asgn6 design

Ryan Hui

rhui1

DESIGN

Assignment 6 implements the LZ78 compression and decrompression algorithm which uses the files encode.c to compress files and decode.c to decompress files with availibity to operate on both big and little endian systems. The trie.c and word.c files are used to compress and decompress the information. io.c is used to read and write the data.

Files

-encode.c
-decode.c
-trie.c
-trie.h
-word.c
-word.h
-io.c
-io.h
-endian.h
-code.h
-README.md -DESIGN.pdf
-WRITEUP.pdf

Pseudocode

trie.c

```
trie node create function
construct trie node
set node code to code
set children node pointers to NULL

trie node delete function
```

```
destruct trie node

trie create function
initalize trie root with empty code
if sucessful
return trie root
else
return NULL

trie reset function
reset trie to root TrieNode
at end of max code
reset trie by deleting children and set them to NULL

trie delete function
delete sub trie starting from trie rooted at node n by recursively calling n's children
set children pointers to null

trie step function
return pointer to children node representing symbol
if symbol doesn't exist
return NULL
```

word.c

```
word create function
construct the word
length of symbol array given by len
if successful
return word
else
return NULL

word append function
construct new word appended with symbol
if word is empty
new word only contains symbol
return new appended new word

word delete function
destruct word

wt create function
```

create new word table array
word table size of max code is equal to uint16 max
word tabe is initialized at index empty code

wt reset function
reset world table to empty word
set all other words in table to NULL

io.c

read bytes function
perform read with wrapper function to loop calls
loop until all bytes are read
return number of read bytes

write bytes function
perform read with wrapper function to loop calls
loop until all bytes are written
return number of written bytes

read header file
read sizeof file header bytes from input file
bytes are read into supplied header
swap endianness if byte order is not little endian
verify magic number

write header function
write sizeof fileheader bytes into output file
bytes are from supplied header
swap endianness if byte order is not little endian

read sym function
index track read symbol in buffer
once all symbols are processed, read another block
if less than a block is read
end of buffer is updated
if there are symbols to be read
return true
else
return false

write pair function
write pair to outfile
pair is comprised of a code and a symbol

```
bits of code are buffered first starting from LSB
bits of symbol are buffered next starting from LSB
code buffered has bit length of bitlen
write buffer when it is filled

flush pairs function
writes out any remaining pairs to output file

read pair function
read pair from input file
place read code to code
one all bits are processed read another block
first bitlen bits are code starting from LSB
last 8 bits of pair are symbol starting from LSB
if there are pairs left to read in butter
return true
else
return false
if read code is not STOP CODE
there are pairs left to read
```

encode.c

```
main function
get opt
case v: print compression statistics
case i: specify infile
case o: specify outfile

open infile
open outfile
obtain file size and protection bit mask
write header with header
follow compression algorithm specified in asgn6.pdf
print out statistics
close infile and outfile
```

decode.c

```
main function
get opt
case v: print decompression statistics
case i: specify infile
case o: specify outfile

open infile
open outfile
read header with header
verify magic number
follow decompression algorithm specified in asgn6.pdf
print out statistics
close infile and outfile
```

Structure

-the functions in trie.c will implement the trie, its nodes, and the trie maintenance for the assignment.

-the word.c functions implements the word and word table for the assignment as well as functions that maintain word such as wt reset and wt delete.

-the io.c functions implements I/O modules and are responsbile for reading and writing bytes and pairs for the assignment.

-encode.c file implements the compression algorithm using word.c and io.c functions. The verbose is obtained from total bits and total syms which are incremented in io.c

-decode.c file implements the decompression algorithm using trie.c, word.c, and io.c functions. The verbose is obtained from total bits and total syms which are incremented in io.c

Credit

-cse13s discord
-piazza
-cse13s tutoring