УДК 004.91

НЕКОТОРЫЕ АСПЕКТЫ В ОБРАБОТКЕ ТЕКСТОВОЙ ИНФОРМАЦИИ

А. Е. Грицук, Т. Г. Хомицкая
Брестский государственный технический университет, г. Брест

В XXI веке объем информации, хранящейся и обрабатываемой в Интернете, удваивается каждые восемнадцать месяцев, причем до 95% всей информации является неструктурированной. Вместе с понятием информационного взрыва в обиход вошел термин Big Data, или Большие Данные, – совокупность структурированных и неструктурированных данных огромных объемов и значительного многообразия [1]. Сложившиеся условия делают способность обработки и анализа Big Data фактором конкурентного преимущества организации.

При обработке данных перед пользователем достаточно часто стоит задача, когда необходимо преобразовать список ФИО с полным вариантом написания имен и отчеств («Фамилия Имя Отчество») в форму «И.О. Фамилия» или «Фамилия И.О.» Для решения данной проблемы можно воспользоваться строенными возможностями и функциями офисных приложений (Microsoft Excel, Google Таблицы) или автоматизировать процесс, разработав алгоритм преобразования, реализованный на одном из языков программирования.

Пусть в исходном файле содержится список, представляющий собой набор из ФИО, где между фамилией и именем, именем и отчеством содержится символ пробела, а в конце строки после отчества – символ конца абзаца, причем некоторые имена или отчества могут быть набраны с нижнего регистра.

Суть преобразования основывается на формировании отдельных наборов из фамилий, имен и отчеств, на основании которых при помощи операции конкатенации формируется новая строковая величина требуемой формы.

В Microsoft Excel для реализации первого этапа можно использовать возможность приложения – разбиение текста по столбцам. Для этого необходимо выделить весь список данных, вызвать диалоговое окно «Мастер текстов (разбор)» (вкладка «Данные», группа «Работа с данными», кнопка «Текст по столбцам»). Затем последовательно выполнить три шага в диалоговом окне: указать, что формат данных содержит знаки-разделители (1) в виде символа «пробел» (2), а также задать ячейку (3), начиная с которой обеспечить вывод данных. Для контроля выполняемых действий диалоговое окно сдержит образец разбора данных.

В Google Таблицах для проведения первого этапа можно использовать функцию SPLIT(), которая позволяет разбить «текстовую строку» по «разделителю» (рис. 1).

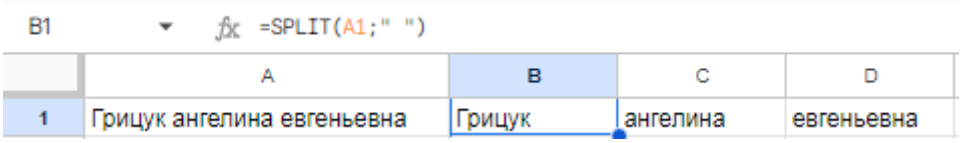

Рисунок 1 – Пример использования функции SPLIT() для первого этапа.

Очевидно, что использование функции в Google Таблицах является более гибким решением первого этапа, нежели процедурная реализация в Microsoft Excel, поскольку позволяет быстрее реагировать на исправления неточностей.

Выполнение второго этапа и в Microsoft Excel, и в Google Таблицах является одинаковым: вводится формула, включающая использование функции СЦЕПИТЬ() для объединения в одну строку последовательности строк, функции ЛЕВСИМВ() для выделения первых букв имени и отчества, функции ПРОПИСН() для преобразования каких-либо букв к прописному виду, т.е. в виде заглавных букв (рис. 2).

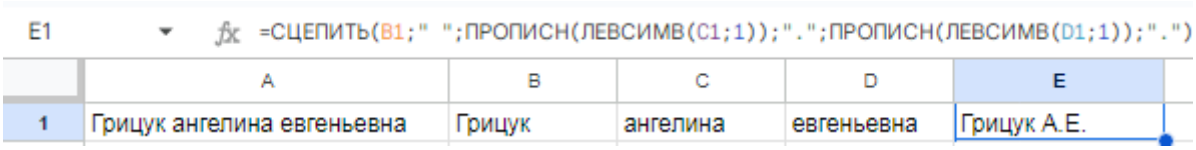

Рисунок 2 – Пример формулы для выполнения второго этапа.

Таким образом может быть получен список фамилий с заглавными буквами инициалов, что соответствует требованиям поставленной задачи.

Рассмотрим подход, реализующий автоматизацию процесса выполнения представленных действий. Для программирования выберем язык C++.

Исходные данные для обработки представлены в текстовом файле ELB_list.txt. Для корректной работы программы необходимо подключить заголовочные файлы, входящие в библиотеку <iostream> для обработки ввода-вывода в консоль; <cstring> для подключения функций обработки строк в C++ (<string.h> для C); <fstream> для работы с файловыми потоками и <clocale> для поддержки локализации параметров, в частности для использования букв русского алфавита.

Для выполнения алгоритма в программной реализации исходный файл ELB_list.txt открывается для чтения, а файл rezult.txt – для записи полученного результата действий. При просмотре первого файла построчно считывается информация в текстовом формате в переменную string для выполнения преобразований до тех пор, пока не будет достигнут конец файла.

Чтобы выполнить изменение считанной информации была разработана функция transform(), в которой для решения проблемы используются два указателя: один устанавливается на начало имени, второй – на начало отчества. На первом шаге определяются положения указателей. Для реализации этой цели был использован стандартный подход, в котором при просмотре каждого элемента массива string, происходит сравнение его кода с кодом символа «пробел». При равенстве кода просматриваемого символа и кода символа пробела устанавливается указатель на следующий элемент, который закономерно является началом имени. При повторном обнаружении пробела устанавливается указатель на следующий символ, являющийся началом отчества (рис. 3). Стоит заметить, что для достижения поставленной цели можно было также использовать функцию strchr().

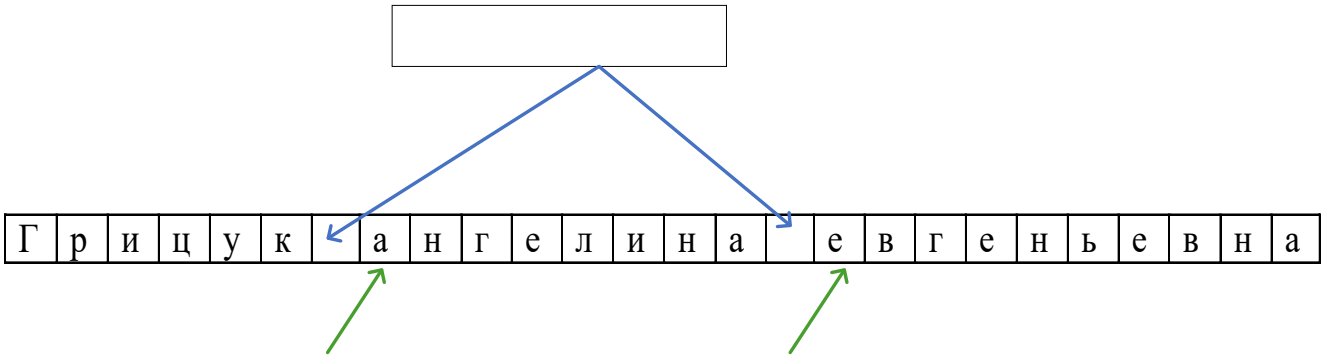

Рисунок 3 – Механизм установления указателей на инициалы.

После установления позиций инициалов на базе текущей строки создается новая строка по правилам составления ФИО в форме «Фамилия И.О.» (рис. 4).

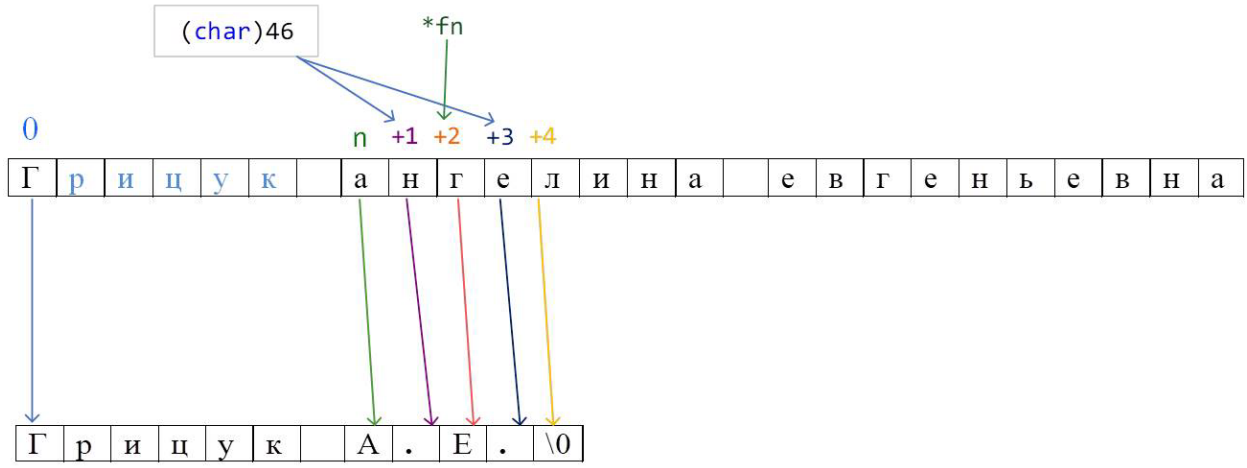

Рисунок 4 – Схема создания результирующей строки.

Если просматриваемый символ является началом строки (т.е. началом фамилии), то он по умолчанию преобразуется к прописному символу при помощи функции toupper(). Все остальные символы до указателя на первую букву имени переносятся без изменения.

Далее компонуется группа инициалов. На позицию, совпадающую с указателем на первую букву имени, устанавливается она же, преобразованная к прописному виду. На следующую позицию устанавливается символ «точка», за ним устанавливается символ, на который установлен указатель, соответствующий первой букве отчества, также преобразованный к прописному виду. Затем опять устанавливается символ «точка», а оставшиеся символы строки отсекаются с помощью нуль-терминатора.

Полученная строка является ожидаемым результатом работы функции (рис. 5).

Рисунок 5 – Демонстрация результата работы программы.

Стоит отметить, что данный алгоритм работает только с ФИО самой распространенной конфигурации: три последовательные однословные компоненты. Например, в случае ввода строки, которая содержит сложное имя, имеющее дефисное написание, результат выполнения программы не будет корректным, поскольку результирующая строка будет содержать единственный инициал имени при наличии в исходном виде двух букв имени. Таким образом, функция transform() для обработки С-строк требует дополнительного анализа и преобразований.

Список литературы

1. Большие Данные [Электронный ресурс]. – Режим доступа: https://bigdata-sfedu.github.io/ – Дата доступа: 08.11.2023.

УДК 681.3:004.89

ПРИМЕНЕНИЕ АВТОЭНКОДЕРНЫХ СЕТЕЙ ДЛЯ ВОССТАНОВЛЕНИЯ ИЗОБРАЖЕНИЙ

Д. А. Дворанинович, И. Н. Аверина
Брестский государственный технический университет, г. Брест
Научный руководитель: А. Л. Михняев

В статье рассматривается возможность применения автоэнкодерных нейронных сетей для восстановления изображений.

Вопрос восстановления изображений в настоящее время достаточно актуален. Хотя большая часть информации, благодаря современным технологиям, получается без видимых искажений, существуют области, где восстановление информации с высокой точностью является важной прикладной задачей. К примеру, восстановление старых данных, а именно: старых аудиозаписей, изображений и прочего, когда на момент создания файлов техника не позволяла без помех записать некоторую информацию. Кроме того особо актуально на сегодняшний день получение точной информации от исследовательских станций в космическом пространстве, когда сигнал может значительно искажаться.

Для задач восстановления изображений давно применяются помехозащитное кодирование и разного рода методы фильтрации. Тем не менее, интересным является иной способ восстановления информации – нейросетевой, с помощью автоэнкодерных нейронных сетей. Так, актуальным является применение автоэнкодерных нейросетей при обработке изображений, например, в задачах