УДК 004.052:372.862
DOI 10.21685/2587-7704-2019-4-1-9

Open Access | RESEARCH ARTICLE

Обучение студентов вопросам надежности программного обеспечения

А. В. Артюхин
Пензенский государственный университет, Россия, 440026 г. Пенза, ул. Красная, 40

Н. А. Егорова
Пензенский государственный университет, Россия, 440026 г. Пенза, ул. Красная, 40

Аннотация. Проведен обзор методов функционального тестирования программ. Были описаны процедуры шифрования текстового файла и внесенных в программу недекларированных возможностей, в частности, это общее описание программы и ее интерфейса, а также описание алгоритма программы. Проведенные исследования показали, что функциональное тестирование позволяет выявлять не только ошибки, но и недекларированные возможности программного обеспечения, что особенно актуально для программ, обеспечивающих конфиденциальность, целостность и доступность.

Ключевые слова: информация, безопасность, защита информации, программное обеспечение, надежность, шифрование, недекларированные возможности, тестирование, функциональное тестирование.

Training students in software reliability

A. V. Artyukhin
Penza State University, 40 Krasnaya Street, 440026, Penza, Russia

N. A. Egorova
Penza State University, 40 Krasnaya Street, 440026, Penza, Russia

Abstract. A review of methods for functional program testing was conducted. The procedures for encrypting a text file and the undocumented feature entered into the program were described. In particular, this is a general description of the program and its interface, as well as a description of the program's algorithm. The research has shown that functional testing allows identifying not only errors, but also the undocumented feature of software, which is especially important for programs that ensure confidentiality, integrity and availability.

Keywords: information, security, information protection, software, reliability, encryption, undocumented feature, testing, functional testing.

В рамках учебного плана дисциплины «Разработка и эксплуатация защищенных автоматизированных систем» студенты изучают проблемы тестирования программного обеспечения (ПО) для определения показателей его надежности.

История тестирования программного обеспечения отражает эволюцию разработки самого программного обеспечения. В течение длительного времени в разработке программного обеспечения уделялось основное внимание крупномасштабным научным программам, а также программам Министерства обороны, связанным с системами корпоративных баз данных, которые проектировались на базе универсальной ЭВМ или мини-компьютера. Тестовые сценарии записывались на бумагу. С их помощью проверялись целевые потоки управления, вычисления сложных алгоритмов и манипулирование данными. Окончательный набор тестовых процедур мог эффективно протестировать всю си-

© *Артюхин А. В., Егорова Н. А.,* 2019.
Данная статья доступна по условиям всемирной лицензии Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), которая дает разрешение на неограниченное использование, копирование на любые носители при условии указания авторства, источника и ссылки на лицензию Creative Commons, а также изменений, если таковые имеют место.

стему полностью. Тестирование обычно начиналось лишь после завершения плана-графика проекта и выполнялось тем же персоналом.

Тестирование систем реального времени потребовало другого подхода к проектированию тестирования из-за того, что рабочие потоки могли вызываться в любом порядке. Эта особенность привела к появлению огромного количества процедур тестирования, способных поддержать бесконечное число перестановок и сочетаний [1].

Чтобы прояснить полную картину тестирования программного обеспечения, необходимо обратиться к определениям, написанным ниже.

Тестирование программного обеспечения – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и ее ожидаемым поведением на конечном наборе тестов, выбранных определенным образом [2].

Функциональное тестирование – это тестирование ПО в целях проверки реализуемости функциональных требований, т.е. способности ПО в определенных условиях решать задачи, нужные пользователям. Функциональные требования определяют, что именно делает ПО, какие задачи оно решает.

Функциональные требования включают в себя:

– функциональную пригодность;

– точность;

– способность к взаимодействию;

– соответствие стандартам и правилам;

– защищенность [3].

Для разработки эффективных тестовых наборов данных при функциональном тестировании используются методы эквивалентного разбиения, анализа граничных значений и функциональных диаграмм.

Метод эквивалентного разбиения осуществляется в два этапа, первый из которых заключается в выделении классов эквивалентности, а второй – в построении тестов.

Метод анализа граничных значений предполагает исследование ситуаций, возникающих на границах и вблизи границ эквивалентных разбиений. Например, если правильная область значений есть интервал от –1,0 до +1,0, то нужно предусмотреть тесты –1,0, 1,0, –1,001 и 1,001.

В соответствии с методом функциональных диаграмм в спецификации программы выделяются причины и следствия. Причина – это отдельное входное условие, или класс эквивалентности входных условий. Следствие – выходные условия или результат преобразования системы. Каждой причине и следствию приписывается уникальный номер [4].

Для изучения перечисленных выше методов тестирования необходимо разработать соответствующий инструмент, т.е. программу, в которую намеренно внесены ошибки. Для выполнения учебной задачи на оценку «отлично», «хорошо» и «удовлетворительно» ошибки должны быть трех уровней сложности, так как каждый студент по-разному воспринимает и осваивает учебно-методический материал.

В качестве целевой функции программы было выбрано шифрование фалов формата *.txt* методом гаммирования. В программу внесены следующие ошибки: производится шифрование только первого мегабайта из файла; для файлов, которые созданы не раньше определенной даты, шифрование производится путем инверсии бит. Кроме того, в программе на самом деле реализовано шифрование любых файлов, а не только текстовых. Однако в описании программы этого не указано. Это противоречие должно показать студентам, что ошибки могут быть не только в самом программном обеспечении, но и в его описании.

Для выполнения первого уровня сложности, т.е. на оценку «удовлетворительно», студенту достаточно воспользоваться методом эквивалентного разбиения, чтобы найти противоречие между описанием программы и ее действительными функциональными возможностями. Для выполнения второго уровня сложности, т.е. на оценку «хорошо», студенту необходимо найти ошибку работы программы с файлами, размер которых будет составлять больше одного мегабайта. Для этого студенту уже необходимо будет применить все три описанных выше метода функционального тестирования. Для выполнения третьего уровня сложности, т.е. на оценку «отлично», студенту необходимо найти ошибку работы программы с файлом, который создан раньше определенной даты. Чтобы найти решение этой задачи, студенту, кроме знаний из изучаемого курса, необходимо применить знания и навыки из дисциплин, изучаемых ранее.

Для разработки программы шифрования текстовых файлов с внесенными ошибками была использована среда Visual Studio 2012.

В данной среде был создан интерфейс программы, показанный на рис. 1.

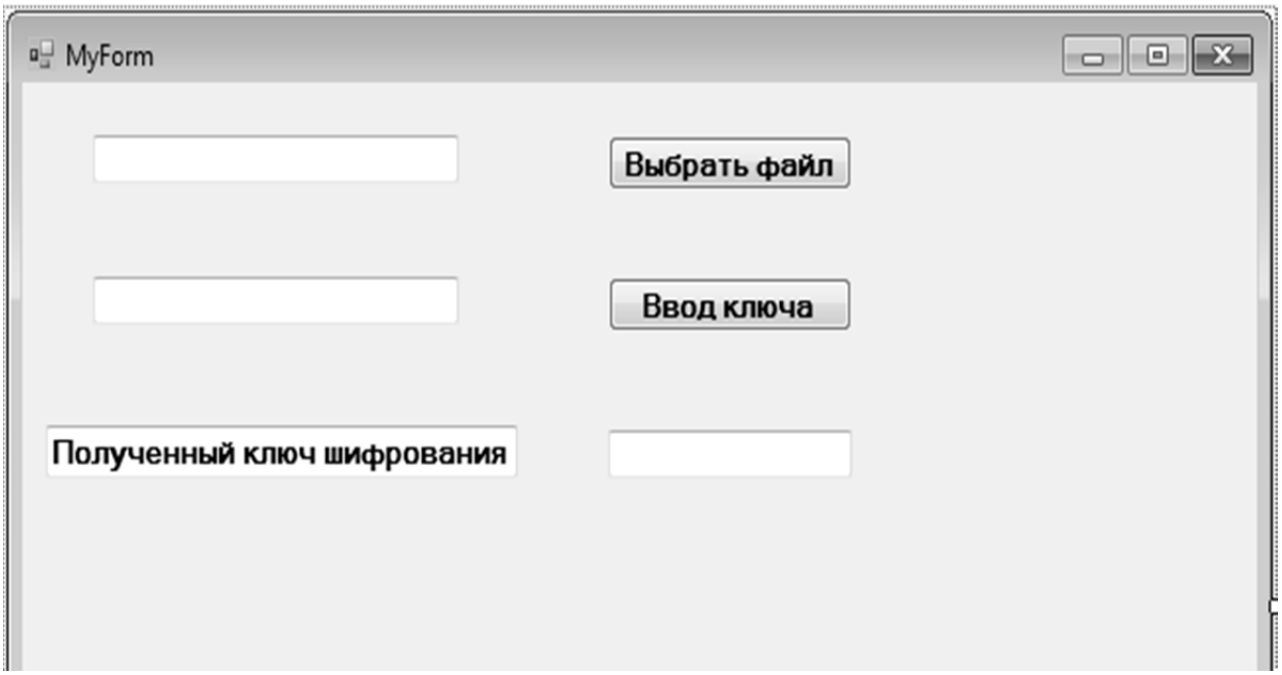

Рис. 1. Интерфейс программы

В этом проекте пользователю предоставляется возможность выбрать файл для шифрования с помощью определенной кнопки. Рядом с кнопкой хранится путь и имя выбранного пользователем файла. Для ввода ключа, который хранится в специальном файле, используется соответствующая вторая кнопка.

На рис. 2, 3 показан пример работы программы исходного файла и зашифрованного файла соответственно.

Таким образом, разработанная программа шифрования текстовых файлов с внесенными ошибками показала свою работоспособность. Она может быть использована при изучении студентами курса «Разработка и эксплуатация защищенных автоматизированных систем».

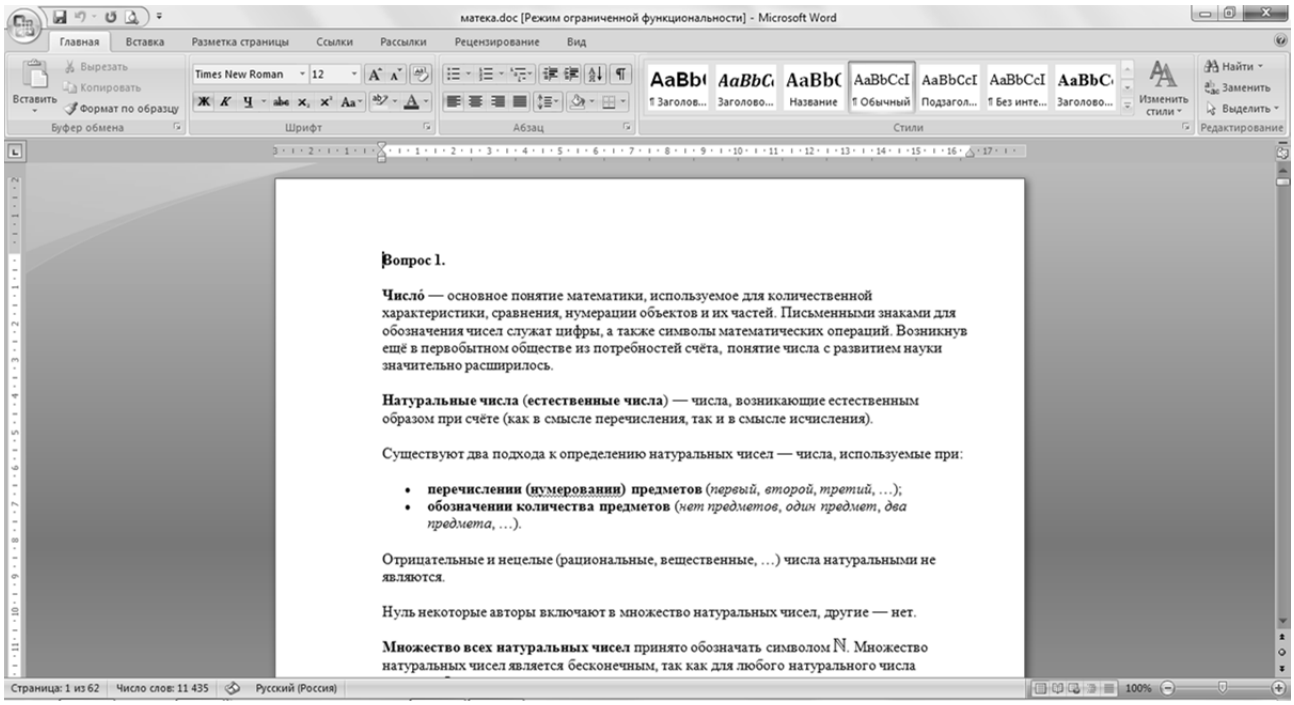

Рис. 2. Исходный файл

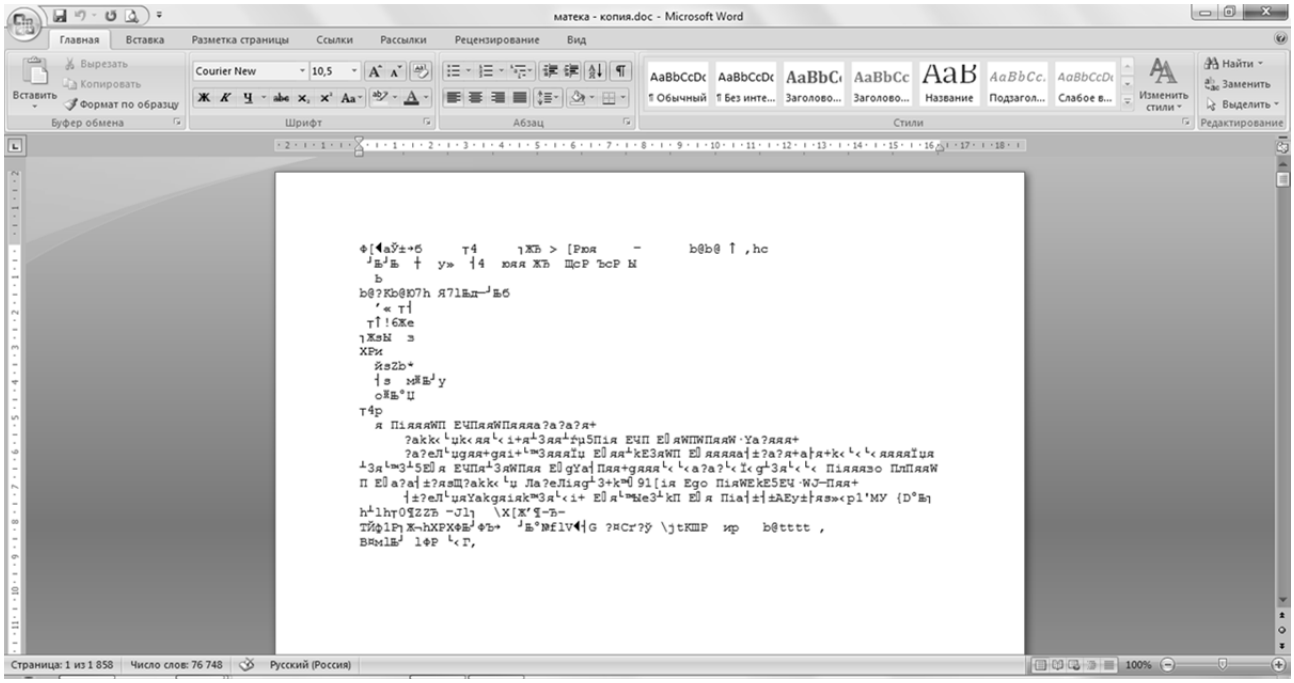

Рис. 3. Зашифрованный файл

Библиографический список

1. Введение. Общие понятия. Тестирование программного обеспечения. – URL: https://studbooks.net/2252834/informatika/vvedenie (дата обращения: 11.10.2018).
2. Тестирование программного обеспечения. – URL: https://ru.wikipedia.org/wiki/Тестирование_программного_обеспечения (дата обращения: 11.10.2018).
3. Функциональное тестирование. – URL: https://ru.wikipedia.org/wiki/Функциональное_тестирование (дата обращения: 11.10.2018).
4. Принципы тестирования. Классификация методов тестирования. Методы структурного тестирования. Методы функционального тестирования. – URL: https://lektsii.com/2-23308.html (дата обращения: 11.10.2018).

Образец цитирования:

Артюхин, А. В. Обучение студентов вопросам надежности программного обеспечения / А. В. Артюхин, Н. А. Егорова // Инжиниринг и технологии. – 2019. – Vol. 4(1). – С. 1–4. – DOI 10.21685/2587-7704-2019-4-1-9.