Encode

Read sym

↓

buffer - pair

↓

write pair in outfile

Decode

Read pair

↓

buffer - word // put pair back t

↓

write - word.

how main work

io.c

```
static uint8_t bytebuf [4096] ;
static uint8_t bytebuffindex = 0 ;

int read_bytes :
    int bytes_read ; = 2 // anything to get into the loop
    int count = to_read
    int total = 0
    while (( total != to_read ) && (bytes_read != 0)
        bytes_read = read (infile , buf +total , count )
        total += bytes_read
        count -= bytes_read ;
    return Total.
```

write_byter

similar to read_bytes

```
read_header ( int infile , FileHeader *header )
    read_bytes (infile , uint8_t *header , sizeof (FileHeader) ;

write_header   // inc total bite, for stat.
    ~~Wr~~ read_header similar.

                       read here                  put in here
bool read_sym (int infile , uint8_t *byte)
    static bool check;
    uint16_t temp
```

Continue on DESIGN 2

```
if  bytebuffindex == 0 {
    int  num_read  =  read_bytes(infile, buffer, 4096
    if num_read = 4096 {
                                          // if really reader 4096
will check            check = true
reading       else                            then check == true
                  temp = num_read
                   False
 byte  = bytebuf [bytebuffindex] ;      //if not,
 bytebuffindex  = ( bytebuffindex + 1)  % 4096 ;

 return   check || bytebuffindex != temp ;

static  uint8_t  pairbuf [4096]
static  uint16_t  pairbuffindex = 0
void  buffer_pair (outfile, code, sym, bit_len)
      // buffer  code

    for  i = 0 :  bit  len
         add code to pairbuf
         make  sure you  going                1001001
         pairbuf index ++

         if   pair b index/8 == 4096
                 write _byte
                 pairbuffindex = 0 ;


          // buffer  the  sym
      for  0 : 8
            add sym to pairbuf                   101 ..
             pairbufindex ++.
            if   pair b index/8 == 4096
                   write _byte
                   pairbuffindex = 0 ;
```

```
flush_pairs (int outfile) {
    if ( pairbuffindex != 0) {
        write_bytes (outfile ,pairbuf , bitindex in bytes )
                                        // write a helper fun
        pairbufindex = 0                that converts bit to bytes


bool  read_pair ( infile, *code, *sym , bit_len)

    for i = 0 : bit_len {

        if pairbuffindex == 0 :
            read_bytes (infile, pairbuf ,4096)
        *code |=


    for i = 0 : 8 {

        if pairbuffindex == 0 :
            read_bytes (infile, pairbuf ,4096)
        *sym |=

                // reuse buffer_byte,


void buffer_word ( outfile , *w) {

    for i =0 : w -> len

        bytebuf [ bytebuffindex] = w -> symbols [i]

        bytebufindex += 1

        if bytebuindex == 4096

            write_bytes (outfile ,bytebuf ,4096)

            bytebuffindex =0,
void flush_words ( outfile) {
    if ( bytebufindex != 0) {
        write_bytes (outfile, bytebuf , bytebuf index)
        bytebuffindex = 0;
```

int infile = STDIN_FILENO

int outfile = STDOUT _FILENO

Encode . c .

main

getopt —— -i : infile ← open (READ only)

-o : outfile ← open (O_WRONLY | O_creat |
O_TRUNC)

-v : stat .

infile

outfile

FileHeader hd = {0, 0}

hd. magic = 0x8badbeef.

header ?

struct stat a -|

fstat (infile , & header_p)   ← take all info from here   ← put in here

int fstat (int fildes, struct stat * buf)

// fstat // get the protection number from input file

hd.protection = header_p. $\underbrace{\text{st\_mode}}_{\text{protection stored in here by fstat}}$

fchmode (outfile , hd.protection) // for decoder.

write_header

then copy pusucode in lab to c

delete (trie)

Makefile

Target

make hatterspeak

all : encode decode

make encode

make decode.

Decode

- getopt
- File header h

  read header (infile , &h)

  h.magic ≟ magic

  fchmod (outfile , h.protection)

  decompress (infile , outfile)

  close (infile)

  close (outfile)