Многократно проведенные косвенные измерения емкостей различных конденсаторов (при различных частотах следования импульсов заряда-разряда, различных величин сопротивления R, через которое реализуются эти процессы) дают результат не более 10 % от величин емкостей, гарантируемых заводом изготовителем.

Указанные уточнения, внесенные при расчете емкости конденсатора, служат основанием и основой для изменения классической физической модели (рисунок 2) и соответствующей ей математической модели переходных процессов конденсаторов в виде формул (1) и (2).

Предложенная методика, как дидактическое средство, интенсифицирует обучение физике раздела «Электричество»; знакомит студентов с современной цифровой измерительной техникой.

Список цитированных источников

1. Ставинский, Н. Н. Изучение процессов заряда-разряда конденсаторов: физический практикум / Н.Н. Ставинский. Дальневост. Федеральн. Университет [Электронный ресурс]. – Владивосток. 2014. – 8 с. – Режим доступа: https://bb.dvfu.ru/bbcswebdav/orgs/FU50701.

2. Крутов, А. В. Теоретические основы электротехники. Лабораторный практикум : в 2 ч. / А. В. Крутов, Н. Г. Крылова, Т. Ф. Гузанова. – Минск : БГАТУ, 2021. – Ч 2 – 91 с.

3. Величко, Л. А. Определение емкости конденсатора на основе переходных процессов в электрической цепи : в 2 ч. / А. В. Величко, Н. Н. Ворсин, К. М. Маркевич / Метод. указан. к вып. лаб. раб. по физике – Брест : БрГТУ, 2018. – Ч 2. – С. 11–19.

УДК 004.413.5

Палто Е. С.
Научный руководитель: ст. преподаватель Анфилец С. В.

СИСТЕМА АНТИПЛАГИАТА КОДА НА ЯП PYTHON

Система антиплагиата кода (также известная как система обнаружения плагиата или система проверки оригинальности кода) представляет собой инструмент, который используется для выявления и сравнения сходства между программными кодами. Она предназначена для обнаружения случаев, когда один программный код сильно похож на другой, что может указывать на плагиат или несанкционированное копирование.

На данный момент разработка системы антиплагиата кода крайне актуальна по нескольким причинам: остро стоит вопрос о необходимости системы для выявления сходств между программными кодами и предотвращения плагиата, не существует подобных продуктов в общем доступе.

Применение: в учреждениях образования, для контроля при написании дипломных, курсовых и лабораторных работ. Также можно включить регистрацию программного обеспечения для контроля проверки плагиата, что позволит повысить значимость интеллектуальной собственности при проверке олимпиадных работ и контестов.

Для разработки данной системы был выбран язык программирования Python. Создание системы антиплагиата кода на Python имеет несколько преимуществ. Во-первых, Python является одним из наиболее популярных и широко используемых языков программирования, что обеспечивает доступность и поддержку со стороны сообщества разработчиков, если будет принято решение загрузить исходный код в общий доступ. Во-вторых, в будущем планируется создание нейронной модели, которая ляжет в основу системы антиплагиата кода, а Python предоставляет огромное количество библиотек и фреймворков. Они предоставляют обширные возможности для построения и обучения различных типов нейронных сетей, а также предлагают гибкую и мощную функциональность для создания сложных моделей и реализации различных архитектур.

Основной идеей создания системы антиплагиата кода послужила мысль: чтобы создать систему антиплагиата кода, нужно научиться «плагиатить» код.

«Сплагиатить» код можно следующими способами: можно ничего не менять, написать комментарии к коду, изменить названия переменных и названия функций, изменить порядок исполнения кода, удалить фрагменты кода. Программа выводит вероятность плагиата кода в процентах.

Было несколько подходов для создания системы антиплагиата кода, результаты работы программ приведены в таблице 1. Тестирование проводилось на лабораторных работах второго семестра.

Описание первого подхода

Был использован алгоритм “Расстояние Левенштейна” [1, с. 23; 2, с. 15].

Расстояние Левенштейна – метрика, измеряющая по модулю разность между двумя последовательностями символов и на его основе была написана система антиплагита кода.

Описание второго подхода

Представление кода в виде текста. В самом начале программа вычищает код от комментариев, т. к. вероятность плагиата не должна понижаться от их количества (ведь счет ведется посимвольно). Далее система находит наиболее похожие строки (с наименьшим редакционным расстоянием), таким образом находя наименьшее расстояние Левенштейна для кода.

Описание третьего подхода

Конвертация кода в AST-строку(AST в Python – это структура данных, которая представляет исходный код Python в виде дерева, где узлы представляют операторы и выражения, а листья – константы и идентификаторы.). Этот подход позволяет работать с кодом как с деревом (а также автоматически чистит код от комментариев и пустых строк, не считая их за часть кода). Что даёт возможность проводить глубокий сравнительный анализ. А после конвертировать AST-строку обратно в код. Для комфортной работы был разработан интерфейс (рисунок 1). Также пользователь сам может создать базу с кодами, с которыми в дальнейшем и будет происходить сравнение (рисунок 2).

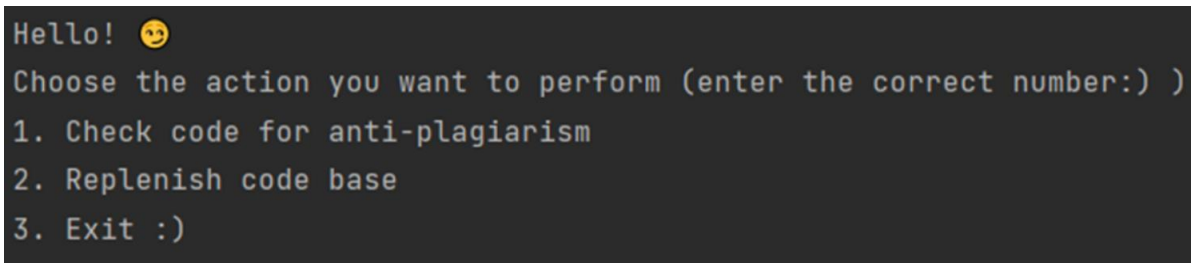

Рисунок 1 – Интерфейс системы антиплагиата кода

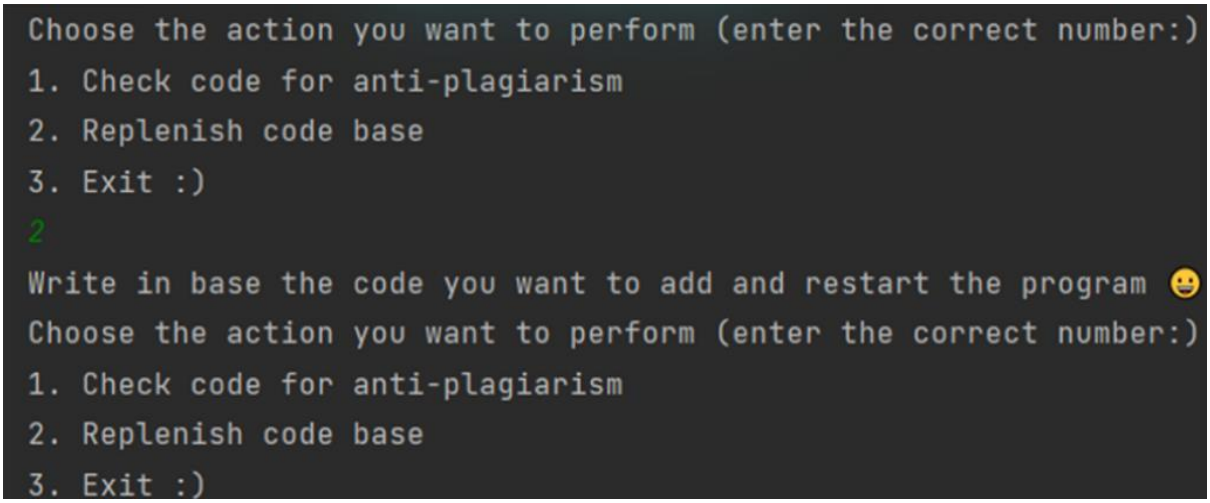

Рисунок 2 – Пополнение/создание базы

Таблица 1 – Результаты работы систем антиплагиата кода

Система антиплагиата/ метод	Вероятность плагиата кода в процентах		
	Первый подход	Второй подход	Третий подход
Нет изменений	100 %	100 %	100 %
Написание комментариев	34 %	100 %	100 %
Изменение названий	64.7 %	64.7 %	100 %
Изменение порядка	83.3 %	100 %	100 %
Удалить фрагмент кода	100 %	100 %	100 %
Всё вместе	11.8 %	64.7 %	100 %
2 разных кода, выполняющих одни и те же действия	0.33 %	1.8 %	10 %

Как показал анализ таблицы, наиболее эффективным оказался третий подход, который и позволяет выводить наиболее точный процент вероятности плагиата кода.

Список цитированных источников

1. Левенштейн, В. И. Двоичные коды с исправлением выпадений, вставок и замещений символов / В. И. Левенштейн // Доклады Академий Наук СССР. – 1965. – Т. 163, № 4 – С. 845–848.

2. Гасфилд, Д. Строки, деревья и последовательности в алгоритмах. Информатика и вычислительная биология / Д. Гасфилд; пер. с английского И. В. Романовского. – Невский Диалект; БВХ-Петербург, 2003. – 654 с.