Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Факультет информационных технологий и управления

Кафедра информационных технологий автоматизированных систем

пояснительная записка

к курсовому проекту

по курсу «Системное программное обеспечение»

на тему:

**«Сжатие данных»**

|  |  |
| --- | --- |
| Выполнил студент гр. 820601Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Крюков С.Ю. (подпись)\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ревотюк М.П. (подпись) |

Минск 2011

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_БГУИР\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 (назва ВНУ)

**Факультэт ІТіК**

 **“ЗАЦВЯРДЖАЮ”**

**Загадчык кафедры ІТАС\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(подпіс)

“\_\_\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2011

**З А Д А Н Н Е**

па курсавому праектаванню

Студэнту\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 (Фамілія, ініцыялы, група)

1. Тэма праекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Тэрмін здачы студэнтам закончанага праекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Зыходныя данныя да праекта

 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Змест разлікова-тлумачальнай запіскі (пералік пытанняў, якія падлягаюўь распрацоўцы)\_\_\_

\_\_Введение\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_Постановочная часть\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_ Системные требования \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_ Разработка программы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_Руководство пользователя\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_Заключение\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_Список использованных источников \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_Приложение А \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_Приложение Б \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Пералік графічнага матэрыяла (з дакладным пазначэннем абавязковых чарцяжоў і графікаў)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Кансультант па праекту (з пазначэннем раздзелаў праекта) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7. Дата выдачы задання \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

8. Каляндарны графік работы над праектам на ўвесь перыяд праектавання (з пазначэннем

тэрмінаў выканання і працаемкасці асобных этапаў) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 КІРАЎНІК\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 (подпіс)

Заданне прыняў для выканання \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 (дата і подпіс студэнта)

СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 4](#_Toc310186377)

[ВВЕДЕНИЕ 5](#_Toc310186378)

[1 Постановочная часть 9](#_Toc310186379)

[2 Системные требования 10](#_Toc310186380)

[3 Разработка программы 11](#_Toc310186381)

[4 Руководство пользователя 12](#_Toc310186382)

[4.1. Установка и запуск программы 12](#_Toc310186383)

[4.2. Использование программы в графическом режиме 15](#_Toc310186384)

[4.3. Использование программы в консольном режиме 18](#_Toc310186385)

[ЗАКЛЮЧЕНИЕ 20](#_Toc310186386)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 21](#_Toc310186387)

[Приложение А 22](#_Toc310186388)

[Приложение Б 26](#_Toc310186389)

ВВЕДЕНИЕ

Сжатие данных — алгоритмическое преобразование данных, производимое с целью уменьшения их объёма. Применяется для более рационального использования устройств хранения и передачи данных. Синонимы — упаковка данных, компрессия, сжимающее кодирование, кодирование источника. Обратная процедура называется восстановлением данных (распаковкой, декомпрессией).

Сжатие основано на устранении избыточности, содержащейся в исходных данных. Простейшим примером избыточности является повторение в тексте фрагментов (например, слов естественного или машинного языка). Подобная избыточность обычно устраняется заменой повторяющейся последовательности ссылкой на уже закодированный фрагмент с указанием его длины. Другой вид избыточности связан с тем, что некоторые значения в сжимаемых данных встречаются чаще других. Сокращение объёма данных достигается за счёт замены часто встречающихся данных короткими кодовыми словами, а редких — длинными (энтропийное кодирование). Сжатие данных, не обладающих свойством избыточности (например, случайный сигнал или шум, зашифрованные сообщения), принципиально невозможно без потерь.

В основе любого способа сжатия лежит модель источника данных, или, точнее, модель избыточности. Иными словами, для сжатия данных используются некоторые априорные сведения о том, какого рода данные сжимаются. Не обладая такими сведениями об источнике, невозможно сделать никаких предположений о преобразовании, которое позволило бы уменьшить объём сообщения. Модель избыточности может быть статической, неизменной для всего сжимаемого сообщения, либо строиться или параметризоваться на этапе сжатия (и восстановления). Методы, позволяющие на основе входных данных изменять модель избыточности информации, называются адаптивными. Неадаптивными являются обычно узкоспециализированные алгоритмы, применяемые для работы с данными, обладающими хорошо определёнными и неизменными характеристиками. Подавляющая часть достаточно универсальных алгоритмов являются в той или иной мере адаптивными.

Все методы сжатия данных делятся на два основных класса:

* сжатие без потерь
* сжатие с потерями

При использовании сжатия без потерь возможно полное восстановление исходных данных, сжатие с потерями позволяет восстановить данные с искажениями, обычно несущественными с точки зрения дальнейшего использования восстановленных данных. Сжатие без потерь обычно используется для передачи и хранения текстовых данных, компьютерных программ, реже — для сокращения объёма аудио- и видеоданных, цифровых фотографий и т. п., в случаях, когда искажения недопустимы или нежелательны. Сжатие с потерями, обладающее значительно большей, чем сжатие без потерь, эффективностью, обычно применяется для сокращения объёма аудио- и видеоданных и цифровых фотографий в тех случаях, когда такое сокращение является приоритетным, а полное соответствие исходных и восстановленных данных не требуется. В общем случае алгоритмы сжатия без потерь универсальны в том смысле, что их применение безусловно возможно для данных любого типа, в то время как возможность применения сжатия с потерями должна быть обоснована. Для некоторых типов данных искажения не допустимы в принципе. В их числе:

* символические данные: программы и их исходные тексты, двоичные данные и т. п.
* жизненно важные данные, изменения в которых могут привести к критическим ошибкам: например, получаемые с медицинской измерительной аппаратуры или контрольных приборов летательных, космических аппаратов и т. п.
* многократно подвергаемые сжатию и восстановлению промежуточные данные при многоэтапной обработке графических, звуковых и видеоданных

Различные алгоритмы могут требовать различного количества ресурсов вычислительной системы, на которых они реализованы:

* оперативной памяти (под промежуточные данные)
* постоянной памяти (под код программы и константы)
* процессорного времени

В целом, эти требования зависят от сложности и «интеллектуальности» алгоритма. Общая тенденция такова: чем эффективнее и универсальнее алгоритм, тем большие требования к вычислительным ресурсам он предъявляет. Тем не менее, в специфических случаях простые и компактные алгоритмы могут работать не хуже сложных и универсальных. Системные требования определяют их потребительские качества: чем менее требователен алгоритм, тем на более простой, а следовательно, компактной, надёжной и дешёвой системе он может быть реализован.

Так как алгоритмы сжатия и восстановления работают в паре, имеет значение соотношение системных требований к ним. Нередко можно усложнив один алгоритм значительно упростить другой. Таким образом, возможны три варианта:

* Алгоритм сжатия требует больших вычислительных ресурсов, нежели алгоритм восстановления. Это наиболее распространённое соотношение, характерное для случаев, когда однократно сжатые данные будут использоваться многократно. В качестве примера можно привести цифровые аудио- и видеопроигрыватели.
* Алгоритмы сжатия и восстановления требуют приблизительно равных вычислительных ресурсов. Наиболее приемлемый вариант для линий связи, когда сжатие и восстановление происходит однократно на двух её концах (например, в цифровой телефонии).
* Алгоритм сжатия существенно менее требователен, чем алгоритм восстановления. Такая ситуация характерна для случаев, когда процедура сжатия реализуется простым, часто портативным устройством, для которого объём доступных ресурсов весьма критичен, например, космический аппарат или большая распределённая сеть датчиков. Это могут быть также данные, распаковка которых требуется в очень малом проценте случаев, например запись камер видеонаблюдения.

Имеется два основных подхода к сжатию данных неизвестного формата:

* На каждом шаге алгоритма сжатия очередной сжимаемый символ либо помещается в выходной буфер сжимающего кодера как есть (со специальным флагом, помечающим, что он не был сжат), либо группа из нескольких сжимаемых символов заменяется ссылкой на совпадающую с ней группу из уже закодированных символов. Поскольку восстановление сжатых таким образом данных выполняется очень быстро, такой подход часто используется для создания самораспаковывающихся программ
* Для каждой сжимаемой последовательности символов однократно либо в каждый момент времени собирается статистика её встречаемости в кодируемых данных. На основе этой статистики вычисляется вероятность значения очередного кодируемого символа (либо последовательности символов). После этого применяется та или иная разновидность энтропийного кодирования, например, арифметическое кодирование или кодирование Хаффмана, для представления часто встречающихся последовательностей короткими кодовыми словами, а редко встречающихся — более длинными
1. Постановочная часть

Требуется разработать программу, которая будет уметь:

* Сжимать файлы без потерь
* Распаковывать сжатые файлы

Приложение должно легко устанавливаться на компьютер, что позволит пользователю сделать это самому, не прибегая к помощи системного администратора. Также желательно наличие интуитивного графического интерфейса для обычных пользователей и возможность работы с программой из командной строки для продвинутых пользователей.

1. Системные требования

Для использования разработанного приложения на компьютере и его функционирования необходимо выполнение следующих условий:

* персональный компьютер, оборудованный устройством отображения информации, клавиатурой и манипулятором ввода типа «мышь» (необходим для работы в графическом режиме)
* наличие на целевой платформе виртуально машины Java (версии не ниже 1.5)
* наличие на целевой платформе системы управления проектами Apache Maven (необходимо только при сборке приложения из исходных кодов)
* наличие на целевой платформе системы контроля версий Mercurial SCM (необходимо только для получения последней версии исходного кода из репозитория)

Из-за использования кроссплатформенного языка программирования Java приложения может быть запущено практически на любой современной ОС – Linux, MacOS, Solaris и т.д.

1. Разработка программы

Разработанное в ходе курсового проекта приложение предназначено для сжатия данных без потерь. Функциональность программы отражена на следующих диаграммах UML:



Рисунок 3.1 — Диаграмма вариантов использования



Рисунок 3.2 — Диаграмма последовательности для варианта сжатия файла



Рисунок 3.2 — Диаграмма последовательности для варианта распаковки файла

1. Руководство пользователя
	1. Установка и запуск программы

Разработанное приложение не требует инсталляции на целевую платформу, не оставляет никаких следов своего использования (за исключением сжатых или разжатых по запросу пользователя файлов) в системных логах, реестре (для платформы Microsoft Windows ™) либо иных, не указанных в данном руководстве директориях. Приложение распространяется под MIT License вместе с исходными кодами, последняя версия всегда может быть скачана с веб-сайта проекта, приведённого в приложении Б. Бинарная дистрибуция приложения представляет собой ZIP-архив (существуют декомпрессоры для большинства современных платформ) с JAR-файлом приложения и набором скриптов под целевую платформу для его запуска в различных конфигурациях.



Рисунок 4.1 — Архив приложения для платформы Microsoft Windows ™

При использовании бинарной дистрибуции приложения пользователь должен лишь распаковать архив в любое, удобное для него место, после чего приложение готово к использованию (при соблюдении системных требований, оговорённых выше).

Также возможно получение исходных кодов программы и самостоятельная сборка. Для этого необходимо получить последнюю версию исходного кода из репозитория, указанного в приложении Б.



Рисунок 4.2 — Получение последней версии исходного кода из репозитория

После успешного клонирования репозитория в исходной директории будет создан новый каталог «lzwj», содержимое которого показано на рисунке 4.3.



Рисунок 4.3 — Содержимое папки с исходным кодом проекта

Директория «.hg» и файл «.hgignore» содержат вспомогательную информацию для Mercurial SCM. В каталоге «src» содержится исходный код проекта, используемые им ресурсы, скрипты для запуска, описатели сборок (assemblies).

Файл «pom.xml» (Project Object Model) содержит в себе описатель проекта, его внешние зависимости, конфигурации плагинов для Maven и профили сборки приложения.

Для наиболее распространённых целей сборки проекта существуют предустановленные скрипты – «distro.bat», «jar.bat», «jar-with-dependencies.bat». В результате их выполнения будет создана поддиректория «target» содержащая ZIP-архив проекта (бинарную дистрибуцию), JAR-файл без внешних зависимостей (может оказаться неработоспособным при раздельном использовании) или JAR-файл со всеми необходимыми зависимостями (может использоваться самостоятельно) соответственно. Также в директорию «target» будут помещены скрипты для запуска, хранящиеся в каталоге «src/main/scripts/runners». За более подробной информацией об использовании Mercurial SCM и Maven пользователю следует обратиться к официальной документации, ссылки на которую приведены в приложении Б.



Рисунок 4.4 — Процесс сборки бинарной дистрибуции приложения

После распаковки приложения, полученного любым из описанных выше путей, для его запуска в графическом режиме необходимо выполнить скрипт gui.bat (для платформы Microsoft Windows ™). После выполнения скрипта на экране отобразится интуитивно-понятный, дружелюбный пользователю графический интерфейс (показан на рисунке 4.5).



Рисунок 4.5 — Главное окно приложения

При возникновении каких-либо проблем пользователь может обратиться за помощью к системному администратору, или связаться с автором приложения, по электронной почте, указанной на сайте проекта.

* 1. Использование программы в графическом режиме

Процесс использования приложения в графическом режиме не должен вызывать проблем. Он прост и доступен большинству пользователей.

Для сжатия данных в графическом режиме пользователю следует нажать кнопку «Compress» на главном окне программы. Появится диалоговое окно выбора файла для сжатия.



Рисунок 4.6 — Диалоговое окно выбора файла для сжатия

После выбора файла процесс сжатия начинается автоматически в фоновом режиме. Сжатые данные помещаются в файл с именем исходного файла и расширением «.lzwj». После окончания сжатия на экране отобразится модальный диалог, сигнализирующий о статусе операции.



Рисунок 4.7 — Модальный диалог успешного окончания сжатия

При возникновении каких-либо ошибок в процессе сжатия, диалоговое окно отразит причину неудачи.



Рисунок 4.8 — Модальный диалог, сигнализирующий об ошибочном указании исходного файла

Для декомпрессии файлов пользователю следует выбрать кнопку «Decompress» на главном окне приложения. После этого отобразится диалоговое окно выбора файла для распаковки.



Рисунок 4.9 — Диалоговое окно выбора файла для распаковки

После выбора файла для декомпрессии на экране отобразится второе окно для выбора выходного файла.



Рисунок 4.10 — Диалоговое окно выбора выходного файла

Процесс распаковки файлов также происходит в фоновом режиме и не затрагивает графический интерфейс. После окончания распаковки на экране отобразится окно с результатом.



Рисунок 4.11 — Модальный диалог успешного окончания распаковки

При возникновении исключительных ситуаций во время распаковки отобразится модальный диалог с ошибкой.



Рисунок 4.12 — Модальный диалог, сигнализирующий об ошибке в процессе распаковки

* 1. Использование программы в консольном режиме

Процесс использования приложения в консольном режиме предназначен для продвинутых пользователей. Он может быть использован для пакетной обработки файлов или при отсутствии манипулятора типа «мышь» (частая ситуация для серверной среды).

Для запуска приложения в режиме консольного использования необходимо первым параметром запуска передать «c» или «d» (запуск в режиме графического использования производится с единственным параметром «g»). Параметр «c» указывает на необходимость упаковки (compress), параметр «d» - на необходимость распаковки (decompress).



Рисунок 4.13 — Использование программного продукта из консоли для компрессии



Рисунок 4.14 — Использование программного продукта из консоли для декомпрессии

ЗАКЛЮЧЕНИЕ

Разрабатываемая в контексте данной работы утилита в перспективе может использоваться для реального применения в частном порядке. Распространение под дружелюбной Open Source лицензией также не препятствует изменению кода для адаптации к новым условиям использования.

В данной работе разработана утилита для сжатия / декомпрессии данных на основе алгоритма LZW, поддерживающая работу из консоли и в графическом режиме.

В процессе выполнения курсовой работы были закреплены знания, полученные при изучении дисциплины «Системное программное обеспечение». Были изучены такие пункты:

* анализ предметной области (проблема сжатия данных);
* построение алгоритма;
* реализация алгоритма на выбранном языке;
* написание интерфейса пользователя;

Была освоена и закреплена работа с такими прикладными программами:

* SpringSource Tool Suite (специализированная Eclipse IDE для работы со Spring);
* Apache Maven;
* Mercurial SCM;
* Gimp;

Предполагается дальнейшее развитие и улучшение созданной утилиты.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Java 2. Тонкости программирования / Кей Хорстман, Гари Корнелл; Вильямс. – Москва, 2010. -857 с.
2. Segerstad D. Be calm while making your Java code / Daniel Segerstad; под ред. Johannes Hedberg. –Oslo, 2003. -1488 с.
3. Кенобьев О.В. Изучаем Mercurial SCM за 24 часа / О.В. Кенобьев; под ред. А. Небоходова. –Спб, 2007. -501 с.

Приложение А

**(справочное)**

**Примеры листингов программы**

package by.dev.madhead.lzwj.compress;

import java.io.BufferedInputStream;

import java.io.BufferedOutputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import by.dev.madhead.lzwj.io.Input;

import by.dev.madhead.lzwj.io.Output;

import by.dev.madhead.lzwj.util.ByteArray;

/\*\*

 \* Class for compressing and decompressing. It is not thread-safe.

 \*

 \* @author madhead

 \*

 \*/

public class LZW {

 /\*\*

 \* Initial size of compress table. Before starting LZW compression each byte

 \* has its mapping set to itself.

 \*/

 public static final int INITIAL\_DICT\_SIZE = 256;

 /\*\*

 \* Size in bits of compressed code.

 \*/

 public static final int DEFAULT\_CODEWORD\_LENGTH = 12;

 private int codeWordLength = DEFAULT\_CODEWORD\_LENGTH;

 private Map<ByteArray, Integer> codeTable;

 private List<ByteArray> decodeTable;

 /\*\*

 \* Returns currently used codeword length.

 \*

 \* @return codeword length.

 \*/

 public int getCodeWordLength() {

 return codeWordLength;

 }

 /\*\*

 \* Sets codeword length in bits to be used in compress/decompress

 \* operations.

 \*

 \* @param codeWordLength

 \* codeword length in bits to be used in compress/decompress

 \* operations.

 \*/

 public void setCodeWordLength(int codeWordLength) {

 // Haha! Expected this method do something useful, didn't you?

 // this.codeWordLength = codeWordLength;

 }

 /\*\*

 \* Compresses <code>in</code> to <code>out</code>. Flushes output after

 \* completion. You must explicitly close this streams after compression.

 \*

 \* @param in

 \* input stream to be compressed.

 \* @param out

 \* output stream to place compression result in.

 \* @throws IOException

 \*/

 public void compress(InputStream in, OutputStream out) throws IOException {

 // Here be dragons!

 init();

 int code = INITIAL\_DICT\_SIZE;

 int maxCode = (1 << codeWordLength) - 1;

 InputStream bufferedIn = new BufferedInputStream(in);

 Output compressedOutput = new Output(new BufferedOutputStream(out),

 codeWordLength);

 int firstByte = bufferedIn.read();

 ByteArray w = new ByteArray((byte) firstByte);

 int K;

 while ((K = bufferedIn.read()) != -1) {

 ByteArray wK = new ByteArray(w).append((byte) K);

 if (codeTable.containsKey(wK)) {

 w = wK;

 } else {

 compressedOutput.write(codeTable.get(w));

 if (code < maxCode) {

 codeTable.put(wK, code++);

 }

 w = new ByteArray((byte) K);

 }

 }

 compressedOutput.write(codeTable.get(w));

 compressedOutput.flush();

 }

 /\*\*

 \* Decompresses <code>in</code> to <code>out</code>. Flushes output after

 \* completion. You must explicitly close this streams after compression.

 \*

 \* @param in

 \* input stream to be decompressed.

 \* @param out

 \* output stream to place decompression result in.

 \* @throws IOException

 \*/

 public void decompress(InputStream in, OutputStream out) throws IOException {

 init();

 Input compressedInput = new Input(new BufferedInputStream(in),

 codeWordLength);

 OutputStream bufferedOut = new BufferedOutputStream(out);

 int oldCode = compressedInput.read();

 bufferedOut.write(oldCode);

 int character = oldCode;

 int newCode;

 while ((newCode = compressedInput.read()) != -1) {

 ByteArray string;

 if (newCode >= decodeTable.size()) {

 string = new ByteArray(decodeTable.get(oldCode));

 string.append((byte) character);

 } else {

 string = decodeTable.get(newCode);

 }

 for (int i = 0; i < string.size(); i++) {

 bufferedOut.write(string.get(i));

 }

 character = string.get(0);

 decodeTable.add(new ByteArray(decodeTable.get(oldCode))

 .append((byte) character));

 oldCode = newCode;

 }

 bufferedOut.flush();

 }

 /\*\*

 \* Initializes class for compression and decompression.

 \*/

 private void init() {

 codeTable = new HashMap<ByteArray, Integer>();

 decodeTable = new ArrayList<ByteArray>();

 for (int i = 0; i < INITIAL\_DICT\_SIZE; i++) {

 codeTable.put(new ByteArray((byte) i), i);

 decodeTable.add(new ByteArray((byte) i));

 }

 }

}

Приложение Б

**(информационное)**

**Репозитории кода**

1. <http://code.google.com/p/lzwj/> - страница проекта
2. <http://code.google.com/p/lzwj/source/browse/> - овновной репозиторий кода
3. <http://code.google.com/p/lzwj/downloads/list> - хранилище дистрибутивов