УДК 51:37

Н. А. Иванова

ВОЗМОЖНЫЕ НАПРАВЛЕНИЯ ПРИМЕНЕНИЯ РЕСУРСОВ ПРОГРАММИРОВАНИЯ СРЕДЫ *MATHEMATICA* ПРИ РЕШЕНИИ МАТЕМАТИЧЕСКИХ ЗАДАЧ

Описаны возможные направления применения ресурсов программирования среды Mathematica *(на примере версии 7.0.1.0) в обучении математике студентов средних специальных учебных заведений в качестве средства автоматического контроля ввода исходных данных, генератора исходных данных и средства повышения наглядности процесса решения математических задач. Представлены примеры кода на* Mathematica *по каждому из трех возможных направлений. Определена последовательность этапов при отображении средствами* Mathematica *всего хода решения поставленной задачи.*

This article describes possible applications of the resources of the Mathematica programming environment (version 7.0.1.0) in teaching mathematics as a means of the automatic control of initial data input, input data generator, and a means to improve the visibility of mathematical problem solving. The author offers examples of Mathematica codes for each case and identifies the sequence of stages of displaying the whole solution process by means of Mathematica.

Ключевые слова: ресурсы программирования *Mathematica*, средство контроля входных данных, генератор данных, средство повышения наглядности.

Key words: Mathematica development environment, input data control tool, data generator, visibility increase tool.

При создании тестирующих систем, интерактивных учебных пособий или при обычном решении задач в среде *Mathematica* важно не только правильно подать материал, но и максимально освободить преподавателя от рутинной работы по «раздаче» исходных данных, самостоятельному решению задач и поиску ошибок в решениях. *Mathematica* при помощи своих ресурсов может с легкостью выполнить данные функции за преподавателя.

Ресурсы программирования среды *Mathematica* могут быть использованы в решении задач, организации тестирующих систем или создании интерактивных учебных пособий по дисциплинам математического цикла в качестве средства контроля входных данных и повышения наглядности при решении задач.

Цель данной статьи — на простейших примерах рассмотреть возможности использования ресурсов программирования среды *Mathematica* при решении математических задач. Ранее данному материалу отдельное внимание не уделялось.

1. Ресурсы программирования среды* Mathematica *как средство контроля входных данных при решении задач.

На начальном этапе изучения математики любой студент сталкивается с различными видами чисел. В связи с этим в *Mathematica* разработчиками предусмотрены функции, позволяющие проверить некоторые свойства числа: является ли, например, число простым (функция

PrimeQ). Есть также возможность определения перечня (списка) простых чисел (всех или входящих в указанный диапазон — функции Prime, NextPrime, PreviousPrime и др.). Однако функции, определяющие некоторые другие свойства, отсутствуют, например нет функции проверки: натуральное число или нет. Стоит отметить, что данные понятия (натуральное, рациональное, простое число и др.) очень часто встречаются и в математических дисциплинах, и в программировании, а значит, есть необходимость в обучении студентов сущности содержания данных понятий.

Предположим, что перед студентом поставлено следующее задание: для любых двух натуральных чисел найти наибольший общий делитель. Для выполнения данного задания необходимо знать понятия: «натуральное число», «делитель», «общий делитель» и «наибольший общий делитель».

Наибольший общий делитель в области целых, рациональных и гауссовых чисел в *Mathematica* можно найти с помощью функции GCD. Однако, использовав вместо натурального числа любое целое (в том числе отрицательное) и получив ответ, студент так и не узнает, что задание выполнено ошибочно. Например, функция GCD[−45, 81] дает ответ «9», а GCD[45/2, 81] дает ответ «9/2». В данном случае ограничить ввод некорректных исходных данных помогут средства программирования в *Mathematica* (рис. 1).

```
n = Input["Введите первое натуральное число"];
m = Input["Введите второе натуральное число"];
k = 0;
If[n < 0 || n != Round[n], Print["Число ", n, " не натуральное"]; k = 1 ];
If[m < 0 || m != Round[m], Print["Число ", m, " не натуральное"]; k = 2 ];
If[k == 0, Print["Оба числа натуральные"];
  Print["НОД(", n, ", ", m, ") = ", GCD[n, m]] ]


Число -5 не натуральное
      21
Число --- не натуральное
      2
```

Рис. 1. Программный код в *Mathematica* по проверке введенных чисел на натуральность и поиск наибольшего общего делителя

Представленный программный код проверяет входные данные, прежде чем начинать вычисление. В случае некорректности данных (в примере были введены −5 и 10,8) выходит соответствующее сообщение, и функция GCD не выполняется. Если данные введены корректно, то выводится сообщение о том, что числа натуральные и дается результат.

Использование средств программирования в данном случае заставляет студента задуматься: почему число не натуральное? что такое натуральное число? каким должно быть натуральное число? Если при этом будет доступен программный код (доступный код для пользователя — это открытый код), который читать интереснее, чем учебник

или справочник по математике, то это избавит студента от поиска ответов на его вопросы и в большинстве случае сэкономит время.

2. Ресурсы программирования среды* Mathematica *как генератор исходных данных при решении задач.

При решении задач математического цикла важно не только правильно вводить исходные данные, но и вводить именно корректные данные. Речь идет о необходимости устранения «идеальных» и слишком простых входных данных. Например, при поиске наибольшего общего делителя студент может ввести в качестве исходных данных «2» и «4», что позволит ему без труда определить (или даже угадать) искомое.

Избежать ввода «идеальных» данных поможет генерация исходных значений, которая в *Mathematica* выполняется при помощи функций Random[], RandomPrime[] и др.

Упрощенно это может быть организовано следующим образом (рис. 2):

```
n = Random[Integer, {1, 50}];
m = Random[Integer, {5, 100}];
Print["НОД(", n, ", ", m, ") = ", GCD[n, m]]


НОД(30, 12) = 6
```

Рис. 2. Программный код в Mathematica
по генерации исходных чисел для поиска наибольшего общего делителя

Отметим, что дополнительно можно использовать сброс генератора случайных чисел, привязку к начальному числу времени дня и другие возможности *Mathematica* в сфере генерации случайных чисел.

3. Ресурсы программирования среды* Mathematica *как средство повышения наглядности решений задач.

Еще одно преимущество использования средств программирования в *Mathematica* — это возможность расписать (наглядно отобразить) весь процесс любого производимого вычисления. Дело в том, что среда *Mathematica* показывает только результат (ответ) по введенным исходным данным; весь процесс вычислений и преобразований происходит внутри системы и пользователю недоступен. Не всегда доступны и промежуточные результаты (хотя по некоторым вычислениям это позволяют сделать вспомогательные функции).

Применяя средства программирования на языке *Mathematica*, можно отобразить весь процесс вычисления «изнутри», то есть дать возможность обучаемому проверить поэтапно работу среды с производимыми пользовательскими действиями.

В вышеприведенных примерах (рис. 1 и 2) в системе *Mathematica* мы обеспечили ввод, генерацию и проверку исходных данных, которые не могут быть реализованы без средств программирования. Однако этого не достаточно. Необходимо показать весь процесс вычисления наибольшего общего делителя, но сделать это, представляется, нужно в такой последовательности:

1) ввод исходных данных пользователем (например, ввод чисел);

2) проверка программой исходных данных на корректность (например, проверка двух чисел на натуральность);

3) решение задания пользователем и ввод результата — пользователь самостоятельно решает задание и вводит результат в отведенное поле;

4) проверка программой результата — программа также решает задание и проверяет результат пользователя;

5) в случае правильного ответа пользователя (когда результат, полученный самой программой, совпадает с введенным результатом пользователя) — извещение об этом;

6) в случае неправильного ответа пользователя — последовательное отображение всего хода решения с выводом промежуточных результатов, если они существуют.

Перечисленные этапы и их последовательность можно отобразить в графическом виде (в виде укрупненной блок-схемы) (рис. 3).

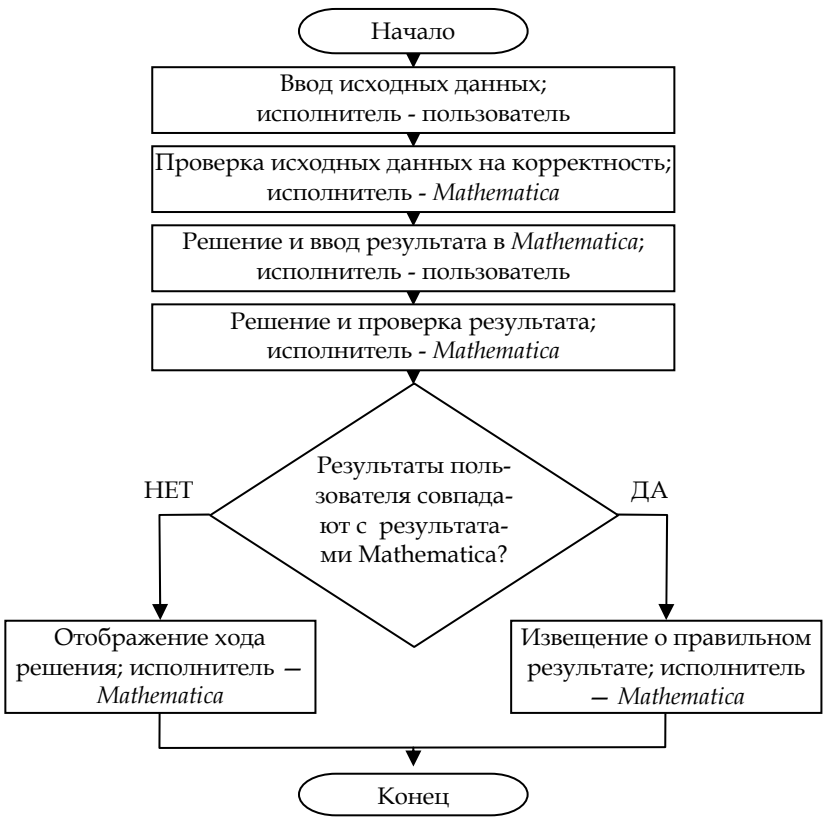

Рис. 3. Последовательность этапов при отображении средствами *Mathematica* всего хода решения поставленной задачи

Представленные этапы не просто позволят отобразить ход решения, но и обеспечат первоначальную самостоятельную работу пользователя и полноценный диалог со средой *Mathematica*.

Рассмотрим несложный пример, когда от студента требуется ввести два натуральных числа и вычислить их наибольший общий делитель (НОД). Программный код по проверке результата пользователя и выводу промежуточных результатов будет выглядеть следующим образом (рис. 4):

```
n = Input["Введите первое натуральное число", 54];
m = Input["Введите второе натуральное число", 81];
k = 0;
If[n < 0 || n != Round[n], Print["Число ", n, " не натуральное"]; k = 1 ];
If[m < 0 || m != Round[m], Print["Число ", m, " не натуральное"]; k = 1 ];
If[k == 0, Print["Оба числа натуральные"]];
ot = Input["Введите Ваш ответ:"];
If[ot == GCD[n, m], Print["Ответ верный"]; Print["НОД(", n, ", ", m, ") = ", GCD[n, m]],
    Print["Ответ неверный"]];
DialogInput[{ TextCell["Вывести промежуточные данные для сверки? "],
      ChoiceButtons[{k = 0; DialogReturn[ ], k = 1; DialogReturn[ ]}]}];
If[k == 0, Print["Делители первого числа – ", n, " = ", Divisors[n],
    " Делители второго числа – ", m, " = ", Divisors[m]] ]
```


Рис. 4. Программный код в *Mathematica*,
обеспечивающий вывод промежуточных результатов вычисления НОД

При запуске данного кода студент взаимодействует со средой *Mathematica* при помощи диалоговых окон (рис. 5).

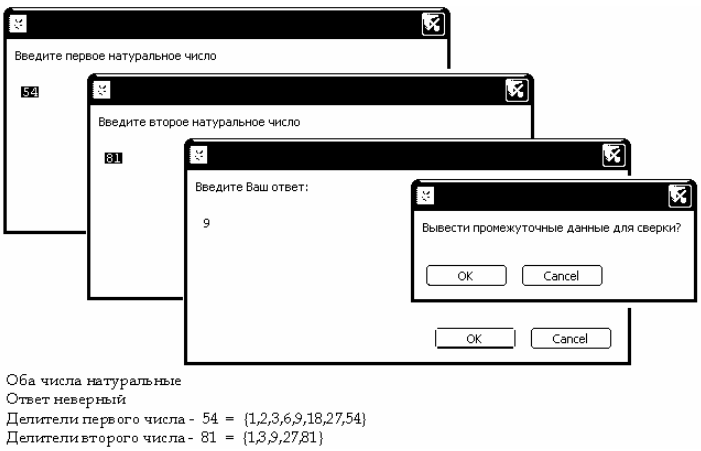

Рис. 5. Взаимодействие пользователя (студента) со средой *Mathematica*
при решении задачи на вычисление НОД

Необходимо отметить, что на рисунках 4 и 5 представлен код, обеспечивающий лишь вывод промежуточных результатов при вычислении НОД. По подобному алгоритму можно обеспечить вывод и всего хода решения (например, для вычисления НОД это может быть алгоритм Евклида).

Указанная последовательность этапов, представляется, должна лежать в основе создания интерактивных учебных пособий на базе *Mathematica*, благодаря которым могут быть достигнуты сразу три цели:

1) изучение основ работы системы *Mathematica*;

2) обучение дисциплинам математического цикла;

3) закрепление или изучение основ программирования (в случае открытого кода и использования таких пособий для студентов, знакомых с программированием).

В качестве заключения необходимо отметить, что рассмотренные в данной статье направления использования ресурсов программирования системы *Mathematica* позволят в процессе обучения математике:

— повысить интерес к математике за счет использования компьютерной системы *Mathematica* и элементов программирования;

— обеспечить наглядность подачи материала и процесса решения задания, так как каждая строка кода сопровождается комментарием и часто выполняет только одно-два действия;

— приучить студентов более внимательно относиться к исходным данным и тщательно анализировать и проверять соответствие входных данных условию задания;

— научить студентов четко и логично мыслить и тщательнее следить за ходом решения задач;

— избегать отказа работы функции из-за некорректности ввода данных и дальнейшего некорректного вычисления по причине отказа.

Об авторе

Наталья Александровна Иванова — преп., Бугульминский машиностроительный техникум, соиск., Набережночелнинский государственный педагогический институт, Набережные Челны.

E-mail: fytmp@bk.ru

About author

Natalia Ivanova, Lecturer, Bugulma Engineering College; PhD student, Naberezhnye Chelny State Pedagogical Institute, Naberezhnye Chelny.

E-mail: fytmp@bk.ru