компьютер в учебном процессе

Абрамян Михаил Эдуардович

УДК 004.42+37.09

РЕАЛИЗАЦИЯ ЭЛЕКТРОННОГО ЗАДАЧНИКА ПО СТРОКОВЫМ АЛГОРИТМАМ БИОИНФОРМАТИКИ[1]

Аннотация

В статье рассматриваются возможности использования электронных задачников при изучении сложных алгоритмов и описывается электронный задачник по строковым алгоритмам биоинформатики Programming Taskbook for Bioinformatics. Задачник содержит 160 учебных заданий и охватывает широкий диапазон классических и получисленных алгоритмов поиска подстрок и алгоритмов неточного сравнения строк, в том числе алгоритмов глобального и локального выравнивания и нахождения наибольшей общей подпоследовательности. Приводятся примеры учебных заданий, иллюстрирующие особенности задачника.

Ключевые слова: электронный задачник, строковые алгоритмы поиска и неточного сопоставления, биоинформатика.

1. ОСОБЕННОСТИ ПРИМЕНЕНИЯ ЭЛЕКТРОННЫХ ЗАДАЧНИКОВ ПРИ ИЗУЧЕНИИ СЛОЖНЫХ АЛГОРИТМОВ

Алгоритмы, связанные с поиском вхождений образца в текст и неточным сопоставлением строк, образуют важный класс строковых алгоритмов, активно используемый в настоящее время в различных отраслях научных исследований, в частности, в биоинформатике [1]. Базовые алгоритмы поиска и сопоставления включаются в общие курсы, посвященные анализу алгоритмов [2]. Эти алгоритмы, как и любые другие виды алгоритмов, могут изучаться в нескольких аспектах. С теоретической точки зрения для них требуется обоснование корректности и эффективности; практический аспект состоит в реализации алгоритма на каком-либо языке программирования и использовании полученной программы при анализе различных вариантов исходных данных. В то время как теоретические вопросы излагаются на лекциях, программной реализации посвящаются практикумы, при этом реализация сложных алгоритмов (а таковыми являются все эффективные алгоритмы поиска и сопоставления), как правило, вызывает у студентов серьезные трудности.

Для того чтобы облегчить практическое изучение сложных алгоритмов, целесообразно использовать специализированные программные средства – *электронные задачники* [3]. Электронный задачник представляет собой компьютерную систему, которая вза-

[1] Работа выполнена в рамках ФЦП «Научные и научно-педагогические кадры инновационной России» (госконтракт № 14.740.11.0006).

имодействует с программой студента, выполняя следующие действия:

– передача программе студента набора исходных данных и получение от нее результатов;

– автоматическая проверка результатов путем сравнения с заранее подготовленными контрольными данными;

– дополнительный контроль правильности операций ввода-вывода;

– наглядное отображение всей информации, связанной с учебным заданием.

Беря на себя часть рутинных действий, связанных с вводом-выводом, электронный задачник оставляет на долю студента главное – реализацию алгоритма. При этом разнообразие предлагаемых задачником исходных данных обеспечивает надежное тестирование разработанного алгоритма. Электронные задачники имеют общие черты с программами-роботами, используемыми при автоматической проверке олимпиадных задач (см., например, [4]). Отличие между ними состоит в том, что электронный задачник используется для поддержки *процесса решения задачи*, а не только для финальной проверки уже разработанной программы. Поскольку электронный задачник должен «сопровождать» весь процесс решения задачи, представляется естественной его интеграция в используемую студентом среду разработки.

При изучении сложных алгоритмов применение электронных задачников дает дополнительные преимущества. Это связано со следующим обстоятельством. Как правило, каждый сложный алгоритм состоит из нескольких этапов и часто использует специальные характеристики, связанные с обрабатываемыми данными и требующие особого вычисления. Например, для алгоритмов поиска подстрок выделяется этап препроцессинга – начальной обработки исходных подстрок, в ходе которой определяются их дополнительные характеристики (в частности, для метода Кнута–Морриса–Пратта это значения функции неудач), – и этап собственно поиска, на котором используются ранее найденные характеристики. При программной реализации каждого последующего этапа алгоритма, очевидно, необходимо иметь данные, полученные на предыдущих этапах. Поэтому для ознакомления с очередным этапом алгоритма студент вынужден реализовать все предыдущие этапы, причем любые ошибки, допущенные им при реализации этих этапов, не позволят выполнить основное задание. На стадии начального изучения алгоритма подобная ситуация является нежелательной. Электронный задачник может решить эту проблему путем автоматической генерации данных, получаемых на предыдущих этапах выполнения алгоритма, и последующей передачи их в качестве исходных данных программе студента. Это позволит студенту сразу приступить к реализации того этапа алгоритма, которому посвящено соответствующее задание.

Разумеется, следует предусматривать и итоговые задания, в которых алгоритм надо реализовать в полном объеме; эти задания не будут вызывать у студента особых трудностей, если ранее он успешно выполнил задания, связанные с каждым из этапов изучаемого алгоритма.

Для ознакомления с новыми понятиями и характеристиками также можно предусматривать специальные задания. Автоматическая проверка правильности выполнения подобных заданий (а также вывод правильного варианта ответа в случае ошибочного решения) оказывается особенно полезной, поскольку при первоначальном знакомстве с новыми понятиями студенту сложно обеспечить надежное тестирование алгоритма их вычисления.

Таким образом, применение электронного задачника, который содержит *серии* учебных заданий, посвященных различным этапам реализации сложных алгоритмов, позволяет быстро и эффективно изучить эти алгоритмы на практике.

Настоящая работа посвящена реализации электронного задачника, связанного с изучением строковых алгоритмов биоинформатики. Эта реализация выполнена на базе универсального электронного задачника Programming Taskbook, который кратко описывается в следующем пункте.

2. УНИВЕРСАЛЬНЫЙ ЭЛЕКТРОННЫЙ ЗАДАЧНИК PROGRAMMING TASKBOOK

Задачник **Programming Taskbook** [3] представляет собой программный комплекс, в котором реализованы все описанные в п. 1 возможности, связанные с поддержкой процесса выполнения учебных заданий. Его дополнительной особенностью является универсальность: входящие в его состав задания можно выполнять на разных языках и в различных программных средах. В версии 4.10 задачника (последней на момент подготовки настоящей статьи) поддерживаются языки Pascal, Visual Basic, C++, C#, Visual Basic .NET, Python. Среди программных сред, в которых может использоваться задачник, можно выделить следующие:

– Turbo Delphi 2006;

– Free Pascal Lazarus 0.9;

– PascalABC.NET [5];

– Visual Studio .NET 2003, 2005, 2008, 2010 (языки C++, C#, Visual Basic .NET);

– IDLE for Python 2.5, 2.6, 2.7, 3.2.

Задачник также интегрирован в вебсреду программирования Programming-ABC.NET [6], в которой может использоваться для выполнения учебных заданий на нескольких языках (Pascal, C#, Visual Basic .NET).

Первоначально задачник разрабатывался для поддержки базового курса программирования. Он включает 1100 учебных заданий по основным темам данного курса – от скалярных типов и управляющих операторов до сложных типов данных (массивы, строки, файлы), рекурсивных алгоритмов, динамических структур (линейных и иерархических), – а также 200 заданий, связанных с ЕГЭ по информатике.

Задачник является расширяемым: дополняющий его программный комплекс для преподавателя **Teacher Pack** [7] содержит *конструктор учебных заданий* PT4TaskMaker. Новые группы заданий оформляются в виде динамических библиотек (dll-файлов), что делает их доступными для любых языков и программных сред, поддерживаемых задачником.

Наличие конструктора учебных заданий позволяет использовать задачник Programming Taskbook в качестве платформы для разработки специализированных электронных задачников. В качестве примеров реализации подобных задачников можно указать задачник по параллельному программированию **Programming Taskbook for MPI** [8] и описываемый в настоящей работе задачник по строковым алгоритмам биоинформатики **Programming Taskbook for Bioinformatics** (PT for Bio).

3. ОБЩЕЕ ОПИСАНИЕ ЗАДАЧНИКА PT FOR BIO

Задачник PT for Bio включает две группы: Match (80 заданий на алгоритмы строкового поиска) и Align (80 заданий на алгоритмы неточного сопоставления строк). Задания разработаны с помощью конструктора PT4TaskMaker; детали реализации алгоритмов соответствуют их описанию в книге [1], являющейся в настоящее время наиболее полным руководством по строковым алгоритмам биоинформатики на русском языке.

Будучи разработанным на основе универсального задачника Programming Taskbook, задачник PT for Bio может использоваться для разных языков и в различных программных средах. Реализация любого из алгоритмов, рассмотренных в задачнике PT for Bio, предполагает использование только базовых типов данных (целочисленных, логических, символьных и строковых) и стандартных управляющих конструкций (условные операторы и циклы), поэтому задания, входящие в задачник PT for Bio, могут выполняться на любом языке, поддерживаемом задачником Programming Taskbook (Pascal, Visual Basic, C++, C#, Visual Basic .NET, Python).

Все базовые возможности задачника Programming Taskbook (генерация исходных данных, автоматическая проверка правильности и т. д.), разумеется, сохранены и в задачнике по строковым алгоритмам. Однако задачник PT for Bio имеет и ряд особеннос-

тей, обусловленных спецификой рассматриваемой предметной области:

– формулировки заданий содержат все формулы, необходимые для выполнения задания, и часто снабжаются примечанием, содержащим дополнительную информацию об особенностях рассматриваемого алгоритма или его эффективности;

– строковые данные при их отображении в окне задачника снабжаются специальной линейкой, позволяющей быстро определить номер позиции любого символа в строке, а также быстро найти символ по номеру его позиции;

– в разделе результатов, наряду с собственно результирующими данными, часто выводится дополнительная информация, позволяющая оценить эффективность реализованных алгоритмов в сравнении с ранее изученными алгоритмами или их модификациями.

В соответствии с отмеченными в п. 1 особенностями применения электронных задачников при изучении сложных алгоритмов, с каждым изучаемым алгоритмом связывается серия заданий, включающая задания на ознакомление с новыми понятиями, на освоение каждого этапа алгоритма и, наконец, на реализацию алгоритма в полном объеме.

Для надежной проверки правильности выполнения заданий, связанных с поиском подстрок, используется следующий прием: в качестве результатов поиска требуется вывести не только позиции найденных вхождений образца в текст, но и дополнительные характеристики, используемые в изучаемом алгоритме, а также количество сравнений символов, потребовавшееся при выполнении алгоритма для предложенных исходных данных. Вывод правильных значений для таких дополнительных данных (подобные данные можно назвать *индикаторами*) гарантирует, что задача была решена именно тем алгоритмом поиска, который указан в ее формулировке. В заданиях, связанных с неточным сопоставлением строк, также часто требуется вывести дополнительные данные-индикаторы.

4. СОСТАВ ЗАДАЧНИКА PT FOR BIO

Группа Match охватывает большинство алгоритмов, описанных в первой части книги [1] («Точное совпадение строк: основная задача»); в группе Align рассматриваются алгоритмы, приведенные в первых двух главах ее третьей части («Неточное сопоставление, выстраивание последовательностей и динамическое программирование»). Последовательность изучаемых алгоритмов соответствует, в основном, порядку их изложения в [1]. Таким образом, задачник PT for Bio может рассматриваться как практикум к первой и третьей части данной книги.

В группу Match включены основные классические алгоритмы поиска (в том числе методы Кнута–Морриса–Пратта и Бойера–Мура) и известные получисленные алгоритмы (битовый метод и метод дактилограмм Карпа–Рабина).

Задания группы Match разбиты на следующие подгруппы (в скобках указано число заданий в подгруппе):

– поиск подстрок: базовые понятия и наивный алгоритм (8);

– основной препроцессинг: базовые понятия и быстрый алгоритм (8);

– поиск с использованием основного препроцессинга (4);

– метод Кнута–Морриса–Пратта (5);

– препроцессинг для метода Кнута–Морриса–Пратта (9);

– метод реального времени (4);

– правила сдвига по плохому символу и по хорошему суффиксу (14);

– метод Бойера–Мура (8);

– битовый метод поиска точных и неточных вхождений (9);

– метод дактилограмм Карпа–Рабина (11).

Связи между рассмотренными алгоритмами приведены на рис. 1. Эти связи отражены в формулировках заданий в виде системы ссылок. Наличие подобных ссылок позволяет уменьшить размеры формулировок и наглядно продемонстрировать общие черты родственных алгоритмов. Большинство классических алгоритмов поиска связано с основным алгоритмом препроцессинга; для каждого из получисленных алгорит-

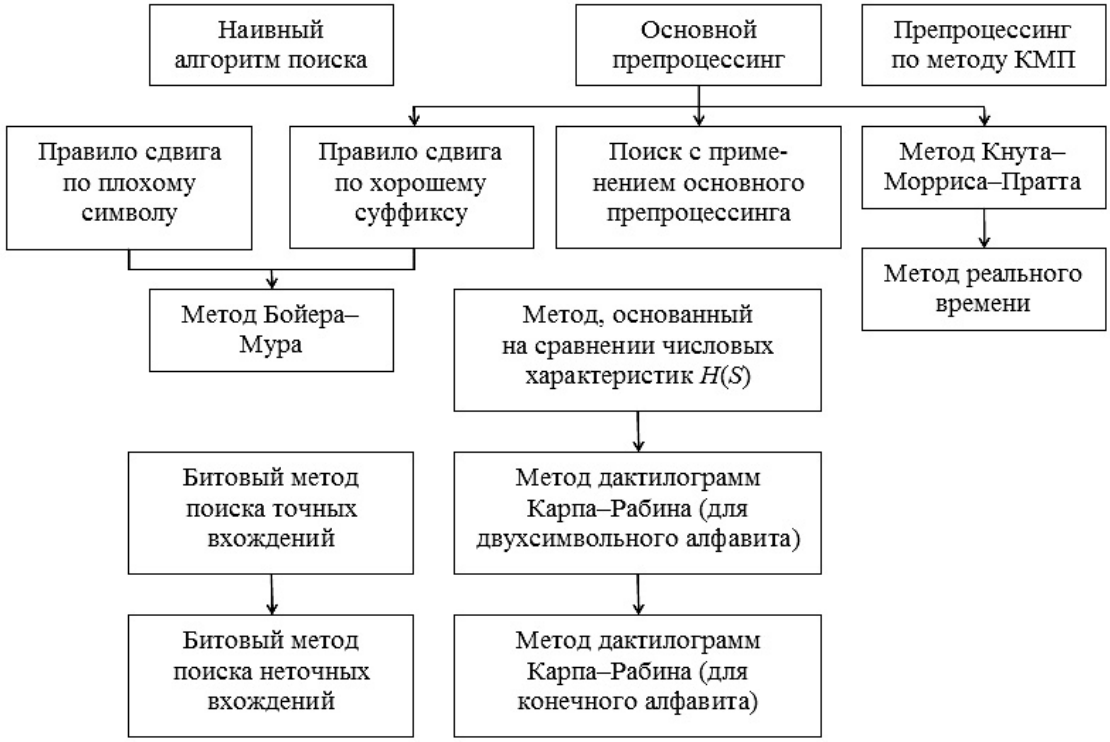

Рис. 1. Связи между алгоритмами группы Match

мов имеется своя цепочка зависимостей. Поясним смысл характеристики $H(S)$, присутствующей на схеме: это число, строковое представление которого в двоичной системе счисления совпадает со строкой S (предполагается, что строка S состоит только из символов «0» и «1»).

Группа Align включает следующие подгруппы:

– редакционное расстояние и оптимальное редакционное предписание (11);

– модификации редакционного расстояния: редакционное расстояние при наличии символов-джокеров и редакционно-взвешенное расстояние (12);

– глобальное выравнивание строк (14);

– модификации глобального выравнивания: выравнивание с бесплатными граничными пробелами и/или с учетом пропусков (15);

– приближенные вхождения строк, локальное выравнивание и нахождение наибольшей общей подстроки (13);

– нахождение наибольшей общей подпоследовательности (the longest common subsequence – LCS): алгоритм на основе выравнивания и комбинаторный алгоритм для двух и трех строк (15).

Зависимости алгоритмов приведены на рис. 2. Здесь можно выделить три набора взаимосвязанных алгоритмов. Первый набор связан с нахождением редакционного расстояния и построением редакционного предписания, второй – с вычислением сходства строк и их выравниванием (различные варианты глобального выравнивания, локальное выравнивание и приближенное сравнение строк). К третьему набору можно отнести варианты комбинаторного алгоритма нахождения LCS, основанные на алгоритмах нахождения наибольшей возрастающей числовой подпоследовательности.

5. ОСОБЕННОСТИ ПРОЦЕССА ВЫПОЛНЕНИЯ ЗАДАНИЙ С ИСПОЛЬЗОВАНИЕМ ЗАДАЧНИКА PT FOR BIO

Для ознакомления с заданиями предназначен специальный программный модуль

Рис. 2. Связи между алгоритмами группы Align

задачника PT4Demo, позволяющий просматривать формулировки заданий в демонстрационном режиме. Имеются два варианта демо-режима. Вариант в виде html-страницы позволяет ознакомиться с формулировками заданий выбранной группы (рис. 3). Как было отмечено выше (см. п. 3), формулировка включает описание алгоритма и может

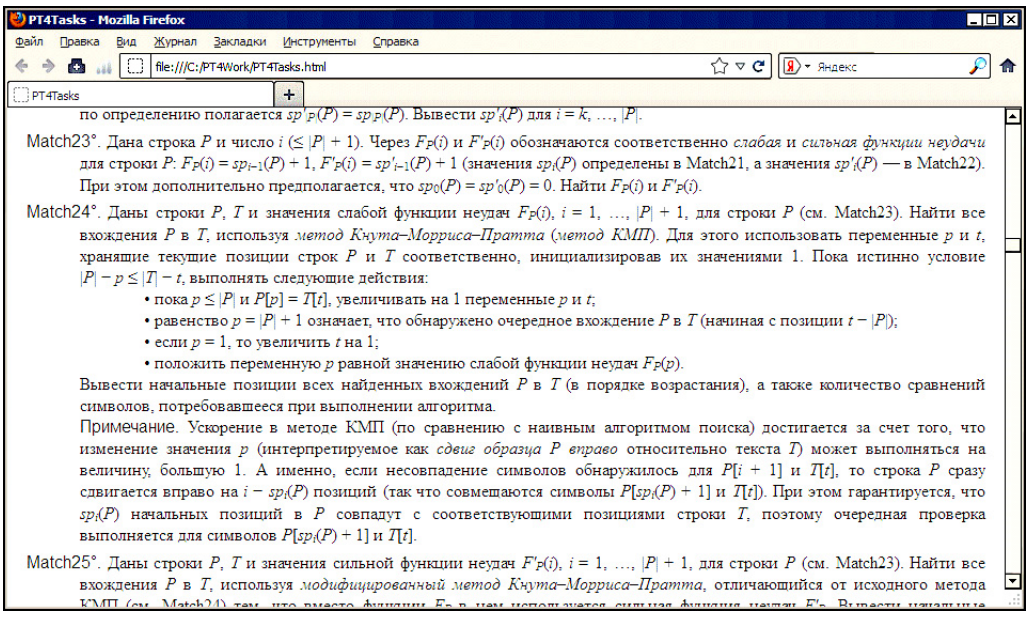

Рис. 3. Окно браузера с описанием задач группы Match

также содержать ссылки на родственные алгоритмы и примечание с дополнительной информацией об особенностях алгоритма.

Можно также просматривать задания в окне задачника (рис. 4); при этом кроме формулировки отображается вариант исходных данных, а также пример правильного решения. Рядом со строковыми данными выводится линейка, позволяющая легко найти символ по его номеру. Задание Match24, приведенное на рисунке, посвящено реализации основного этапа алгоритма Кнута–Морриса–Пратта, поэтому сам задачник предварительно выполняет действия, связанные с этапом препроцессинга, и предоставляет найденные значения функции неудач F_P программе студента в виде исходных данных. Исходные строки генерируются с использованием датчика случайных чисел. Следует также обратить внимание на раздел результатов. В нем требуется вывести не только позиции найденных вхождений образца P в строку T (что можно сделать с помощью различных алгоритмов), но и количество сравнений символов, выполненных в процессе поиска. Понятно, что правильное количество сможет вывести только та программа, в которой запрограммирован алгоритм, описанный в условии задачи.

Приступая к выполнению задания, студент с помощью программного модуля PT4Load может выбрать требуемую среду программирования из числа имеющихся на данном компьютере и создать проект-заготовку для нужного задания. После создания проекта-заготовки модуль запускает выбранную программную среду и загружает в нее созданный проект. Приведем в качестве примера программу-заготовку для выполнения задания Match73 на языке Python (листинг 1). Данная программа импортирует специальный модуль pt4 с функциями задачника и вызывает функцию task, инициализирующую требуемое задание.

Листинг 1

```
# -*- coding: cp1251 -*-
from pt4 import *
task("Match73")
```

Созданную заготовку можно немедленно запустить на выполнение; в результате на экране появится окно задачника (рис. 5). Подобный запуск будет считаться ознакомительным, поскольку программа не выполняет действий, связанных с вводом-выводом данных.

Задание Match73 относится к серии заданий на получисленный метод дактилограмм Карпа–Рабина. В нем требуется вычислить дактилограмму строки, используя специальную модификацию схемы Горнера. Ниже приводится полный текст формулировки этого задания.

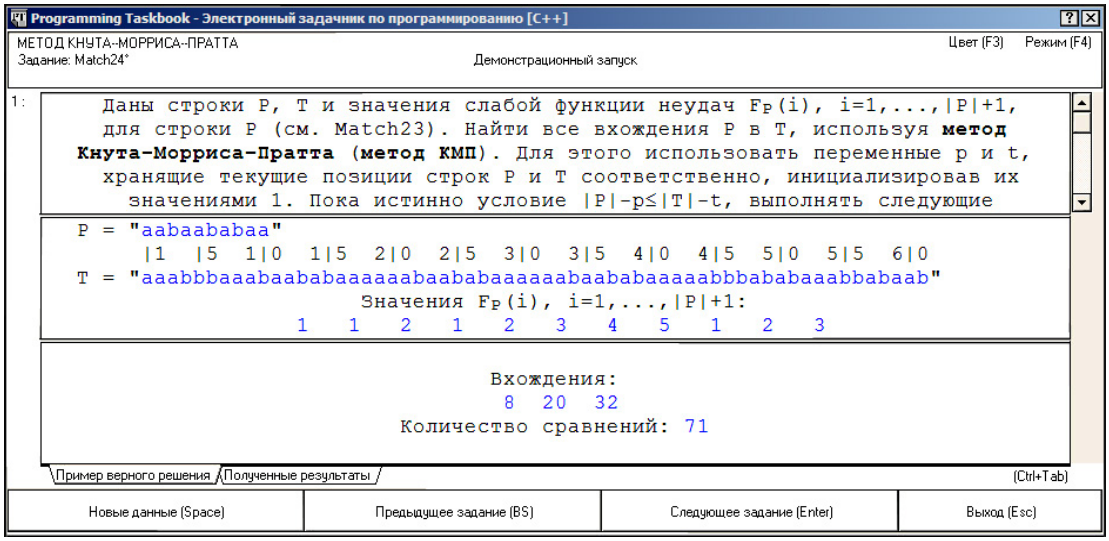

Рис. 4. Окно задачника в демонстрационном режиме

Листинг 2

```
# -*- coding: cp1251 -*-
from pt4 import *
task("Match73")
s = get()
q = get()
res = ord(s[0]) - ord("0")
for i in range(1, len(s)):
    put(res)
    res = res * 2 % q + ord(s[i]) - ord("0")
put(res % q)
```

Match73. Дана строка S, состоящая из символов «0» и «1», и число q. Через $H_q(S)$ обозначается число, равное $H(S) \bmod q$, где число $H(S)$ определено в Match70, а «mod» обозначает операцию взятия остатка от деления нацело. Значение $H_q(S)$ лежит в диапазоне от 0 до $q - 1$ и называется *дактилограммой* строки S. Найти $H_q(S)$, используя вариант схемы Горнера, в котором ни один результат умножения не превосходит $2 \cdot q$:

$$H_q(S) = ((\ldots(((d(S[1]) \cdot 2 \bmod q + d(S[2])) \cdot 2 \bmod q + d(S[3])) \cdot 2 \bmod q + d(S[4])) \ldots) \cdot 2 \bmod q + d(S[|S|])) \bmod q.$$

Вывести промежуточные результаты вычислений, полученные перед каждой операцией умножения на 2, а также найденное значение $H_q(S)$.

В листинге 2 приводится полный текст программы с решением задачи Match73. Следует обратить внимание на то, что для операций ввода-вывода используются специальные методы **get** и **put**, импортируемые из модуля pt4, связанного с электронным задачником. Это обусловлено тем, что исходные данные программа должна получать от задачника и ему же передавать найденные результаты. Промежуточные значения дактилограммы, вычисляемые в ходе работы схемы Горнера и выводимые в цикле **for**, являются данными-индикаторами, позволяющими проверить, что ее вычисление проводилось с применением алгоритма, описанного в задании.

При каждом запуске программы задачник передает ей новый набор исходных данных, сгенерированный с применением датчика случайных чисел, проверяет правильность переданных ему результатов и отображает в окне, аналогичном приведенному на рис. 5, всю информацию, связанную с тестовым запуском: набор исходных данных, полученный набор результатов, а также контрольный («правильный») набор результатов, отображаемый на вкладке «Пример верного решения». Задачник автоматически проверяет правильность операций ввода-

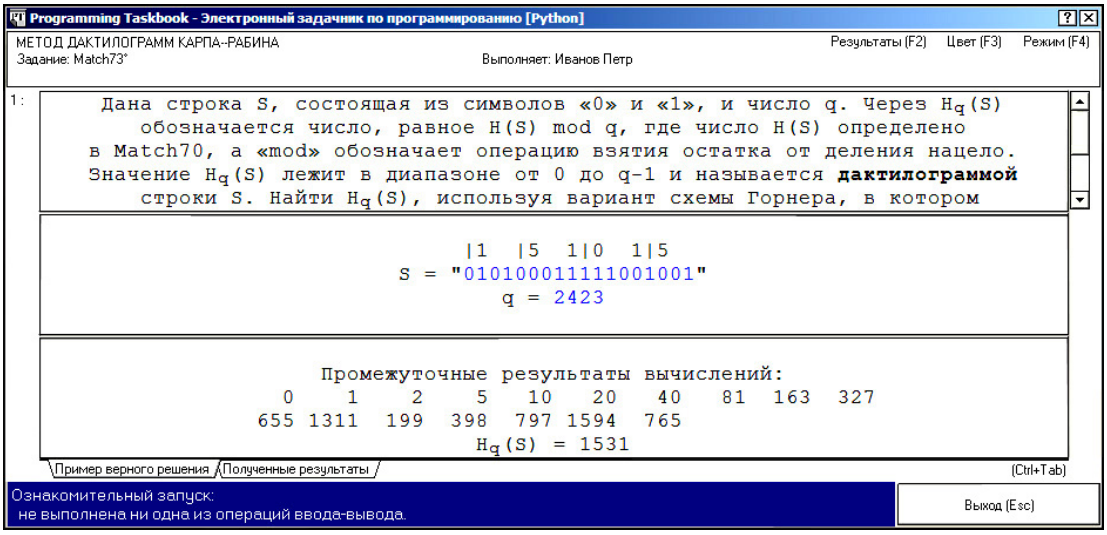

Рис. 5. Ознакомительный запуск программы с заданием

вывода и при выполнении ошибочных действий (например при попытке ввести лишние исходные данные или при выводе не всех требуемых результатов) выводит соответствующие сообщения. Результаты каждого запуска программы не только выводятся на экране, но и фиксируются в специальном зашифрованном файле, просмотреть который можно с помощью программного модуля PT4Results, входящего в состав задачника. Задание считается выполненным, если программа правильно обработает несколько наборов исходных данных, предложенных ей задачником; для заданий групп Match и Align требуется успешно пройти 5 тестовых испытаний, причем в случае ошибочной обработки исходных данных для какого-либо теста отсчет успешных тестов начинается заново.

6. ИНТЕРНЕТ-РЕСУРСЫ, СВЯЗАННЫЕ С ЗАДАЧНИКОМ

Подробная информация о задачнике Programming Taskbook и его дополнениях приводится на сайте http://ptaskbook.com/; с этого сайта можно также скачать дистрибутивы задачника и его дополнений. Различные дополнения к задачнику представлены в виде разделов сайта. Раздел «PT for Bio» включает общее описание задачника по строковым алгоритмам биоинформатики, тексты формулировок всех заданий групп Match и Align, а также описание процесса выполнения нескольких типовых заданий на различных языках.

Около формулировки каждого задания указываются ссылки *Pascal*, *C#*, *VB.NET*, позволяющие немедленно приступить к выполнению этого задания на выбранном языке в веб-среде программирования ProgrammingABC.NET WDE. При выборе одной из этих ссылок в браузере отображается веб-среда, и в нее сразу загружается программа-заготовка для требуемого задания. Процесс выполнения заданий в веб-среде ничем не отличается от выполнения заданий в обычных программных средах, однако не требует предварительной установки дополнительного программного обеспечения на компьютер студента: вся работа выполняется в интернет-браузере [9].

7. ЗАКЛЮЧЕНИЕ

Использование задачника PT for Bio позволяет повысить эффективность изучения строковых алгоритмов на практических занятиях; это достигается, прежде всего, за счет автоматизации действий по подготовке исходных данных и проверке полученных результатов. Последовательное выполнение серии заданий, посвященных определенному алгоритму, дает возможность ознакомиться с новыми понятиями, используемыми в алгоритме, реализовать и протестировать каждый этап алгоритма и, в итоговом задании, – алгоритм в целом. Вспомогательные выходные данные-индикаторы обеспечивают дополнительную проверку правильности реализованного алгоритма или его этапа, система ссылок в формулировках заданий демонстрирует связи между родственными алгоритмами. Задания могут выполняться на разных языках и в различных программных средах.

Подход, использованный при разработке задачника PT for Bio, может с успехом применяться при создании аналогичных обучающих программных комплексов для курсов, связанных с программированием, разработкой алгоритмов, изучением различных языков и стандартных библиотек. В качестве платформы для подобных комплексов может применяться универсальный задачник Programming Taskbook, а в качестве инструмента их разработки – входящий в его состав конструктор учебных заданий.

Литература

1. *Гасфилд Д.* Строки, деревья и последовательности в алгоритмах: Информатика и вычислительная биология. СПб.: БХВ-Петербург, 2003.

2. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ. М.: МЦНМО, 2000.

3. *Абрамян М.Э.* Реализация универсального электронного задачника по программированию // Информатика и образование, 2009. № 6. С. 118–120.

4. Ejudge contest management system / http://ejudge.ru (дата обращения: 30.04.2012).

5. PascalABC.NET / http://pascalabc.net/ (дата обращения: 30.04.2012).

6. Веб-среда программирования ProgrammingABC.NET / http://pascalabc.net/WDE/ (дата обращения: 30.04.2012).

7. *Абрамян М.Э.* Использование специализированного программного обеспечения для преподавателя при организации и проведении лабораторных занятий по программированию // Информатика и образование, 2011. № 5. С. 78–80.

8. *Абрамян М.Э.* Электронный задачник по параллельному программированию на основе технологии MPI // Компьютерные инструменты в образовании, 2011. № 6. С. 47–54.

9. *Абрамян М.Э., Белякова Ю.В., Михалкович С.С.* Многоязыковая Web-среда программирования ProgrammingABC.NET и интеграция в нее электронного задачника Programming Taskbook // Труды XIX Всероссийской научно-методической конференции «Телематика'2012». Том 2. СПб., 2012. С. 288–289.

Abstract

We discuss some aspects of educational software aimed to help students to improve their skills of construction correct implementations for subtle and tricky algorithms. The paper describes the *Programming Taskbook for Bioinformatics*, an electronic book of educational training tasks on string algorithms. The electronic book contains 160 tasks covering a wide range of classical and semi-numerical algorithms of exact string matching and algorithms of approximate string matching including algorithms of global and local alignment and searching for the longest common subsequence. The paper also contains several examples of the tasks illustrating some features of the electronic book.

Keywords: educational software, exact and approximate string matching algorithms, bioinformatics.

Абрамян Михаил Эдуардович,
кандидат физико-математических наук, доцент кафедры алгебры и дискретной математики Южного федерального университета,
mabr@math.sfedu.ru