**Структура внешней спецификации программы**

1. Наименование программы
2. Назначение программы
3. Требования к режиму упаковки
   1. Состав и структура входных данных
   2. Состав и структура выходных данных
   3. Ограничения на выполнение
4. Требования к режиму распаковки
   1. Состав и структура входных данных
   2. Состав и структура выходных данных
   3. Ограничения на выполнение
5. Ограничения на выполнение
6. Приложение А. Алгоритм упаковки и распаковки данных
7. Приложение Б. Пример входных и выходных данных
8. Приложение В. Описание ситуаций с корректным и некорректным оформлением модели алфавита
9. Приложение Г. Список ошибок
10. Приложение Д. Таблицы ситуаций для функций верхнего уровня
11. **Наимен**о**вание программы**

NumeralPacking – программа, предназначенная для упаковки и распаковки текста, используя представление этого текста в виде вещественного числа, записанного в виде двух целых чисел.

1. **Назначение программы**

Программа предназначена для упаковки/распаковки текста и может использоваться для архивации файлов.

1. **Требования к режиму упаковки** 
   1. **Состав и структура входных данных**

Для режима упаковки входными данными является исходный текст.

Входные данные передаются через аргументы командной строки. Сначала записывается название программы, а затем через пробел указывается, какое действие должна выполнить программа (при упаковке – -p) и далее указывается путь к файлу для упаковки, а также путь к новому упакованному файлу.

Пример командной строки для упаковки:

NumeralPacking.exe -p C:\Users\User\Desktop\fileForPack.txt C:\Users\User\Desktop\resultFile.txt

* 1. **Состав и структура выходных данных**

Результатом работы программы являются текстовой файл. В нем содержатся два целых числа, которое представляет собой текст и длину получившегося числа, а также модель алфавита вида:

<число, равное левой границе интервала символа><символ><длина числа левой границы>

Строка, описывающая вероятность EOF-символа, выглядит следующим образом:

<число левой границы><e><длина числа левой границы><o><число, равное вероятности EOF-символа><f><длина числа вероятности>

* 1. **Ограничения на выполнение**

Если файл с логом не может быть создан или не будет возможна запись в него, то будет выведено сообщение об ошибке и прекратится выполнение программы.

Если файл с входным текстом не существует или не может быть прочитан, то будет выведено сообщение об ошибке в лог и прекратится выполнение программы.

Длина входного текста ограничена: приблизительно не более 100 Мб.

Если выходной файл не может быть записан, то будет выведено сообщение об ошибке в лог и прекратится выполнение программы.

Подробная информация о сообщениях об ошибках, записях лога и ситуациях, к ним приводящим, содержится в приложении Б.Г.

1. **Требования к режиму распаковки** 
   1. **Состав и структура входных данных**

Для режима распаковки на вход программы подается файл, в котором содержится упакованный текст и модель алфавита.

Входные данные передаются через аргументы командной строки. Сначала записывается название программы, а затем через пробел указывается, какое действие должна выполнить программа (при распаковке – -u) и далее указывается путь к упакованному файлу и путь к новому распакованному файлу.

Пример командной строки для распаковки:

NumeralPacking.exe –u C:\Users\User\Desktop\packedFile.txt C:\Users\User\Desktop\unpackedFile.txt

* 1. **Состав и структура выходных данных**

Результатом работы программы является текстовый файл с исходными символами.

* 1. **Ограничения на выполнение**

Если файл с логом не может быть создан или не будет возможна запись в него, то будет выведено сообщение об ошибке и прекратится выполнение программы.

Если файл с входным текстом и с моделью алфавита не существует или не может быть прочитан, то будет выведено сообщение об ошибке в лог и прекратится выполнение программы.

Модель алфавита должна быть записана следующим образом:

<число, равное левой границе интервала символа><символ><длина числа левой границы>

Строка, описывающая вероятность EOF-символа, выглядит следующим образом:

<число левой границы><e><длина числа левой границы><o><число, равное вероятности EOF-символа><f><длина числа вероятности>

Если модель задана некорректно, то будет выведено сообщение об ошибке в лог и выполнение программы будет остановлено. Подробное описание ситуаций с корректным и некорректным оформлением моделей приведено в приложении Б.В.

Если выходной файл не может быть записан, то будет выведено сообщение об ошибке в лог и прекратится выполнение программы.

Подробная информация о сообщениях об ошибках, записях лога и ситуациях, к ним приводящим, содержится в приложении Б.Г.

1. **Ограничения на выполнение**

Если аргументы командной строки заданы некорректно, т.е.:

1. режим упаковки/распаковки не указан или указан неверно;
2. если количество аргументов менее или более 4;

то будет выведено сообщение об ошибке в лог и прекратится выполнение программы.

Размер входного файла с текстом для упаковки не должен превышать 100 Мб памяти на диске.

Размеры входного файла с упакованным текстом и выходного файла определяются операционной системой.

Подробная информация о сообщениях об ошибках, записях лога и ситуациях, к ним приводящим, содержится в приложении Б.Г.

**Приложение Б.А. Алгоритм упаковки и распаковки данных**

1 Открыть лог-файл для записи и сделать запись о начале программы

2 Если открыть файл не удалось

2.1 Вывести сообщение об ошибке и завершить работу программы

3 Проверить корректность переданных аргументов командной строки и разобрать их (inspectAndParseArgumentsOfCmd)

4 Если аргументы указаны неверно

4.1 Вывести сообщение об ошибке в лог и завершить работу программы

4.2 Закрыть лог-файл

5 Открыть входной файл для чтения

6 Если файл не может быть прочитан

6.1 Вывести сообщение об ошибке в лог и завершить работу программы

6.2 Закрыть лог-файл

7 Открыть выходной файл для записи

8 Если невозможна запись в файл

8.1 Вывести сообщение об ошибке в лог и завершить работу программы

8.2 Закрыть лог-файл

8.3 Закрыть входной файл

9 Определить режим работы программы

10 Если выбран режим упаковки (функция packFile)

10.1 Составить модель алфавита (readDataForAlphabet)

10.1.1 Считать входной текст посимвольно

10.1.1.1 Считывая посимвольно

10.1.1.2 Если символ есть в алфавите – инкрементировать счетчик данного символа (buildAlphabetModel)

10.1.1.3 Иначе – добавить символ в алфавит и присвоить счетчику 1

10.1.1.4 Инкрементировать счетчик числа символов в тексте

10.1.2 Добавить информацию о EOF-символе (счетчик – 1)

10.1.3 Рассчитать вероятности и левые границы всех символов алфавита на основе данных счетчика и длины входного текста (convertAlphabetModel)

10.2 Считать входной текст порциями по 10000 символов (readDataForPack)

10.3 Упаковать текст (packText)

10.3.1 Для каждого символа текста:

10.3.1.1Сузить интервал для текущего символа (constrictInterval)

10.3.1.1.1Сохранить полученный интервал

10.3.2 Рассчитать среднее арифметического границ полученного интервала (findArithmeticalMean)

10.4 Записать полученный интервал и модель алфавита в выходной файл (writeDataForPack)

11 Если выбран режим распаковки (unpackFile)

11.1 Считать строку с интервалом из входного файла и записать его в число типа number (readDataForUnpack)

11.2 Считать модель алфавита и записать ее в массив строк

11.3 Разобрать считанную модель (buildModelFromText)

11.4 Если модель некорректна

11.4.1 Вывести сообщение об ошибке в лог и прекратить работу программы

11.5 Произвести распаковку текста (unpackText)

11.5.1 Пока не встретился символ конца файла

11.5.1.1 Найти для данного числа соответствующий интервал с помощью модели алфавита (findSymbolAccordingToInterval)

11.5.1.1.1 Получить букву из алфавита исходя из полученного интервала

11.5.1.2 Сохранить в строку с выходным текстом полученную букву

11.6 Записать полученный текст в выходной файл (writeDataForUnpack)

12 Записать в лог сообщение об окончании работы программы

Примечание: числа в памяти компьютера представлены с помощью библиотеки GMP, в которой реализована «длинная» арифметика в виде классов целых чисел.

Подробная информация о сообщениях об ошибках, записях лога и ситуациях, к ним приводящим, содержится в приложении Б.Г.

Подробное описание функций приведено в приложении В.

**Приложение Б.Б. Пример входных и выходных данных**

Режим упаковки:

Входные данные:

Исходный текст:

trollollo

Выходные данные:

Среднее значение полученного диапазона:

143079744

Длина числа: 10

Модель алфавита:

|  |  |  |
| --- | --- | --- |
| Символ | Вероятность | Интервал |
| t | 0.1 | [0.0 ; 0.1] |
| r | 0.1 | [0.1 ; 0.2] |
| o | 0.3 | [0.2 ; 0.5] |
| l | 0.4 | [0.5 ; 0.9] |
| EOF | 0.1 | [0.9 ; 1.0] |

Режим распаковки:

Входные данные:

Среднее значение полученного диапазона:

9406833165

Длина числа: 11

Модель алфавита:

|  |  |  |
| --- | --- | --- |
| Буква | Вероятность | Интервал |
| а | 0.3 | [0.0 ; 0.3] |
| д | 0.1 | [0.3 ; 0.4] |
| п | 0.1 | [0.4 ; 0.5] |
| т | 0.1 | [0.5 ; 0.6] |
| ц | 0.1 | [0.6 ; 0.7] |
| и | 0.1 | [0.7 ; 0.8] |
| я | 0.1 | [0.8 ; 0.9] |
| EOF | 0.1 | [0.9 ; 1.0] |

Выходные данные:

Исходный текст:

адаптация

**Приложение Б.В. Описание ситуаций с корректным и некорректным оформлением модели алфавита**

Таблица 1. Ситуации с оформлением модели

|  |  |  |  |
| --- | --- | --- | --- |
| Тип ситуации | Описание ситуации | Запись | Правильная запись |
| Кор-ректная запись | - | 456g3 | - |
| Некор-ректная запись | Отсутствует символ | 4563 | 456g3 |
| Более 1 символа | 456gw3 | 456g3 |
| Неверный порядок символа и чисел | g4563 | 456g3 |
| В одной строке более одного описания | 456g3 4t1 | 456g3  4t1 |

**Приложение Б.Г. Описание ситуаций с корректным и некорректным оформлением модели алфавита**

Таблица 2. Коды возможных ошибок с расшифровкой

| Код ошибки | Описание ошибки | Пример, приводящий к ошибке | Текст ошибки, записываемый в лог |
| --- | --- | --- | --- |
| 001 | Программе передано недостаточное количество аргументов. | NumeralPacking.exe –u C:\Users\User\Desktop\packedFile.txt | Error: There aren’t enough files. |
| 002 | Программе передано слишком много аргументов. | NumeralPacking.exe -u C:\Users\User\Desktop\packedFile.txt C:\Users\User\Desktop\unpackedFile.txt  C:\Users\User\Desktop\unpackedFile.txt | Error: There are too much files. |
| 003 | Выбран неверный режим. | NumeralPacking.exe -w C:\Users\User\Desktop\packedFile.txt C:\Users\User\Desktop\unpackedFile.txt | Error: The mode is incorrect. |
| 004 | Не удалось прочитать входной файл. | NumeralPacking.exe -u C:\Users\User\Desktop\packedFile.txt C:\Users\User\Desktop\unpackedFile.txt | Error: Input file can’t be read. |
| 005 | Не удалось записать выходной файл. | NumeralPacking.exe -u C:\Users\User\Desktop\packedFile.txt C:\Users\User\Desktop\unpackedFile.txt | Error: Output file can’t be written. |
| 006 | Упакованный текст записан некорректно. | 123433$78778 | Error: Interval is incorrect. |
| 007 | Модель алфавита некорректна. | 456gw3 | Error: Model is incorrect. |

Таблица 3. Все возможные сообщения лога

|  |  |  |
| --- | --- | --- |
| **№** | **Текст сообщения** | **Описание ситуации, при которой происходит запись данного сообщения** |
| 1 | Program started its execution. | Программа начала свое выполнение. |
| 2 | Pack mode is selected. | Выбран режим упаковки. |
| 3 | Unpack mode is selected. | Выбран режим распаковки. |
| 4 | Inspection of command line arguments was successful. Command line arguments are correct and were parsed successfully. | Проверка командной строки была успешной. Аргументы командной строки верны и были успешно разобраны. |
| 5 | The first byte was read for the alphabet building. The alphabet building was begun. | Первый байт был прочитан для построения алфавита. Построение алфавита началось. |
| 6 | The alphabet building was completed. | Алфавит построен. |
| 7 | Data reading was started. | Началось считывание данных. |
| 8 | The text was read. Its amount of symbols is: <количество прочитанных символов> | Текст считан. Количество прочитанных символов: |
| 9 | Packing was started: file <название файла> | Началась упаковка файла |
| 10 | Packing was finished: file <название файла> | Завершена упаковка файла |
| 11 | The packed text and alphabet were written in file <название файла> | Запакованный текст и алфавит были записаны в файл |
| 12 | The alphabet was read successfully. | Алфавит успешно считан из файла. |
| 13 | The alphabet was parsed successfully. | Алфавит успешно разобран (при считывании в готовом виде из файла). |
| 14 | Unpacking was started: file <название файла> | Началась распаковка файла |
| 15 | Unpacking was finished: file <название файла> | Завершена распаковка файла |
| 16 | The unpacked text was written in file <название файла> | Распакованный текст был записан в файл |
| 17 | Program finished its execution. | Программа завершила свое выполнение. |