Лабораторная работа. Утилита WordCount

Задача.

Реализовать утилиту *WordCount* подсчитывающую количество строк, слов и байт для указанного файла и выводить эту информацию в поток вывода.

Программа должна поддерживать следующие опции:

-l, --lines вывод только количества строк
-c, --bytes вывод размера файла в байтах
-w, --words вывод количества слов

Название файла и опции передаются через аргументы командной строки в следующем формате:

WordCont.exe [OPTION] filename

Примечание.

1. Для реализации утилиты потребуется воспользоваться стандартной библиотекой вводы\вывода (описание).
2. Пример того, как можно организовать парсинг аргументов командной строки, можно посмотреть здесь.

Лабораторная работа. uint1024_t

Задача

Реализовать пользовательский тип для целого беззнакового числа фиксированной длины ***uint1024_t***

Для вышеуказанного типа реализовать функции с следующими сигнатурами:

1. *uint1024_t from_uint(unsigned int x)* - генерация из числа
2. *uint1024_t add_op(uint1024_t x, uint1024_t y)* - сложение
3. *uint1024_t subtr_op(uint1024_t x, uint1024_t y)* - вычетание
4. *uint1024_t mult_op(uint1024_t x, uint1024_t y)* - умножение
5. *void printf_value(uint1024_t x)* - вывод в стандартный поток вывода
6. *void scanf_value(uint1024_t* x)* - чтение из стандартного потока ввода

Примечание:

1. Переполнение - Undefined Behavior
2. При реализации думать об оптимальном использовании памяти
3. Реализовать программу демонстрирующую работоспособность вышеуказанных функций

Лабораторная работа. Анализ логов сервера

Задача

Вам предоставлен access.log одного из серверов NASA (скачать). Это текстовый файл, каждая строка которого имеет следующий формат:

$remote_addr - - [$local_time] “$request” $status $bytes_send

$remote_addr - источник запроса
$local_time - время запроса
$request - запрос
$status - статус ответ
$bytes_send - количество переданных в ответе байт

Например:

198.112.92.15 - - [03/Jul/1995:10:50:02 -0400] "GET /shuttle/countdown/ HTTP/1.0" 200 3985

Требуется

1. Подготовить список запросов, которые закончились 5xx ошибкой, с количеством неудачных запросов
2. Найти временное окно (длительностью параметризуются), когда количество запросов на сервер было максимально

Примечание:

1. Для парсинга строк проще всего воспользоваться библиотеками *stdio.h* и *string.h* стандартной библиотеки
2. Про коды ответа можно почитать например вот тут

Лабораторная работа. Редактор метаинформации mp3-файла

Задача.

Реализовать редактор текстовой метаинформации mp3 файла. В качестве стандарта метаинформации принимаем ID3v2.

Редактор представлять из себя консольную программу принимающую в качестве аргументов имя файла через параметра *--filepath* , а также одну из выбранных команд

1. --show - отображение всей метаинформации в виде таблицы
2. *--set=prop_name --value=prop_value* - выставляет значение определенного поля метаинформации с именем prop_name в значение prop_value
3. --get=prop_name - вывести определенное поле метаинформации с именем prop_name

Например:

app.exe --filepath=Song.mp3 --show
app.exe --filepath=Song.mp3 --get=TIT2
app.exe --filepath=Song.mp3 --set=COMM --value=Test

Примечание.

При выполнения данной работы разрешается использовать только стандартную библиотеку языка C. Исключением может являться процесс разбора аргументов командной строки.

Лабораторная работ. Игра жизнь

Целью лабораторной работы является реализация игры “Жизнь” , позволяющая выводить поколение игры в монохромную картинку в формате BMP. Плоскость “вселенной” игры ограничена положительными координатами.

Лабораторная работы должна быть выполнена в виде консольного приложения принимающего в качестве аргументов следующие параметры:

1. ***--input input_file.bmp***

   Где input_file.bmp - монохромная картинка в формате bmp, хранящая начальную ситуация (первое поколение) игры
2. -***-output dir_name***

   Название директории для хранения поколений игры в виде монохромной картинки
3. ***--max_iter N***

   Максимальное число поколений которое может эмулировать программа. Необязательный параметр, по-умолчанию бесконечность
4. ***--dump_freq N***

   Частота с которой программа должно сохранять поколения виде картинки. Необязательный параметр, по-умолчанию равен 1

Программа должна предусматривать исключительные ситуации, которые могут возникать во время ее работы и корректно их обрабатывать.

Лабораторная работ. Архиватор файлов.

Целью лабораторной работы является разработка программы по архивированию и распаковке нескольких файлов в один архив. Архиватор должен

1. Уметь архивировать несколько (один и более) указанных файлов в архив с расширением ***.arc**
2. Уметь распаковывать файловых архив, извлекая изначально запаковонные файлы
3. Предоставлять список файлов упакованных в архиве
4. *Сжимать и разжимать данные при архивировании с помощью алгоритма Хаффмана (опциональное задание, оценивается доп баллами)*

Архиватор должен быть выполнен в виде консольного приложения, принимающего в качестве аргументов следующий параметры

- **--file FILE**
  Имя файлового архива с которым будет работать архиватор
- **--create**
  Команда для создания файлового архива
- **--extract;**
  Команда для извлечения из файлового архива файлов
- **--list**
  Команда для предоставления списка файлов, хранящихся в архиве
- **FILE1 FILE2 …. FILEN**
  Свободные аргументы для передачи списка файлов для запаковки

Примеры использования:

arc --file data.arc --create a.txt b.bin c.bmp
arc --file data.arc --extract
arc --file data.arc --list