Сцена V: Конец.

[Уходят]

## Обратный порядок<sup>1</sup>

Выводя входящее наоборот

Отелло, человек стека.  
Леди Макбет, которая кладёт ему, пока он не отдаст.

Акт I: Один, совсем один.  
Сцена I: В начале ничего не было.

[Входят Отелло и леди Макбет]

---

<sup>1</sup> Эта программа принимает от пользователя некоторую последовательность символов и выводит её на экран в обратном порядке. — *прим. перев.*

Отелло:  
Ты есть ничто!

Сцена II: Кладя в самый конец.

Леди Макбет:  
Открой свой разум миру ты! И помни о себе.

Отелло:  
Ты так жестка, как сумма от тебя самой и каменной стены. Я так же от-  
вратителен, как ветреник?

Леди Макбет:  
А если нет, вернёмся к сцене II. О неизбежной смерти вспомни ты!

Отелло:  
Ты так ничтожна, как и разница между тобой самой и волосами.

Сцена III: Коль ты извлёк на свет, то передышки нет!

Леди Макбет:  
О, вспомни о своём несчастном детстве! Что на уме твоём?

Отелло:  
Ты так гнусна, как сумма от тебя самой и жабы! Разве ты лучше, чем ни-  
что?

Леди Макбет:  
А если так, то мы вернёмся к сцене III.

Сцена IV: Конец.

[Уходят]

### *Бьёрн Стенберг и Линус Нильсен Фельтцинг*

Драма Фибоначчи<sup>1</sup>

Коль скоро захотите вы, чтоб вняли мы вам, то  
Всё сделайте, как ниже о том сказано. Пошлите  
Почтовое вы отправление по этим адресам:  
bjorn(at)haxx(dot)se or linus(at)haxx(dot)se

Гамлет, маяк истины в мире сумрачного беспорядка.  
Джюльетта, бедная девушка, начинающая с самого низа и добивающаяся  
возвышения.

---

<sup>1</sup>Эта программа отсутствует в документации к языку Шекспир, поскольку написана не его создателями, а другими программистами-энтузиастами. Числа Фибоначчи — это элементы последовательности, в которой каждое последующее число равно сумме двух предыдущих. Например: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89. Данная программа выводит такую последовательность до числа 701 408 733.

Ромео, связанной у Гамлета и Джульетты.  
Офелия, прелестная женщина, переводящая наших героев на новый уровень.  
Макбет, бедная оскорблённая женщина, которая появляется совсем ненадолго.

Акт I: Поиск Фибоначчи.  
Сцена I: Колкость Макбет.

[Входят Гамлет и Макбет]

Макбет:  
Ты есть ничто. Открой же миру сердце!

Гамлет:  
Ты столь же гниlostна, как отвратительная, грязная и злая, жирная, бездомная с ворсинками испачканными, страшная, больная, лживая свинья.

Макбет:  
Ты подл, как разница между ничем и суммой чванливой, смрадной, половумной свиноматки и жабы крохотной. Что на уме твоём?

Гамлет:  
Ты так придурковата, как куб разницы между пустотой и тобой самой.

[Макбет уходит]

[Входит Офелия]

Гамлет:  
Ты так же благородна, как и я!

[Офелия уходит]

[Входит Ромео]

Ведь ты герой! Открой же миру сердце.

Ромео:  
Говори!

Сцена II: Напряжение нарастает.

Ромео:  
Ты так же гнусен, как и сумма Джульетты и меня. А Макбет хуже ли тебя? Но если так, то к сцене V должны мы перейти.  
Открой же сердце!

[Гамлет уходит]

Сцена III: Новые уровни мужества.

[Входит Офелия]

Ромео:  
Говори!

[Уходят]

Сцена III: Джульетта передумывает.

[Входит Джульетта и Гамлет]

Гамлет:  
Ты так же хороша, как и Ромео!

[Гамлет уходит]

[Входит Ромео]

Джульетта:  
Ты столько же коварен, как и Гамлет.

[Джульетта уходит]

[Входит Гамлет]

Гамлет:  
Вернёмся к сцене II.

Сцена V: Конец.

[Уходят]