Самостоятельная 1, модуль 2. Обработка текстовых данных CSV-файлов

Вариант 1. Неизученные свойства покемонов

Подготовьте программу обработки данных, получаемых из набора данных о покемонах. Первичные данные для работы программы находятся в файле **Pokemon.csv (https://www.kaggle.com/datasets/abcsds/pokemon?resource=download)**.
Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о покемонах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **Pokemon.csv**.
2. Выводить на экран из набора исходных данных информацию о всех покемонах, имеющих свойство *Type 1 = Poison*.
    a. Сохранять в файл **Pokemon-Poison.csv** выборку о всех покемонах, имеющих свойство *Type 1 = Poison.*
3. Выводить на экран переупорядоченный набор исходных данных о покемонах, в котором выделены группы по полю основной тип (*Type 1*), при этом в каждой группе следует упорядочить покемонов по возрастанию атаки (*Attack*). Перед упорядоченным набором должна выводиться строка с указанием модуля разницы (дельту) между максимальной и минимальной атакой в группе покемонов *Type 1 = Dark*.
    a. Сохранять результат переупорядочения в файле **Sorted-Pokemon.csv** с сохранением порядка.
4. Выводить на экран выборку покемонов, не имеющих дополнительных особенностей (*Type 2* отсутствует / пуст). Перед выводом перечня покемонов должно выводиться среднее значение здоровья (*HP*) покемонов этой выборки.
    a. Сохраните выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество строк с данными о покемонах (без учёта заголовков)
    b. Количество групп покемонов по типам (свойство *Type 1*) и количество покемонов в каждой группе.
    c. Количество покемонов привидений (*Gost*) в каждой из групп покемонов по свойству *Type 1*. Обратите внимание, что покемоны могут быть привидениями по разным свойствам *Type*.
    d. Количество летающих (*Flying*) покемонов по основному типу *Type 1*, которые являются драконами (*Dragon*) по типу *Type 2*.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.

2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 2. Легенда о покемонах

Подготовьте программу обработки данных, получаемых из набора данных о покемонах. Первичные данные для работы программы находятся в файле **Pokemon.csv** (https://www.kaggle.com/datasets/abcsds/pokemon?resource=download).
Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о покемонах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **Pokemon.csv**.
2. Выводить на экран из набора исходных данных информацию о группах покемонов, являющихся легендарными и нелегендарными (легендарность устанавливается по полю *Legendary*).
    a. Сохранять в файлы **Pokemon-Legendary.csv** и **Pokemon-Usual.csv** списки легендарных и нелегендарных покемонов соответственно..
3. Выводить на экран список покемонов, относящихся к одному и тому же поколению (*Generation*). Перед каждой группой поколения выводить строку с данными о самом мощном покемоне (*Attack* в рамках поколения).
    a. Сохранять перечень покемонов поколения в файле **Pokemon-Gen-N.csv,** где *N* - номер поколения.
4. Выводить на экран выборку покемонов, скорость которого либо не более, чем на 10 единиц меньше максимального значения скорости среди всех покемонов.
    a. Сохраните выборку в файл **Speed-Pokemon.csv**
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество покемонов по поколениям (*Generation*).
    b. Полные данные о самом мощном (*Attack*) и самом слабом покемоне среди всех покемонов, загруженных из файла.
    c. Количество ядовитых (*Poison*) жуков (*Bug*). Обратите внимание, что эти свойства могут быть выражены разными полями *Type*.
    d. Количество покемонов второго поколения, у которых защита меньше 50.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 3. Потомственные студенты

Подготовьте программу обработки данных, получаемых из набора данных о студентах. Первичные данные для работы программы находятся в файле **student_data.csv** (https://www.kaggle.com/datasets/devansodariya/student-performance-data?resource=download). Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о студентах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **student_data.csv**.
2. Выводить на экран информацию о всех студентах, чьи родители учились на оценки выше или равные 4 (столбцы *Medu* и *Fedu*).
    a. Сохранять в файл **Student-Parents-Education.csv** выборку о всех студентах, чьи родители подходят под вышеуказанное условие.
3. Выводить на экран информацию о всех студентах, количество пропусков(столбец absences) у которых находится в диапазоне [min + 10, max - 10], где min - минимальное количество пропусков в выборке, max - максимальное.
    a. Сохранять в файл **Students-Absences.csv** выборку о всех студентах, количество пропусков у которых находится в вышеуказанном диапазоне.
4. Выводить на экран исходный набор данных о студентах, сгруппированный по столбцу reason, при этом в каждой группе следует упорядочить студентов по убыванию значений столбца age.
    a. Сохранять в файл **Grouped-Students.csv** результат группировки с сохранением порядка.
5. Выводить сводную статистику по данным загруженного файла:
    a. Общее количество студентов.
    b. Статистику по столбцу *romantic* в процентном отношении.
        i. Пример для понимания: yes 63% no 37%.
    c. Средний возраст девушек у которых хотя бы один из родителей работает из дома(столбцы *Mjob* и *Fjob*).
    d. Медиану возраста студентов(столбец *age*).

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 4. “От сессии до сессии…”

Подготовьте программу обработки данных, получаемых из набора данных о студентах. Первичные данные для работы программы находятся в файле **student_data.csv** (https://www.kaggle.com/datasets/devansodariya/student-performance-data?resource=download). Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о студентах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **student_data.csv**.
2. Выводить на экран информацию о всех студентах, у которых средняя оценка за три периода выше одиннадцати(столбцы *G*1, *G*2, *G*3).
    a. Сохранять в файл **Student-Grades.csv** выборку о всех студентах, чьи оценки удовлетворяют вышеуказанному условию.
3. Выводить на экран информацию о всех студентах у которых значение столбца *freetime* превышает значение столбца *studytime* и при этом количество пропусков(столбец absences) не более семи.
    a. Сохранять в файл **Students-Time.csv** выборку о всех студентах, удовлетворяющих вышеуказанному условию.
4. Выводить на экран исходный набор данных о студентах, сгруппированный по столбцу reason, при этом в каждой группе следует упорядочить студентов по возрастанию значений столбца *absences*.
    a. Сохранять в файл **Grouped-Students.csv** результат группировки с сохранением порядка.
5. Выводить сводную статистику по данным загруженного файла:
6. Общее количество студентов.
7. Статистику по столбцу reason в количественном отношении.
    a. Пример для понимания: home 120 course 214 reputation 97.
8. Средний возраст юношей(столбец *age*) у которых опекуном является мама(столбец *guardian*).
9. Медиану количества пропусков(столбец *absences*).

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 5. Отличный денёк для рыбалки в Австралии

Так-то тонкостей и деталей в выборе рыболовной погоды очень много, но этого для начала хватит.

Подготовьте программу обработки данных, получаемых из набора данных о погоде в Австралии. Первичные данные для работы программы находятся в файле **weatherAUS.csv (**https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package**).**

Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о погодных условиях в Австралии. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **weatherAUS.csv**.
2. Выводить на экран из набора исходных данных информацию о погоде собранной в Сиднее (*Location = Sydney*) за 2009 и 2010 год.
    a. Сохранять в файл **Sydney_2009_2010_weatherAUS.csv** выборку о погоде за 2009-2010 гг.
3. Выводить на экран переупорядоченный набор исходных данных о записях, в котором выделены группы по месту расположения станции сбора метеоданных (*Location*), при этом в каждой группе следует упорядочить записи по убыванию осадков (*Rainfall*). Перед каждой группой должна выводиться строка с указанием среднего арифметического показателей осадков в данной локации.
    a. Сохранять результат переупорядочения в файле **average_rain_weatherAUS.csv** с сохранением порядка групп и записей внутри этих групп. Средние арифметические записывать в файл не надо.
4. Выводить на экран выборку записей дней, когда солнечная погода держалась хотя бы 4 часа за день (*Sunshine >= 4*). Перед выводом перечня записей выведите Дату (*Date*) и Максимальную температуру (*MaxTemp*) которая была в день, когда солнце светило дольше всего.
    a. Сохраните только выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Количество дней пригодных для рыбалки, то есть когда скорость ветра днем была меньше 13 км/ч (*WindSpeed3pm < 13*).
        i. Опционально (если нет то на оценку не повлияет): отфильтровать по направлению ветра(*WinDir3pm*), чтобы из направлений были только W, *WSW*, *SW*, *SSW*, *S*.
    b. Количество групп записей если группировать их по локации (*Location*) и количество записей в каждой группе.
    c. Количество теплых дождливых дней. День считается теплым, если максимальная температура была больше либо равна 20 градусам Цельсия (*MaxTemp*). И дождливым, если в этот день был дождь (*RainToday = Yes*).
    d. Количество дней с нормальным атмосферным давлением с утра. Атмосферное давление будем считать нормальным если оно находится в пределах [1000; 1007] КПа (1000 <= *Pressure9am <= 1007*)

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 6. Ищем теплые денёчки

Подготовьте программу обработки данных, получаемых из набора данных о погоде в Австралии. Первичные данные для работы программы находятся в файле **weatherAUS.csv** (https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package).

Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о погодных условиях в Австралии. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **weatherAUS.csv**.
2. Выводить на экран из набора исходных данных информацию о погоде собранной в Мельбурне (*Location = Melbourne*) за 2015 и 2016 год.
    a. Сохранять в файл **Melbourne_2015_2016_weatherAUS.csv** выборку о погоде за 2015-2016 гг.
3. Выводить на экран переупорядоченный набор исходных данных о записях, в котором выделены группы по месту расположения станции сбора метеоданных (*Location*), при этом в каждой группе следует упорядочить записи по убыванию максимальной температуры (*MaxTemp*). Перед каждой группой должна выводиться строка с указанием среднего арифметического показателей максимальной температуры в данной локации.
    a. Сохранять результат группировки в файле **average_max_temp_weatherAUS.csv** с сохранением порядка групп и записей внутри этих групп. Средние арифметические записывать в файл не надо.
4. Выводить на экран выборку почти безоблачных дней, когда облака и утром и вечером занимали максимум 3/8 неба (*Cloud9am <= 3, Cloud3pm <= 3*). Перед выводом перечня записей выведите Дату (*Date*) и сколько светило солнце (*Sunshine*) в любой из дней, когда на небе было ни облачка (*Cloud9am = 0 & Cloud3pm = 0*).
    a. Сохраните только выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество идеально-безоблачных дней (*Cloud9am = 0 & Cloud3pm = 0*).
    b. Полные данные о записи с самым большим испарением (*Evaporation*).
    c. Среднее давление в КПа (*(Pressure9am + Pressure3pm) / 2*) среди всех идеально-безоблачных дней (*Cloud9am = 0 & Cloud3pm = 0*) и среднее давление в КПа (*(Pressure9am + Pressure3pm) / 2*) среди всех идеально-очень-облачных дней (*Cloud9am = 8 & Cloud3pm = 8*).
    d. Количество идеальных дней для купания. Идеальным днем для купания считаем день с максимальной температурой >= 28 градусов Цельсия (*MaxTemp*) и время которое за день должно светить солнце >= 7 (*Sunshine*)

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.

2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 7. Back in the game

Подготовьте программу обработки данных, получаемых из набора данных о компьютерных играх. Первичные данные для работы программы находятся в файле **computer_games.csv** (https://www.kaggle.com/datasets/iamsouravbanerjee/computer-games-dataset).

Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о компьютерных играх. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **computer_games.csv**.
2. Выводить на экран из набора исходных данных информацию о всех компьютерных играх, имеющих свойство *Developer = Maxis*
    a. Сохранять в файл **Developer_Maxis.csv** выборку о всех компьютерных играх, имеющих свойство Developer = Maxis.
3. Выводить на экран переупорядоченный набор исходных данных об играх, в котором выделены группы по *Operating System* (а именно - *Microsoft Windows* и *Microsoft Windows, macOS*, последняя пара может быть указана в произвольном порядке), при этом в каждой группе следует упорядочить игры по возрастанию *Date Released*. Перед упорядоченным набором должна выводиться строка с указанием модуля разницы (дельту) между максимальной и минимальной датой релиза (*Date Released*).
   Обратите внимание на формат данных *Date Released*, его нужно привести к единому виду (можно делать, опираясь только на год релиза)
    a. Сохранять результат переупорядочения в файлы **Microsoft_Windows_Sort.csv** и **Microsoft_Windows_Sort.csv** с сохранением порядка.
4. Вывести на экран выборку игр, год релиза которых больше среднего значения Date Released.
    a. Сохраните выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество игр по каждому производителю (producer)
    b. Данные о самой ранней выпущенной игре, причем *Operating System = Microsoft Windows*, а жанр *Genre = First-person shooter*.
    c. Информация о среднем значении года выпуска игры (*Date Released*) по жанрам (*Genre*). Иначе говоря, для каждого жанра найти среднее значение года выхода компьютерных игр, которые с ним связаны)
    d. Продюсер, который связан с минимальным количеством компьютерных игр.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 8. Релизы и жанры

Подготовьте программу обработки данных, получаемых из набора данных о компьютерных играх. Первичные данные для работы программы находятся в файле **computer_games.csv** **(**https://www.kaggle.com/datasets/iamsouravbanerjee/computer-games-dataset**)**.

Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о компьютерных играх. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **computer_games.csv**.
2. Выводить на экран из набора исходных данных информацию о всех компьютерных играх, дата выхода которых позже 1997 года.
    a. Сохранять в файл **Date_Released.csv** выборку о всех компьютерных играх, имеющих свойство *Date Released* больше 1997.
3. Выводить на экран переупорядоченный набор исходных данных об играх, в котором выделены группы по жанру (*Genre*), при этом в каждой группе следует упорядочить игры по убыванию *Date Released*. Перед упорядоченным набором должна выводиться строка с указанием модуля разницы (дельту) между максимальной и минимальной датой релиза (*Date Released*).
   Обратите внимание на формат данных *Date Released*, его нужно привести к единому виду (можно делать, опираясь только на год релиза)
    a. Сохранять результат переупорядочения в файлы **Game_Sort.csv** с сохранением порядка.
4. Вывести на экран выборку игр, жанр которой - *First-person shooter*, операционная система (*Operating System*) - *Microsoft Windows, macOS*, а год выхода позже 2003.
    a. Сохраните выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество строк с данными о компьютерных играх (без учёта заголовков)
    b. Данные об игре, выпущенной перед последней, причем *Developer = Maxis* , а *Producer = Electronic Arts*.
    c. Информация о среднем значении года выпуска игры (*Date Released*) по производителю (*producer*). Другими словами, для каждого продюсера найти среднее значение года выхода компьютерных игр, которые с ним связаны).
    d. Год, когда было выпущено наибольшее количество компьютерных игр.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 9. Что мы знаем о Spotify-исполнителях?

Подготовьте программу обработки данных, получаемых из набора данных о музыкантах. Первичные данные для работы программы находятся в файле **spotify_artist_data.csv (**https://www.kaggle.com/datasets/adnananam/spotify-artist-stats**)**.
Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о артистах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **spotify_artist_data.csv.**
2. Выводить на экран из набора исходных данных информацию о всех артистах, имеющих более 200 треков.
    a. Сохранять в файл **Artists.csv** выборку о всех артистах, имеющих более 200 треков.
3. Выводить на экран упорядоченный набор исходных данных об артистах, в котором выделены группы по полю миллиард прослушиваний (*One Billion*), при этом в каждой группе следует упорядочить исполнителей по 100 миллионам прослушиваний (100 *Million*).
    a. Сохранять результат упорядочения в файле **Sorted-Artists.csv** с сохранением порядка.
4. Выводить на экран выборку *id* (*Serial Number*), выпадающих из анализа, так как данные в этих строках некорректны.
    a. Сохраните выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество строк с данными об артистах (без учета заголовков)
    b. Количество групп артистов по количеству прослушиваний (5 групп) и количество артистов в каждой группе.
    c. Количество артистов имеющих больше чем 2 слова в названии группы.
    d. Количество артистов миллионников, имеющих более 100 треков.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 10. Неужели кто-то это слушает?

Подготовьте программу обработки данных, получаемых из набора данных о музыкантах. Первичные данные для работы программы находятся в файле **spotify_artist_data.csv (**https://www.kaggle.com/datasets/adnananam/spotify-artist-stats**)**.
Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о артистах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **spotify_artist_data.csv.**
2. Выводить на экран из набора исходных данных информацию о всех артистах, имеющих 0 *Feats*.
    a. Сохранять в файл **Artists.csv** выборку о всех артистах, имеющих 0 *Feats*.
3. Выводить на экран список артистов, последнее изменение у которых произошли в один день (сгруппировать по дню изменения).
    a. Сохранять перечень в файле **Artists-day.csv.**
4. Выводить на экран выборку артистов, количество прослушиваний которых достигает от 40 до 60 процентов от максимума.
    a. Сохраните выборку в файл **Meed-artist.csv**
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество артистов, исключая строки непригодные для анализа.
    b. Полные данные о самом прослушиваемом артисте и наименее прослушиваемом артисте.
    c. Количество артистов, имеющих в названии цифры.
    d. Количество артистов миллиардников, менее чем с 150 треками.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 11. Студент ради зачёта идет на всё, даже на занятия

Подготовьте программу обработки данных, получаемых из набора данных об успеваемости студентов. Первичные данные для работы программы находятся в файле **exams.csv (**https://www.kaggle.com/datasets/whenamancodes/students-performance-in-exams**)**. Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные об успеваемости. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **exams.csv**.
2. Выводить на экран из набора исходных данных информацию о всех студентах, имеющих свойство *test preparation course = completed*.
    a. a. Сохранять в файл **Test_Preparation.csv** выборку о всех студентах, имеющих свойство *test preparation course = completed.*
3. Выводить на экран переупорядоченный набор исходных данных о студентах, в котором
   выделены группы по типу обеда (*standard* и *free/reduced*), при этом в каждой группе следует упорядочить студентов по возрастанию оценки по математике (*math score*). Перед упорядоченным набором должна выводиться строка с указанием модуля разницы (дельты) между максимальной и минимальной оценкой по математике.
    a. a. Сохранять результат переупорядочения в файле **Sorted_Students.csv** с сохранением порядка.
4. Выводить на экран выборку студентов, которые являются женщинами (*female*), посчитать для каждой строки среднюю оценку за все экзамены (*math score*, *reading score, writing score*)
    a. a. Сохраните выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество строк с данными о студентах (без учёта заголовков)
    b. Количество групп, к которым относятся студенты (*race/ethnicity*) и количество людей в каждой группе.
    c. Количество студентов, которые сдали экзамены (*math score*, *reading score*, *writing score*) больше, чем на 50 баллов. Статистику необходимо вывести по каждому экзамену.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 12. Первая сессия

Подготовьте программу обработки данных, получаемых из набора данных об успеваемости студентов. Первичные данные для работы программы находятся в файле **exams.csv (**https://www.kaggle.com/datasets/whenamancodes/students-performance-in-exams**)**. Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные об успеваемости. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **exams.csv**.
2. Выводить на экран из набора исходных данных информацию о всех студентах, имеющих свойство уровень образования родителей *(parental level of education) = либо high school, либо some college.*
3. Выводить на экран переупорядоченный набор исходных данных о студентах, в котором выделены группы по типу *test preparation course*, при этом в каждой группе следует упорядочить студентов по группе (*race* / *ethnicity*) в лексикографическом порядке (например, group **A**, затем group **B**…).
    a. Сохранять результат переупорядочения в файле **Sorted_Students.csv** с сохранением порядка.
4. Выводить на экран выборку студентов, у которых test preparation *course = completed* и
   баллы за каждый экзамен больше 60, посчитать их количество.
    a. Сохраните выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество студентов по группам (*race* / *ethnicity*)
    b. Студент (студенты, если есть кто-то одинаковым минимальным баллом) с самым большим суммарным баллом за все экзамены *(math score, reading score, writing score*), аналогично с минимальным баллом.
    c. Количество мужчин (*gender = male*), которые входят в диапазон сдачи экзамена по математике (*math score*) от 0 до 100 с шагом 10.

Требования к работе со входными и выходными данными

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 13. Сколько стоит дата-сайнс?

Подготовьте программу обработки данных, получаемых из набора данных о зарплатах в секторе наук о данных. Первичные данные для работы программы находятся в файле **Data_Science_Fields_Salary_Categorization.csv (https://www.kaggle.com/datasets/whenamancodes/data-science-fields-salary-categorization)**. Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о работниках сферы наук о данных. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **Data_Science_Fields_Salary_Categorization.csv.**
2. Выводить на экран из набора исходных данных информацию о всех работниках больших (*L*) компаний.
    a. Сохранять эту выборку в файл **Large-Workers.csv**.
3. Выводить на экран упорядоченный набор исходных данных о работниках, в котором выделены группы по полю *Experience*, при этом в каждой группе следует упорядочить работников по возрастанию зарплаты.
    a. Сохранять результат упорядочения в файле **Sorted-DS.csv** с сохранением порядка.
4. Выводить на экран выборку работников, место работы которых не совпадает с местом расположения компании.
    a. Сохраните выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество строк с данными о Data-scientist-ах
    b. Количество групп работников по *Designation* и количество работников в каждой группе.
    c. Количество работников *Remote_Working_Ratio* которых находится в диапозоне [50, 60]
    d. Количество средних компаний, зарплата которых на 25% больше средней.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 14. Опытные инженеры данных

Подготовьте программу обработки данных, получаемых из набора данных о зарплатах в секторе наук о данных. Первичные данные для работы программы находятся в файле **Data_Science_Fields_Salary_Categorization.csv (https://www.kaggle.com/datasets/whenamancodes/data-science-fields-salary-categorization)**. Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о работниках сферы наук о данных. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла
2. Выводить на экран из набора исходных данных информацию о группах работников по Experience.
    a. Сохранять в файл **workers.csv** группируя по опыту работы.
3. Выводить на экран список работников, группируя их по году работы, для каждой группы перед ней указать самую большую зарплату в группе.
    a. Сохранять перечень работников по годам в файле **Workers-N.csv,** где *N* - номер года.
4. Выводить на экран выборку работников, зарплата которых находится в диапазоне от 70 до 80% от максимальной.
    a. Сохраните выборку в файл **Salary-workers.csv**
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество строк с данными
    b. Работников с наибольшей и наименьшей зарплатой.
    c. Количество *Data Engineer* работающих из *GB*.
    d. Количество работников работающих в компаниях из *GB*, но работающих из иной страны, перед каждым из них написать из какой страны он работает.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 15. В Кейптаунском порту…

Подготовьте программу обработки данных, получаемых из набора данных о портах. Первичные данные для работы программы находятся в файле **Port_Data.csv (**https://www.kaggle.com/datasets/sanjeetsinghnaik/ship-ports**)**.
Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о портах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **Port_Data.csv**.
2. Выводить на экран из набора исходных данных информацию о всех портах, имеющих свойство *Country = USA*.
    a. Сохранять в файл **Port_Data-Country.csv** выборку о всех портах, имеющих свойство *Country = USA*.
3. Выводить на экран переупорядоченный набор исходных данных о портах, в котором выделены группы по полю основной тип (*Country*), при этом в каждой группе следует упорядочить порты по возрастанию индекса (*#*). Перед упорядоченным набором должна выводиться строка с указанием модуля разницы (дельта) между максимальным и минимальным индексом в группе портов *Country = China*.
    a. Сохранять результат переупорядочения в файле **Sorted-Port_Data.csv** с сохранением порядка, полученного при сортировке.
4. Выводить на экран выборку покемонов, не имеющих дополнительных особенностей (*UN Code* отсутствует / пуст). Перед выводом перечня портов должно выводиться среднее значение судов (*Vessels in Port*) портов этой выборки.
    a. Сохраните выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество строк с данными о портах (без учёта заголовков)
    b. Количество групп портов по типам (свойство *Area Local*) и количество портов в каждой группе.
    c. Количество портов *Anchorage* (свойство *Type*) в каждой из групп портов по свойству *Area Local*.
    d. Количество портов в жёлтом море *(Yellow Sea)* по типу *Local Area,* которые являются *Anchorage* по типу *Type*.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 16. Как провожают пароходы?

Подготовьте программу обработки данных, получаемых из набора данных о портах. Первичные данные для работы программы находятся в файле **Port_Data.csv (**https://www.kaggle.com/datasets/sanjeetsinghnaik/ship-ports**)**.
Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о покемонах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **Port_Data.csv**.
2. Выводить на экран из набора исходных данных информацию о группах портов, являющихся Port и Anchorage (устанавливается по полю *Type*).
    a. Сохранять в файлы **Port-Port.csv** и **Port-Anchorage.csv** списки *Port* и *Anchorage* портов соответственно..
3. Выводить на экран список портов, относящихся к одной и той же стране (*Country*). Перед каждой группой поколения выводить строку с данными о порте, у которого больше всего судов (*Vessels in Port **в рамках страны***).
    a. Сохранять перечень портов в файле **Port-Country-N.csv,** где *N* - название страны.
4. Выводить на экран выборку портов, *Departures* которых не более, чем на 10 единиц меньше максимального значения *Departures* среди всех портов.
    a. Сохраните выборку в файл **Departures-Port.csv**
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество портов по странам (*Country*).
    b. Полные данные о самом загруженном судами портом (*Vessels in Port*) и самом менее загруженном порте среди всех портов, загруженных из файла.
    c. Количество *Yellow Sea* (свойство *Area Local*) *Bohai Sea* (свойство *Area Local*).
    d. Количество портов страны *China*, у которых *Arrivals* меньше 500.

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 17. С оглядкой на обратную сторону шара

Подготовьте программу обработки данных, получаемых из набора данных о зарплатах дата-сайентистов из разных стран. Первичные данные для работы программы находятся в файле **ds_salaries.csv** (https://www.kaggle.com/datasets/ruchi798/data-science-job-salaries?resource=download).

Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о зарплатах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **ds_salaries.csv**.
2. Выводить на экран из набора исходных данных информацию о всех зарплатах, имеющих свойство *employee_residence = US*.
    a. Сохранять в файл **US_ds_salaries.csv** выборку о всех зарплатах, имеющих свойство employee_residence = *US.*
3. Выводить на экран переупорядоченный набор исходных данных о зарплатах, в котором выделены группы по полю страна пребывания работника (*employee_residence*), при этом в каждой группе следует упорядочить покемонов по возрастанию зарплаты в долларах (*salary_in_usd*). Перед упорядоченным набором должна выводиться строка с указанием модуля разницы (дельту) между максимальной и минимальной зарплатой в группе с *employee_residence = CA*.
    a. Сохранять результат переупорядочения в файле **sorted_ds_salaries.csv** с сохранением порядка.
4. Выводить на экран выборку зарплат, для работников не на полной занятости (*employment_type* не равен *FT*). Перед выводом перечня зарплат должно выводиться среднее значение зарплаты в долларах (*salary_in_usd*) в этой выборке.
    a. Сохраните только выборку в CSV-файл с именем, полученным от пользователя.
5. Выводить на экран сводную статистику по данным загруженного файла:
    a. Общее количество строк с данными о зарплатах (без учёта заголовков)
    b. Количество групп зарплат разбитых по странам пребывания работника (свойство *employee_residence*) и количество записей о зарплатах в каждой группе.
    c. Количество записей с *job_title = Data Engineer* в каждой из групп зарплат по свойству *employee_residence*. Обратите внимание, что дата инженеры могут жить в любой стране.
    d. Количество резидентов, проживающих в США (*employee_residence = US)*, которые работают не на компанию из США (*company_location != US*).

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.

3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 18. В мире джунов с Николаем Дроздовым

Здравствуйте дорогие друзья! Сегодня мы посмотрим на удивительный мир джунов, их повадки и основные места обитания.

Подготовьте программу обработки данных, получаемых из набора данных о зарплатах дата-сайентистов из разных стран. Первичные данные для работы программы находятся в файле **ds_salaries.csv** (https://www.kaggle.com/datasets/ruchi798/data-science-job-salaries?resource=download).

Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные о зарплатах. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **ds_salaries.csv**.
2. Выводить на экран из набора исходных данных информацию о двух группах:
   1) зарплаты программистов уровня джуниор (*experience_level = EN*)
   2) зарплаты программистов уровня миддл (*experience_level = MI*).
      a. Сохранять в файлы **junior_ds_salaries.csv** и **middle_ds_salaries.csv** списки зарплат джунов и зарплат миддлов соответственно.
3. Выводить на экран разбитый на группы по профессиональному направлению (*job_title*) список записей зарплат. Перед каждой группой зарплат выводить строку с данными о самой маленькой зарплате в группе. Фильтровать списки по уровню профессионализма не надо, то есть *experience_level может быть любой.*
      a. Сохранить список групп в файле **min_ds_salaries.csv,** где *N* - номер поколения.
4. Выводить на экран среднее значение зарплат джунов (*experience_level = EN*), которые:
   1) работают в маленьких компаниях (*company_size = S*).
   2) работают в средних компаниях (*company_size = M*).
   3) работают в крупных компаниях (*company_size = L*).
      a. Сохраните средние значения в файл **avg_jun_company_size_ds_salaries.csv**
5. Выводить на экран сводную статистику по данным загруженного файла:
      a. Общее количество записей о зарплатах для разных профессиональных направлений (*job_title*).
      b. Полные данные о записи с самой большой зарплатой и о записи с самой маленькой зарплатой, загруженных из файла.
      c. Количество джунов (*experience_level = EN*) занимающихся Data Science (*job_title = Data Scientist*).
      d. Количество джунов которые полностью работают удаленно (*remote_ratio = 100*).

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 19. Ловцы НЛО

Подготовьте программу обработки данных, получаемых из набора данных о НЛО. Первичные данные для работы программы находятся в файле **ufo_sighting_data.csv** (https://www.kaggle.com/datasets/camnugent/ufo-sightings-around-the-world?resource=download) Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные об НЛО. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **ufo_sighting_data.csv**.
2. Выводить на экран информацию о всех датах, в которые видели НЛО треугольной формы(столбец *UFO shape*) более 1000 секунд (столбец *length of encounter seconds*).
    a. Сохранять в файл **UFO-Shape-Time.csv** выборку о всех датах, удовлетворяющих вышеуказанному условию.
3. Выводить на экран исходный набор данных об НЛО, сгруппированный по столбцу *city*, при этом в каждой группе стоит упорядочить записи по возрастанию широты(столбец *latitude*), а также исключить все записи, сделанные во вторник.
    a. Сохранять в файл **Grouped-UFOs.csv** результат группировки с сохранением порядка.
4. Выводить на экран информацию о всех записях, в которых НЛО треугольной, цилиндрической или круглой формы(столбец *UFO shape*) видели в период между 20:00 и 6:30 (столбец *Datetime*).
    a. Сохранять в файл **UFOs-Schedule.csv** выборку о всех записях, удовлетворяющих вышеуказанному условию.
5. Выводить сводную статистику по данным загруженного файла:
    a. Общее количество записей о НЛО.
    b. Статистику по формам НЛО(столбец *UFO shape*) в процентном отношении.
        i. Пример для понимания: Треугольник 34% Цилиндр 18% Сфера 48%.
    c. Среднюю продолжительность времени наблюдения(столбец *length of encounter seconds*) НЛО треугольной формы(столбец *UFO shape*).
    d. Медиану широт(столбец *latitude*).

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.

Вариант 20. Секретные материалы

Подготовьте программу обработки данных, получаемых из набора данных об НЛО. Первичные данные для работы программы находятся в файле **ufo_sighting_data.csv** (https://www.kaggle.com/datasets/camnugent/ufo-sightings-around-the-world?resource=download)
.
Программа должна предоставлять пользователю текстовое меню, позволяющее обратиться к следующим операциям:

1. Вводить адрес файла, из которого загружаются данные об НЛО. Обратите внимание, что название файла может быть произвольным, структура файла должна совпадать со структурой первичного файла **ufo_sighting_data.csv**.
2. Выводить на экран информацию о всех датах, в которые в Лос-Анджелесе или Сиэтле (столбец *city*) видели НЛО цилиндрической или сферической формы (столбец *UFO shape*).
    a. Сохранять в файл **UFO-City-Shape.csv** выборку о всех датах, удовлетворяющих вышеуказанному условию.
3. Выводить на экран исходный набор данных об НЛО, сгруппированный по столбцу *UFO shape*, при этом в каждой группе стоит упорядочить записи по возрастанию времени записи (столбец *Datetime*), а также исключить все записи, сделанные о НЛО формы огненного шара (столбец *UFO shape*).
    a. Сохранять в файл **Grouped-UFOs.csv** результат группировки с сохранением порядка.
4. Выводить на экран информацию о всех записях, в которых НЛО видели в Финиксе, Портленде или Сиэтле (столбец *city*) в период между 10:00 и 19:15 (столбец *Datetime*).
    a. Сохранять в файл **UFOs-Schedule.csv** выборку о всех записях, удовлетворяющих вышеуказанному условию.
5. Выводить сводную статистику по данным загруженного файла:
    a. Общее количество записей о НЛО.
    b. Статистику по городам(столбец *city*) в процентном соотношении.
    c. Пример для понимания: Нью-Йорк 34% Лас-Вегас 18% Сиэтл 48%.
    d. Среднюю продолжительность времени наблюдения(столбец *length of encounter seconds*) НЛО в Лас-вегасе (столбец *city*).
    e. Медиану долгот (столбец *longitude*).

Требования к работе со входными и выходными данными:

1. Программа должна сохранять работоспособность при вводе некорректных адресов и имён файлов, с учётом различных платформ запуска файлов.
2. Некорректно структурированный файл программой не обрабатывается, пользователю выводится сообщение об ошибке и предлагается вернуться к основному меню.
3. Программа обязательно должна корректно открывать созданные ей файлы и позволять выполнять над ними все операции из меню.
4. Программа автоматически разбирается с кодировками файла и отображает данные на экран и в файлы человекочитаемом виде.