```
Licensed under the Apache License, Version 2.0 (the "License");
                                                                                                                                                                                                                                                                                                              log.Read.Printf("parseTrailerInfo: Root object: %s\n", *xRefTable.Root)
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if s[9] = 0 \times 0A  {
      you may not use this file except in compliance with the License.
                                                                                                                                                           log.Read.Printf("parseObjectStream begin: decoding %d objects.\n", osd.ObjCount
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 eolCount = 2
      You may obtain a copy of the License at
                                                                                                                                                                                                                                                                                                          if xRefTable.Info = nil {
                                                                                                                                                           decodedContent := osd.Content
                                                                                                                                                                                                                                                                                                                                                                                                                                                          http://www.apache.org/licenses/LICENSE-2.0
                                                                                                                                                                                                                                                                                                             infoObjRef := d.IndirectRefEntry("Info")
                                                                                                                                                          prolog := decodedContent[:osd.FirstObjOffset]
                                                                                                                                                                                                                                                                                                                                                                                                                                                             return nil, 0, errCorruptHeader
                                                                                                                                                                                                                                                                                                              if infoObjRef ≠ nil {
     Ounless required by applicable law or agreed to in writing, software
                                                                                                                                                                                                                                                                                                                 xRefTable.Info = infoObjRef
                                                                                                                                                          objs := strings.Fields(string(prolog))
     1 distributed under the License is distributed on an "AS IS" BASIS,
                                                                                                                                                                                                                                                                                                                  log.Read.Printf("parseTrailerInfo: Info object: %s\n", *xRefTable.Info)
                                                                                                                                                                                                                                                                                                                                                                                                                                                         log.Read.Printf("headerVersion: end, found header version: %s\n", pdfVersion)
     2 WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
                                                                                                                                                              return errors.New("pdfcpu: parseObjectStream: corrupt object stream dict")
      See the License for the specific language governing permissions and
                                                                                                                                                                                                                                                                                                                                                                                                                                                         return &pdfVersion, eolCount, nil
      limitations under the License.
                                                                                                                                                                                                                                                                                                          if xRefTable.ID = nil {
                                                                                                                                                                                                                                                                                                              idArray := d.ArrayEntry("ID")
     7 package pdfcpu
                                                                                                                                                          var objArray Array
                                                                                                                                                                                                                                                                                                              if idArray \neq nil
                                                                                                                                                                                                                                                                                                                 xRefTable.ID = idArray
                                                                                                                                                                                                                                                                                                                  log.Read.Printf("parseTrailerInfo: ID object: %s\n", xRefTable.ID)
                                                                                                                                                          var offsetOld int
                                                                                                                                                                                                                                                                                                                                                                                                                                                  039 func bypassXrefSection(ctx *Context) error {
                                                                                                                                                                                                                                                                                                              } else if xRefTable.Encrypt ≠ nil {
                                                                                                                                                                                                                                                                                                                  return errors.New("pdfcpu: parseTrailerInfo: missing entry \"ID\"")
                                                                                                                                                                                                                                                                                                                                                                                                                                                       var z int64
                                                                                                                                                          for i := 0; i < len(objs); i += 2 {
                                                                                                                                                                                                                                                                                                                                                                                                                                                        g := FreeHeadGeneration
                                                                                                                                                                                                                                                                                                                                                                                                                                                         ctx.Table[0] = &XRefTableEntry{
                                                                                                                                                              offset, err := strconv.Atoi(objs[i+1])
                                                                                                                                                                                                                                                                                                                                                                                                                                                            Free:
                                                                                                                                                              if err ≠ nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                            Offset: &z,
                                                                                                                                                                                                                                                                                                          log.Read.Println("parseTrailerInfo end")
                                                                                                                                                                                                                                                                                                                                                                                                                                                             Generation: &g}
                                                                                                                                                                                                                                                                                                          return ni
                                                                                                                                                              offset += osd.FirstObjOffset
           "github.com/pdfcpu/pdfcpu/pkg/log"
"github.com/pkg/errors"
                                                                                                                                                                                                                                                                                                                                                                                                                                                        eolCount := ctx.Read.EolCount
                                                                                                                                                                                                                                                                                                                                                                                                                                                        var off, offset int64
                                                                                                                                                                                                                                                                                                   702            <mark>func parseTrailerDict</mark>(trailerDict Dict, ctx *Context) (*int64, error) {
                                                                                                                                                                  dstr := string(decodedContent[offsetOld:offset])
                                                                                                                                                                                                                                                                                                                                                                                                                                                         rd, err := newPositionedReader(rs, &offset)
                                                                                                                                                                  log.Read.Printf("parseObjectStream: objString = %s\n", dstr)
                                                                                                                                                                                                                                                                                                          log.Read.Println("parseTrailerDict begin")
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if err ≠ nil {
          defaultBufSize = 1024
                                                                                                                                                                                                                                                                                                                                                                                                                                                             return err
                                                                                                                                                                  if err ≠ nil {
                                                                                                                                                                                                                                                                                                          xRefTable := ctx.XRefTable
                                                                                                                                                                      return err
                                                                                                                                                                                                                                                                                                         err := parseTrailerInfo(trailerDict, xRefTable)
                                                                                                                                                                                                                                                                                                                                                                                                                                                         s := bufio.NewScanner(rd)
                                                                                                                                                                                                                                                                                                                                                                                                                                                         s.Split(scanLines)
                                                                                                                                                                   log.Read.Printf("parseObjectStream: [%d] = obj %s:\n%s\n", i/2-1, objs[i
                                                                                                                                                                                                                                                                                                              return nil, err
        unc ReadFile(inFile string, conf *Configuration) (*Context, error) {
                                                                                                                                                                   objArray = append(objArray, o)
                                                                                                                                                                                                                                                                                                                                                                                                                                                         bb := []byte{}
                                                                                                                                                                                                                                                                                                                                                                                                                                                         var (
                                                                                                                                                                                                                                                                                                          if arr := trailerDict.ArrayEntry("AdditionalStreams"); arr ≠ nil {
           log.Info.Printf("reading %s..\n", inFile)
                                                                                                                                                                                                                                                                                                             log.Read.Printf("parseTrailerInfo: found AdditionalStreams: %s\n", arr)
                                                                                                                                                                                                                                                                                                                                                                                                                                                           withinObj bool
                                                                                                                                                              if i = len(objs)-2  {
                                                                                                                                                                                                                                                                                                                                                                                                                                                            withinXref bool
                                                                                                                                                                                                                                                                                                             a ≔ Array{}
           f, err := os.Open(inFile)
                                                                                                                                                                  dstr := string(decodedContent[offset:])
                                                                                                                                                                                                                                                                                                                                                                                                                                                             withinTrailer bool
                                                                                                                                                                                                                                                                                                              for _, value := range arr {
           if err ≠ nil {
                                                                                                                                                                  log.Read.Printf("parseObjectStream: objString = %s\n", dstr)
                                                                                                                                                                                                                                                                                                                 if indRef, ok := value.(IndirectRef); ok {
              return nil, errors.Wrapf(err, "can't open %q", inFile)
                                                                                                                                                                  o, err := compressedObject(dstr)
                                                                                                                                                                                                                                                                                                                     a = append(a, indRef)
                                                                                                                                                                  if err ≠ nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                             line, err := scanLineRaw(s)
           defer func() -
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if err ≠ nil {
                                                                                                                                                                                                                                                                                                              xRefTable.AdditionalStreams = &a
             f.Close()
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 break
                                                                                                                                                                  log.Read.Printf("parseObjectStream: [%d] = obj %s:\n%s\n", i/2, objs[i],
                                                                                                                                                                                                                                                                                                         offset := trailerDict.Prev()
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if withinXref {
          return Read(f, conf)
                                                                                                                                                                  objArray = append(objArray, o)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 offset += int64(len(line) + eolCount)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 if withinTrailer {
                                                                                                                                                                                                                                                                                                             log.Read.Printf("parseTrailerDict: previous xref table section offset:%d\n'
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     bb = append(bb, ' ')
                                                                                                                                                               offsetOld = offset
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     bb = append(bb, line...)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     i := strings.Index(line, "startxref")
      func Read(rs io.ReadSeeker, conf *Configuration) (*Context, error) {
                                                                                                                                                                                                                                                                                                         offsetXRefStream := trailerDict.Int64Entry("XRefStm")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     if i ≥ 0 {
                                                                                                                                                                                                                                                                                                                                                                                                                                              localhost:49203
localhost:49203
                                   /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
           log.Read.Println("Read: begin")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                           _, err = processTrailer(ctx, s, string(bb))
                                                                                                                                                           log.Read.Println("parseObjectStream end")
                                                                                                                                                                                                                                                                                                              if !ctx.Reader15 & xRefTable.Version() ≥ V14 & !ctx.Read.Hybrid {
          ctx, err := NewContext(rs, conf)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                           return err
                                                                                                                                                                                                                                                                                                                return nil, errors.Errorf("parseTrailerDict: PDF1.4 conformant reader:
                                                                                                                                                                                                                                                                                                       ound incompatible version: %s", xRefTable.VersionString())
                                                                                                                                                          return nil
              return nil, err
                                                                                                                                                                                                                                                                                                              log.Read.Println("parseTrailerDict end")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 i ≔ strings.Index(line, "trailer")
           if ctx.Reader15 {
                                                                                                                                                                                                                                                                                                              return offset, nil
             log.Info.Println("PDF Version 1.5 conforming reader")
                                                                                                                                                        unc extractXRefTableEntriesFromXRefStream(buf []byte, xsd *XRefStreamDict, ctx
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     bb = append(bb, line...)
             log.Info.Println("PDF Version 1.4 conforming reader - no object streams or
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     withinTrailer = true
           streams allowed")
                                                                                                                                                          log.Read.Printf("extractXRefTableEntriesFromXRefStream begin")
                                                                                                                                                                                                                                                                                                          if !ctx.Read.Hybrid {
                                                                                                                                                                                                                                                                                                             ctx.Read.Hybrid = true
                                                                                                                                                                                                                                                                                                                                                                                                                                                              i := strings.Index(line, "xref")
           if err = readXRefTable(ctx); err ≠ nil {
                                                                                                                                                                                                                                                                                                              ctx.Read.UsingXRefStreams = true
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if i ≥ 0 {
              return nil, errors.Wrap(err, "Read: xRefTable failed")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 offset += int64(len(line) + eolCount)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 withinXref = true
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if !withinObj {
                                                                                                                                                          i1 := xsd.W[0]
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 i := strings.Index(line, "obj")
                                                                                                                                                          i2 := xsd.W[1]
           if err = dereferenceXRefTable(ctx, conf); err ≠ nil {
                                                                                                                                                                                                                                                                                                          if ctx.Reader15 {
                                                                                                                                                          i3 := xsd.W[2]
                                                                                                                                                                                                                                                                                                             if err := parseHybridXRefStream(offsetXRefStream, ctx); err ≠ nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                                    withinObj = true
                                                                                                                                                                                                                                                                                                                 return nil, err
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     off = offset
                                                                                                                                                          xrefEntryLen := i1 + i2 + i3
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     bb = append(bb, line[:i+3]...)
                                                                                                                                                          log.Read.Printf("extractXRefTableEntriesFromXRefStream: begin xrefEntryLen =
           if *ctx.XRefTable.Size < len(ctx.XRefTable.Table) {</pre>
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 offset += int64(len(line) + eolCount)
              *ctx.XRefTable.Size = len(ctx.XRefTable.Table)
                                                                                                                                                                                                                                                                                                          log.Read.Println("parseTrailerDict end")
                                                                                                                                                          if len(buf)%xrefEntryLen > 0 {
                                                                                                                                                             return errors.New("pdfcpu: extractXRefTableEntriesFromXRefStream: corrupt
                                                                                                                                                                                                                                                                                                          return offset, nil
           log.Read.Println("Read: end")
                                                                                                                                                                                                                                                                                                                                                                                                                                                             offset += int64(len(line) + eolCount)
                                                                                                                                                                                                                                                                                                   762 func scanLineRaw(s *bufio.Scanner) (string, error) {
           return ctx, nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                            bb = append(bb, ' ')
bb = append(bb, line...)
                                                                                                                                                          log.Read.Printf("extractXRefTableEntriesFromXRefStream: objCount:%d %v\n",
                                                                                                                                                                                                                                                                                                         if ok := s.Scan(); !ok {
                                                                                                                                                                                                                                                                                                            if s.Err() \neq nil {
                                                                                                                                                         jCount, xsd.Objects)
                                                                                                                                                                                                                                                                                                                                                                                                                                                             i = strings.Index(line, "endobj")
                                                                                                                                                                                                                                                                                                                 return "", s.Err()
                                                                                                                                                          log.Read.Printf("extractXRefTableEntriesFromXRefStream: len(buf):%d
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 l := string(bb)
                                                                                                                                                            unt*xrefEntryLen:%d\n", len(buf), objCount*xrefEntryLen)
                                                                                                                                                                                                                                                                                                              return "", errors.New("pdfcpu: scanLineRaw: returning nothing")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 objNr, generation, err ≔ parseObjectAttributes(&l)
                                                                                                                                                                                                                                                                                                          return s.Text(), nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     return err
       func scanLines(data []byte, atEOF bool) (advance int, token []byte, err error) {
                                                                                                                                                              return errors.New("pdfcpu: extractXRefTableEntriesFromXRefStream: corrupt
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 of := off
           if atEOF \delta \theta len(data) = 0 {
                                                                                                                                                                                                                                                                                                    func scanLine(s *bufio.Scanner) (s1 string, err error) {
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 ctx.Table[*objNr] = &XRefTableEntry{
              return 0, nil, nil
                                                                                                                                                                                                                                                                                                            s1, err = scanLineRaw(s)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     Offset: &of,
                                                                                                                                                                                                                                                                                                             if err ≠ nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     Generation: generation}
           indCR := bytes.IndexByte(data, '\r')
                                                                                                                                                                                                                                                                                                                 return "", err
           indLF := bytes.IndexByte(data, '\n')
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 withinObj = false
                                                                                                                                                          bufToInt64 := func(buf []byte) (i int64) {
                                                                                                                                                                                                                                                                                                              if len(s1) > 0 {
           switch {
                                                                                                                                                                                                                                                                                                                  break
                                                                                                                                                              for _, b := range buf {
                                                                                                                                                                                                                                                                                                                                                                                                                                                        return ni
          case indCR \geqslant 0 & indLF \geqslant 0:
                                                                                                                                                                 i |= int64(b)
             if indCR < indLF {</pre>
                  if indLF = indCR+1 {
                                                                                                                                                                                                                                                                                                          i := strings.Index(s1, "%")
                                                                                                                                                                                                                                                                                                                                                                                                                                                   33 |func buildXRefTableStartingAt(ctx *Context, offset *int64) error {
                                                                                                                                                               return
                       return indLF + 1, data[0:indCR], nil
                                                                                                                                                                                                                                                                                                            s1 = s1[:i]
                                                                                                                                                                                                                                                                                                                                                                                                                                                        log.Read.Println("buildXRefTableStartingAt: begin")
                                                                                                                                                                                                                                                                                                                                                                                                                                              localhost:49203
                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                   return indCR + 1, data[0:indCR], ni
                                                                                                                                                           for i := 0; i < len(buf) & j < len(xsd.Objects); i += xrefEntryLen {</pre>
                                                                                                                                                               objectNumber := xsd.Objects[j]
                                                                                                                                                                                                                                                                                                                                                                                                                                                         hv, eolCount, err := headerVersion(rs)
              return indLF + 1, data[0:indLF], nil
                                                                                                                                                                                                                                                                                                  792 func isDict(s string) (bool, error) {
                                                                                                                                                                                                                                                                                                                                                                                                                                                            return err
                                                                                                                                                              c2 := bufToInt64(buf[i2Start : i2Start+i2])
                                                                                                                                                                                                                                                                                                        o, err ≔ parseObject(&s)
                                                                                                                                                              c3 := bufToInt64(buf[i2Start+i2 : i2Start+i2+i3])
              return indCR + 1, data[0:indCR], nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                         ctx.HeaderVersion = hv
                                                                                                                                                                                                                                                                                                             return false, err
                                                                                                                                                               var xRefTableEntry XRefTableEntry
                                                                                                                                                                                                                                                                                                                                                                                                                                                         ctx.Read.EolCount = eolCount
                                                                                                                                                               switch buf[i] {
                                                                                                                                                                                                                                                                                                                                                                                                                                                          for offset ≠ nil {
                                                                                                                                                                                                                                                                                                          return ok, nil
              return indLF + 1, data[0:indLF], nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                             rd, err := newPositionedReader(rs, offset)
                                                                                                                                                                                                                                                                                                  801 | func scanTrailer(s *bufio.Scanner, line string) (string, error) {
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if err ≠ nil {
                                                                                                                                                                   log.Read.Printf("extractXRefTableEntriesFromXRefStream: Object #%d is
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 return err
                                                                                                                                                                ext free is object#%d, generation=%d\n", objectNumber, c2, c3)
                                                                                                                                                                                                                                                                                                        var buf bytes.Buffer
                                                                                                                                                                  g := int(c3)
                                                                                                                                                                                                                                                                                                                                                                                                                                                            s := bufio.NewScanner(rd)
              return len(data), data, nil
                                                                                                                                                                  xRefTableEntry =
                                                                                                                                                                                                                                                                                                                                                                                                                                                             s.Split(scanLines)
                                                                                                                                                                      XRefTableEntry{
                                                                                                                                                                                                                                                                                                          log.Read.Printf("line: <%s>\n", line)
                                                                                                                                                                                                                                                                                                                                                                                                                                                              line, err := scanLine(s)
                                                                                                                                                                           Compressed: false,
           return 0, nil, nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if err \neq nil {
                                                                                                                                                                           Offset: &c2,
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 return err
                                                                                                                                                                           Generation: &g}
                                                                                                                                                                                                                                                                                                            i = strings.Index(line, "<<")
    40 func newPositionedReader(rs io.ReadSeeker, offset *int64) (*bufio.Reader, error) {
                                                                                                                                                                                                                                                                                                                                                                                                                                                             log.Read.Printf("line: <%s>\n", line)
                                                                                                                                                                                                                                                                                                                 break
           if _, err := rs.Seek(*offset, io.SeekStart); err ≠ nil {
                                                                                                                                                                   log.Read.Printf("extractXRefTableEntriesFromXRefStream: Object #%d is ir
                                                                                                                                                                                                                                                                                                              line, err = scanLine(s)
              return nil, err
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if strings.TrimSpace(line) = "xref" {
                                                                                                                                                              offset=%d, generation=%d\n", objectNumber, c2, c3)
                                                                                                                                                                                                                                                                                                              log.Read.Printf("line: <%s>\n", line)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 log.Read.Println("buildXRefTableStartingAt: found xref section")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 if offset, err = parseXRefSection(s, ctx); err ≠ nil {
                                                                                                                                                                                                                                                                                                              if err ≠ nil {
           log.Read.Printf("newPositionedReader: positioned to offset: %d\n", *offset)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     return err
                                                                                                                                                                  xRefTableEntry =
                                                                                                                                                                      XRefTableEntry{
           return bufio.NewReader(rs), nil
                                                                                                                                                                          Free: false,
                                                                                                                                                                           Compressed: false,
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 log.Read.Println("buildXRefTableStartingAt: found xref stream")
                                                                                                                                                                           Offset: &c2,
                                                                                                                                                                                                                                                                                                        buf.WriteString(line)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 ctx.Read.UsingXRefStreams = true
                                                                                                                                                                           Generation: &g}
                                                                                                                                                                                                                                                                                                        buf.WriteString(" ")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 rd, err = newPositionedReader(rs, offset)
                                                                                                                                                                                                                                                                                                          log.Read.Printf("scanTrailer dictBuf after start tag: <%s>\n", line)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 if err \neq nil
     func offsetLastXRefSection(ctx *Context) (*int64, error) {
                                                                                                                                                               case 0×02:
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     return err
         rs := ctx.Read.rs
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 if offset, err = parseXRefStream(rd, offset, ctx); err ≠ nil {
                                                                                                                                                                  log.Read.Printf("extractXRefTableEntriesFromXRefStream: Object #%d is
                                                                                                                                                                                                                                                                                                                                                                                                                                                                      log.Read.Printf("bypassXRefSection after %v\n", err)
                                                                                                                                                         mpressed at obj %5d[%d]\n", objectNumber, c2, c3)
                                                                                                                                                                                                                                                                                                          for {
                                                                                                                                                                  objNumberRef := int(c2)
              prevBuf, workBuf []byte
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     return bypassXrefSection(ctx)
                                                                                                                                                                  objIndex := int(c3)
              bufSize
                              int64 = 512
                                                                                                                                                                                                                                                                                                              if len(line) = 0 {
              offset
                               int64
                                                                                                                                                                                                                                                                                                                  line, err = scanLine(s)
                                                                                                                                                                  xRefTableEntry =
                                                                                                                                                                                                                                                                                                                  if err ≠ nil {
                                                                                                                                                                      XRefTableEntry{
                                                                                                                                                                                                                                                                                                                     return "", err
                                                                                                                                                                           Free:
           for i := 1; offset = 0; i++ {
                                                                                                                                                                                                                                                                                                                                                                                                                                                         log.Read.Println("buildXRefTableStartingAt: end")
                                                                                                                                                                           Compressed: true,
                                                                                                                                                                                                                                                                                                                  buf.WriteString(line)
                                                                                                                                                                           ObjectStream: &objNumberRef,
              off, err := rs.Seek(-int64(i)*bufSize, io.SeekEnd)
                                                                                                                                                                                                                                                                                                                  buf.WriteString(" ")
                                                                                                                                                                                                                                                                                                                                                                                                                                                         return nil
                                                                                                                                                                           ObjectStreamInd: &objIndex}
              if err ≠ nil {
                                                                                                                                                                                                                                                                                                                  log.Read.Printf("scanTrailer dictBuf next line: <%s>\n", line)
                   return nil, errors.New("pdfcpu: can't find last xref section")
                                                                                                                                                                  ctx.Read.ObjectStreams[objNumberRef] = true
                                                                                                                                                                                                                                                                                                              i = strings.Index(line, "<<")
              log.Read.Printf("scanning for offsetLastXRefSection starting at %d\n", off)
                                                                                                                                                                                                                                                                                                              if i < 0 {
                                                                                                                                                               if ctx.XRefTable.Exists(objectNumber) {
               curBuf := make([]byte, bufSize)
                                                                                                                                                                                                                                                                                                                  j = strings.Index(line, ">>")
                                                                                                                                                                   log.Read.Printf("extractXRefTableEntriesFromXRefStream: Skip entry %d
                                                                                                                                                                                                                                                                                                                                                                                                                                                 localhost:49203
                                                                                                                                               localhost:49203
                                                                                                                                                                                                                                                                                               localhost:49203
                                                                                                                                                                                                                                                                                                                                                                                                                                              localhost:49203
                                   /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
               if err ≠ nil {
                                                                                                                                                               } else {
                                                                                                                                                                                                                                                                                                                                                                                                                                                          log.Read.Println("readXRefTable: begin")
                                                                                                                                                                  ctx.Table[objectNumber] = &xRefTableEntry
                   return nil, err
                                                                                                                                                                                                                                                                                                                           ok, err := isDict(buf.String())
                                                                                                                                                                                                                                                                                                                                                                                                                                                         offset, err := offsetLastXRefSection(ctx)
                                                                                                                                                                                                                                                                                                                          if err = nil \delta \delta ok {
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if err ≠ nil {
               workBuf = curBuf
                                                                                                                                                                                                                                                                                                                               return buf.String(), nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                             return
              if prevBuf ≠ nil {
                   workBuf = append(curBuf, prevBuf...)
                                                                                                                                                                                                                                                                                                                      } else {
                                                                                                                                                                                                                                                                                                                                                                                                                                                        err = buildXRefTableStartingAt(ctx, offset)
                                                                                                                                                          log.Read.Println("extractXRefTableEntriesFromXRefStream: end")
               j := strings.LastIndex(string(workBuf), "startxref")
                                                                                                                                                                                                                                                                                                                                                                                                                                                            return errors.Wrap(err, "readXRefTable: unexpected eof")
                                                                                                                                                          return nil
                                                                                                                                                                                                                                                                                                                      line = line[j+2:]
                  prevBuf = curBuf
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if err ≠ nil {
                                                                                                                                                        unc xRefStreamDict(ctx *Context, o Object, objNr int, streamOffset int64)
                   continue
                                                                                                                                                                                                                                                                                                                  line. err = scanLine(s)
                                                                                                                                                                                                                                                                                                                  if err ≠ nil {
              p := workBuf[j+len("startxref"):]
                                                                                                                                                                                                                                                                                                                     return "", err
               posEOF := strings.Index(string(p), "%EOF")
                                                                                                                                                                                                                                                                                                                  buf.WriteString(line)
              if posEOF = -1 {
                                                                                                                                                              return nil, errors.New("pdfcpu: xRefStreamDict: no dict")
                                                                                                                                                                                                                                                                                                                  buf.WriteString(" '
                   return nil, errors.New("pdfcpu: no matching %%EOF for startxref")
                                                                                                                                                                                                                                                                                                                  log.Read.Printf("scanTrailer dictBuf next line: <%s>\n", line)
                                                                                                                                                                                                                                                                                                                                                                                                                                                         err = ctx.EnsureValidFreeList()
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if err ≠ nil {
              p = p[:posEOF]
                                                                                                                                                                                                                                                                                                                                                                                                                                                             return
                                                                                                                                                           streamLength, streamLengthObjNr := d.Length()
               offset, err = strconv.ParseInt(strings.TrimSpace(string(p)), 10, 64)
                                                                                                                                                                                                                                                                                                                  j = strings.Index(line, ">>")
                                                                                                                                                           if streamLength = nil & streamLengthObjNr = nil {
              if err ≠ nil {
                                                                                                                                                              return nil, errors.New("pdfcpu: xRefStreamDict: no \"Length\" entry")
                                                                                                                                                                                                                                                                                                                                                                                                                                                         log.Read.Println("readXRefTable: end")
                  return nil, errors.New("pdfcpu: corrupted last xref section")
                                                                                                                                                                                                                                                                                                                      line = line[i+2:]
                                                                                                                                                           filterPipeline, err := pdfFilterPipeline(ctx, d)
                                                                                                                                                                                                                                                                                                                   } else {
                                                                                                                                                             return nil, err
           log.Read.Printf("Offset last xrefsection: %d\n", offset)
                                                                                                                                                                                                                                                                                                                      if i < j {
                                                                                                                                                                                                                                                                                                                                                                                                                                                   26 func growBufBy(buf []byte, size int, rd io.Reader) ([]byte, error) {
          return &offset, nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                       b := make([]byte, size)
                                                                                                                                                                                                                                                                                                                          line = line[i+2:]
                                                                                                                                                           log.Read.Printf("xRefStreamDict: streamobject #%d\n", objNr)
                                                                                                                                                                                                                                                                                                                      } else {
                                                                                                                                                                                                                                                                                                                                                                                                                                                          _, err := rd.Read(b)
                                                                                                                                                          sd := NewStreamDict(d, streamOffset, streamLength, streamLengthObjNr,
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if err \neq nil {
     func parseXRefTableEntry(s *bufio.Scanner, xRefTable *XRefTable, objectNumber int)
                                                                                                                                                                                                                                                                                                                          if k = 0  {
                                                                                                                                                                                                                                                                                                                                                                                                                                                            return nil, err
                                                                                                                                                          if _, err = loadEncodedStreamContent(ctx, &sd); err ≠ nil {
                                                                                                                                                                                                                                                                                                                               ok, err := isDict(buf.String())
                                                                                                                                                             return nil, err
           log.Read.Println("parseXRefTableEntry: begin")
                                                                                                                                                                                                                                                                                                                              if err = nil \delta \delta ok {
                                                                                                                                                                                                                                                                                                                                   return buf.String(), nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                        return append(buf, b...), nil
           line, err := scanLine(s)
           if err \neq nil {
                                                                                                                                                                                                                                                                                                                            } else {
                                                                                                                                                          if err = saveDecodedStreamContent(nil, &sd, 0, 0, true); err ≠ nil {
              return err
                                                                                                                                                                                                                                                                                                                                                                                                                                                  239 func nextStreamOffset(line string, streamInd int) (off int) {
                                                                                                                                                                                                                                                                                                                              k --
                                                                                                                                                             return nil, errors.Wrapf(err, "xRefStreamDict: cannot decode stream for
                                                                                                                                                          #:%d\n", objNr)
                                                                                                                                                                                                                                                                                                                           line = line[j+2:]
                                                                                                                                                                                                                                                                                                                                                                                                                                                        off = streamInd + len("stream")
           if xRefTable.Exists(objectNumber) {
             log.Read.Printf("parseXRefTableEntry: end - Skip entry %d - already
                                                                                                                                                          return parseXRefStreamDict(&sd)
             ned\n", objectNumber)
              return nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                        for ; line[off] = 0×20; off++ {
           fields := strings.Fields(line)
                                                                                                                                                                                                                                                                                                 899 func processTrailer(ctx *Context, s *bufio.Scanner, line string) (*int64, error) {
                                                                                                                                                       unc parseXRefStream(rd io.Reader, offset *int64, ctx *Context) (prevOffset *int64,
           if len(fields) \neq 3 |
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if line[off] = '\n' {
              len(fields[0]) \neq 10 \mid len(fields[1]) \neq 5 \mid len(fields[2]) \neq 1 
                                                                                                                                                                                                                                                                                                         var trailerString string
              return errors.New("pdfcpu: parseXRefTableEntry: corrupt xref subsection
                                                                                                                                                          log.Read.Printf("parseXRefStream: begin at offset %d\n", *offset)
                                                                                                                                                                                                                                                                                                                                                                                                                                                             return
                                                                                                                                                                                                                                                                                                              trailerString = line[7:]
                                                                                                                                                          buf, endInd, streamInd, streamOffset, err ≔ buffer(rd)
                                                                                                                                                                                                                                                                                                              log.Read.Printf("processTrailer: trailer leftover: <%s>\n", trailerString)
                                   /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                               conv.ParseInt(fields[0], 10, 64)
                                                                                                                                                                                                                                                                                                              log.Read.Printf("line (len %d) <%s>\n", len(line), line)
              return err
                                                                                                                                                                                                                                                                                                                                                                                                                                                              // Skip ODOA eo
                                                                                                                                                           log. Read. Printf("parseXRefStream: endInd=%[1]d(%[1]x) streamInd=%[2]d(%[2]x) \\ \\ \label{eq:log_Read_Printf} \\ \label{eq:log_Read
                                                                                                                                                                                                                                                                                                                                                                                                                                                            if line[off] = '\n' {
                                                                                                                                                                                                                                                                                                          trailerString, err := scanTrailer(s, trailerString)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                off++
           generation, err := strconv.Atoi(fields[1])
                                                                                                                                                          line := string(buf)
           if err \neq nil {
                                                                                                                                                                                                                                                                                                              return nil, err
              return err
                                                                                                                                                                                                                                                                                                                                                                                                                                                         return
                                                                                                                                                                                                                                                                                                          log.Read.Printf("processTrailer: trailerString: (len:%d) <%s>\n",
                                                                                                                                                                                                                                                                                                         (trailerString), trailerString)
          entryType := fields[2]
           if entryType \neq "f" \delta\delta entryType \neq "n" {
                                                                                                                                                                                                                                                                                                                                                                                                                                                   66 func lastStreamMarker(streamInd *int, endInd int, line string) {
                                                                                                                                                           if streamInd < 0 \parallel (endInd > 0 \delta \delta endInd < streamInd) {
             return errors.New("pdfcpu: parseXRefTableEntry: corrupt xref subsection
                                                                                                                                                                                                                                                                                                         o, err := parseObject(&trailerString)
                                                                                                                                                             return nil, errors.New("pdfcpu: parseXRefStream: corrupt pdf file")
                                                                                                                                                                                                                                                                                                          if err \neq nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if *streamInd > len(line)-len("stream") {
                                                                                                                                                                                                                                                                                                                                                                                                                                                             *streamInd = −1
           var xRefTableEntry XRefTableEntry
                                                                                                                                                           l := line[:streamInd]
           if entryType = "n" {
                                                                                                                                                          objectNumber, generationNumber, err ≔ parseObjectAttributes(&l)
                                                                                                                                                                                                                                                                                                              return nil, errors.New("pdfcpu: processTrailer: corrupt trailer dict")
                                                                                                                                                                                                                                                                                                                                                                                                                                                         bufpos := *streamInd + len("stream")
                                                                                                                                                             return nil, err
              log.Read.Printf("parseXRefTableEntry: Object #%d is in use at offset=%d,
                                                                                                                                                                                                                                                                                                          log.Read.Printf("processTrailer: trailerDict:\n%s\n", trailerDict)
             ation=%d\n", objectNumber, offset, generation)
                                                                                                                                                                                                                                                                                                                                                                                                                                                          i := strings.Index(line[bufpos:], "stream")
                                                                                                                                                                                                                                                                                                          return parseTrailerDict(trailerDict, ctx)
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if i < 0 {
                                                                                                                                                          log.Read.Printf("parseXRefStream: xrefstm obj#:%d gen:%d\n", *objectNumber,
                  log.Info.Printf("parseXRefTableEntry: Skip entry for in use object #%d
                                                                                                                                                                                                                                                                                                                                                                                                                                                            *streamInd = −1
                                                                                                                                                           log.Read.Printf("parseXRefStream: dereferencing object %d\n", *objectNumber)
          th offset 0\n", objectNumber)
                                                                                                                                                           o, err := parseObject(&l)
                                                                                                                                                                                                                                                                                                   33 func parseXRefSection(s *bufio.Scanner, ctx *Context) (*int64, error) {
                                                                                                                                                          if err \neq nil
                                                                                                                                                              return nil, errors.Wrapf(err, "parseXRefStream: no object")
                                                                                                                                                                                                                                                                                                          log.Read.Println("parseXRefSection begin")
              xRefTableEntry =
                                                                                                                                                                                                                                                                                                                                                                                                                                                         *streamInd += len("stream") + i
                  XRefTableEntrv{
                                                                                                                                                                                                                                                                                                          line, err := scanLine(s)
                                                                                                                                                           log.Read.Printf("parseXRefStream: we have an object: %s\n", o)
                                                                                                                                                                                                                                                                                                          if err \neq nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                         if endInd > 0 & *streamInd > endInd {
                      Offset: &offset,
                                                                                                                                                           streamOffset += *offset
                       Generation: &generation}
                                                                                                                                                                                                                                                                                                                                                                                                                                                             *streamInd = −1
                                                                                                                                                          sd, err := xRefStreamDict(ctx, o, *objectNumber, streamOffset)
                                                                                                                                                          if err \neq nil {
           log.Read.Printf("parseXRefSection: <%s>\n", line)
                                                                                                                                                              return nil, err
                                                                                                                                                                                                                                                                                                          fields := strings.Fields(line)
              log.Read.Printf("parseXRefTableEntry: Object #%d is unused, next free is
                                                                                                                                                                                                                                                                                                                                                                                                                                                  296 <mark>func buffer</mark>(rd io.Reader) (buf []byte, endInd int, streamInd int, streamOffset int64,
            t#%d, generation=%d\n", objectNumber, offset, generation)
                                                                                                                                                          err = parseTrailerInfo(sd.Dict, ctx.XRefTable)
                                                                                                                                                                                                                                                                                                          for !strings.HasPrefix(line, "trailer") & len(fields) = 2 {
                                                                                                                                                          if err \neq nil {
                                                                                                                                                              return nil, err
                                                                                                                                                                                                                                                                                                              if err = parseXRefTableSubSection(s, ctx.XRefTable, fields); err ≠ nil {
                  XRefTableEntry{
                                                                                                                                                                                                                                                                                                                  return nil, err
                     Free: true,
Offset: &offset,
                       Generation: &generation}
                                                                                                                                                          err = extractXRefTableEntriesFromXRefStream(sd.Content, sd, ctx)
                                                                                                                                                          if err \neq nil {
                                                                                                                                                                                                                                                                                                              if line, err = scanLine(s); err ≠ nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                        endInd, streamInd = -1, -1
           log.Read.Printf("parseXRefTableEntry: Insert new xreftable entry for Object %d\r
                                                                                                                                                                                                                                                                                                                                                                                                                                                        for endInd < 0 \delta \delta streamInd < 0 {
                                                                                                                                                                                                                                                                                                              if len(line) = 0 {
           xRefTable.Table[objectNumber] = &xRefTableEntry
                                                                                                                                                             XRefTableEntry{
                                                                                                                                                                                                                                                                                                                                                                                                                                                            buf, err = growBufBy(buf, defaultBufSize, rd)
                                                                                                                                                                                                                                                                                                                  if line, err = scanLine(s); err ≠ nil {
           log.Read.Println("parseXRefTableEntry: end")
                                                                                                                                                                  Offset: offset,
                                                                                                                                                                   Generation: generationNumber,
                                   /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                                                 /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                                                                                                                                                                              fields = strings.Fields(line)
                                                                                                                                                                                                                                                                                                                                                                                                                                                              line := string(buf)
                                                                                                                                                           log.Read.Printf("parseXRefStream: Insert new xRefTable entry for Object %d\n"
                                                                                                                                                                                                                                                                                                                                                                                                                                                              endInd = strings.Index(line, "endobj")
    88 func parseXRefTableSubSection(s *bufio.Scanner, xRefTable *XRefTable, fields []string
                                                                                                                                                                                                                                                                                                                                                                                                                                                             streamInd = strings.Index(line, "stream")
                                                                                                                                                         ctx.Table[*objectNumber] = &entry
                                                                                                                                                                                                                                                                                                          log.Read.Println("parseXRefSection: All subsections read!")
                                                                                                                                                          ctx.Read.XRefStreams[*objectNumber] = true
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if endInd > 0 & (streamInd < 0 || streamInd > endInd) {
           log.Read.Println("parseXRefTableSubSection: begin")
                                                                                                                                                          prevOffset = sd.PreviousOffset
                                                                                                                                                                                                                                                                                                          if !strings.HasPrefix(line, "trailer") {
                                                                                                                                                                                                                                                                                                              return nil, errors.Errorf("xrefsection: missing trailer dict, line = <%s>"
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 break
           startObjNumber, err := strconv.Atoi(fields[0])
                                                                                                                                                          log.Read.Println("parseXRefStream: end")
           if err ≠ nil {
              return err
                                                                                                                                                          return prevOffset, nil
                                                                                                                                                                                                                                                                                                          log.Read.Println("parseXRefSection: parsing trailer dict..")
                                                                                                                                                                                                                                                                                                                                                                                                                                                             for streamInd > 0 & !keywordStreamRightAfterEndOfDict(line, streamInd) {
           objCount, err := strconv.Atoi(fields[1])
                                                                                                                                                                                                                                                                                                          return processTrailer(ctx, s, line)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 lastStreamMarker(&streamInd, endInd, line)
                                                                                                                                                   28 func parseHybridXRefStream(offset *int64, ctx *Context) error {
           if err \neq nil {
                                                                                                                                                          log.Read.Println("parseHybridXRefStream: begin")
                                                                                                                                                                                                                                                                                                                                                                                                                                                              log.Read.Printf("buffer: endInd=%d streamInd=%d\n", endInd, streamInd)
           log.Read.Printf("detected xref subsection, startObj=%d length=%d\n",
                                                                                                                                                          rd, err := newPositionedReader(ctx.Read.rs, offset)
                                                                                                                                                                                                                                                                                                                                                                                                                                                             if streamInd > 0 {
                                                                                                                                                              return err
           for i := 0; i < objCount; i++ {
              if err = parseXRefTableEntry(s, xRefTable, startObjNumber+i); err ≠ nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 slack := 10 // for optional whitespace + eol (max 2 chars)
need := streamInd + len("stream") + slack
                                                                                                                                                           _, err = parseXRefStream(rd, offset, ctx)
                                                                                                                                                                                                                                                                                                 986 func headerVersion(rs io.ReadSeeker) (v *Version, eolCount int, err error) {
                  return err
                                                                                                                                                          if err ≠ nil {
                                                                                                                                                             return err
                                                                                                                                                                                                                                                                                                         log.Read.Println("headerVersion begin")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 if len(line) < need {</pre>
                                                                                                                                                                                                                                                                                                        var errCorruptHeader = errors.New("pdfcpu: headerVersion: corrupt pdf stream - r
          log.Read.Println("parseXRefTableSubSection: end")
                                                                                                                                                          log.Read.Println("parseHybridXRefStream: end")
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     buf, err = growBufBy(buf, need-len(line), rd)
          return nil
                                                                                                                                                          return nil
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     if err ≠ nil {
                                                                                                                                                                                                                                                                                                                                                                                                                                                                          return nil, 0, 0, 0, err
                                                                                                                                                                                                                                                                                                         if _, err = rs.Seek(0, io.SeekStart); err ≠ nil {
       func compressedObject(s string) (Object, error) {
                                                                                                                                                         nc parseTrailerInfo(d Dict, xRefTable *XRefTable) error {
                                                                                                                                                                                                                                                                                                                                                                                                                                                                     line = string(buf)
           log.Read.Println("compressedObject: begin")
                                                                                                                                                          log.Read.Println("parseTrailerInfo begin")
                                                                                                                                                                                                                                                                                                         buf := make([]byte, 20)
                                                                                                                                                                                                                                                                                                                                                                                                                                                                 streamOffset = int64(nextStreamOffset(line, streamInd))
                                                                                                                                                                                                                                                                                                         if _, err = rs.Read(buf); err ≠ nil {
          o, err := parseObject(&s)
                                                                                                                                                          if _, found := d.Find("Encrypt"); found {
   encryptObjRef := d.IndirectRefEntry("Encrypt")
          if err ≠ nil {
                                                                                                                                                              if encryptObjRef ≠ nil {
                                                                                                                                                                  xRefTable.Encrypt = encryptObjRef
                                                                                                                                                                                                                                                                                                         s := string(buf)
                                                                                                                                                                  log.Read.Printf("parseTrailerInfo: Encrypt object: %s\n",
                                                                                                                                                                                                                                                                                                         prefix := "%PDF-
                                                                                                                                                     *xRefTable.Encrypt)
          d, ok := o.(Dict)
          if !ok {
                                                                                                                                                                                                                                                                                                                                                                                                                                                         return buf, endInd, streamInd, streamOffset, nil
                                                                                                                                                                                                                                                                                                         if len(s) < 8 || !strings.HasPrefix(s, prefix) {</pre>
             // return trivial Object: Integer, Array, etc.
log.Read.Println("compressedObject: end, any other than dict")
                                                                                                                                                                                                                                                                                                             return nil, 0, errCorruptHeader
                                                                                                                                                          if xRefTable.Size = nil {
                                                                                                                                                             size := d.Size()
                                                                                                                                                                                                                                                                                                                                                                                                                                                  B59 func keywordStreamRightAfterEndOfDict(buf string, streamInd int) bool {
                                                                                                                                                                                                                                                                                                         pdfVersion, err := PDFVersion(s[len(prefix) : len(prefix)+3])
                                                                                                                                                              if size = nil {
           streamLength, streamLengthRef := d.Length()
                                                                                                                                                                  return errors.New("pdfcpu: parseTrailerInfo: missing entry \"Size\"")
                                                                                                                                                                                                                                                                                                              return nil, 0, errors.Wrapf(err, "headerVersion: unknown PDF Header Version
          if streamLength = nil & streamLengthRef = nil {
              log.Read.Println("compressedObject: end, dict")
                                                                                                                                                                                                                                                                                                                                                                                                                                                        b := buf[:streamInd]
              return d, nil
                                                                                                                                                               xRefTable.Size = size
                                                                                                                                                                                                                                                                                                         s = strings.TrimLeft(s, "\t\f<u>"</u>)
                                                                                                                                                                                                                                                                                                                                                                                                                                                          // Look for last end of dict marker.
```

/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go

return nil, errors.New("pdfcpu: compressedObject: stream objects are not to be

localhost:49203

if xRefTable.Root = nil {

rootObjRef := d.IndirectRefEntry("Root")

localhost:49203

Copyright 2018 The pdfcpu Authors.

/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go

44 **func parseObjectStream**(osd \*ObjectStreamDict) **error** {

/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go

eolCount = 1

eolCount = 1

 $else if s[0] = 0 \times 0D$ 

eod := strings.LastIndex(b, ">>")

**if** eod < 0 {

localhost:49203

return errors.New("pdfcpu: parseTrailerInfo: missing entry \"Root\"")

xRefTable.Root = rootObjRef

```
/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                              /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                  708 <mark>func dereferencedDict</mark>(ctx *Context, objectNumber int) (Dict, error) {
  ok := strings.TrimSpace(b[eod:]) = ">>"
                                                                                                                                        o, err := dereferencedObject(ctx, objectNumber)
                                                                                                                                           return nil, err
   return ok
                                                                                                                                       d, ok := o.(Dict)
<mark>func buildFilterPipeline</mark>(ctx *Context, filterArray, decodeParmsArr Array, decodeParms
                                                                                                                                           return nil, errors.New("pdfcpu: dereferencedDict: corrupt dict")
  var filterPipeline []PDFFilter
                                                                                                                                        return d, nil
   for i, f := range filterArray {
      filterName, ok := f.(Name)
     if !ok {
                                                                                                                                      func int640bject(ctx *Context, objectNumber int) (*int64, error) {
          return nil, errors.New("pdfcpu: buildFilterPipeline: filterArray elemer
                                                                                                                                        log.Read.Printf("int640bject begin: %d\n", objectNumber)
      if decodeParms = nil || decodeParmsArr[i] = nil {
                                                                                                                                         i, err := dereferencedInteger(ctx, objectNumber)
         filterPipeline = append(filterPipeline, PDFFilter{Name:
                                                                                                                                        if err \neq nil {
 lterName.Value(), DecodeParms: nil})
                                                                                                                                        i64 := int64(i.Value())
      dict, ok := decodeParmsArr[i].(Dict)
      if !ok {
                                                                                                                                        log.Read.Printf("int640bject end: %d\n", objectNumber)
          indRef, ok := decodeParmsArr[i].(IndirectRef)
                                                                                                                                        return &i64, nil
             return nil, errors.Errorf("buildFilterPipeline: corrupt Dict: %s\n
          d, err := dereferencedDict(ctx, indRef.ObjectNumber.Value())
          if err ≠ nil {
             return nil, err
                                                                                                                                      inc readContentStream(rd io.Reader, streamLength int) ([]byte, error) {
                                                                                                                                        log.Read.Printf("readContentStream: begin streamLength:%d\n", streamLength)
                                                                                                                                        buf := make([]byte, streamLength)
     filterPipeline = append(filterPipeline, PDFFilter{Name: filterName.String()
   odeParms: dict})
                                                                                                                                        for totalCount := 0; totalCount < streamLength; {</pre>
                                                                                                                                           count, err := rd.Read(buf[totalCount:])
                                                                                                                                           if err \neq nil {
   return filterPipeline, nil
                                                                                                                                           log.Read.Printf("readContentStream: count=%d, buflen=%d(%X)\n", count,
 inc pdfFilterPipeline(ctx *Context, dict Dict) ([]PDFFilter, error) {
                                                                                                                                           totalCount += count
  log.Read.Println("pdfFilterPipeline: begin")
                                                                                                                                        log.Read.Printf("readContentStream: end\n")
   var err error
                                                                                                                                        return buf, nil
   o, found := dict.Find("Filter")
  if !found {
                         /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                                               /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
                                                                                                                                        log.Read.Printf("LoadEncodedStreamContent: begin\n%v\n", sd)
  var filterPipeline []PDFFilter
                                                                                                                                        var err error
   if indRef, ok := o.(IndirectRef); ok {
     o, err = dereferencedObject(ctx, indRef.ObjectNumber.Value())
      if err ≠ nil {
                                                                                                                                        if sd.Raw ≠ nil {
          return nil, err
                                                                                                                                            return sd.Raw, nil
  if name, ok := o.(Name); ok {
                                                                                                                                        if sd.StreamLength = nil {
                                                                                                                                           if sd.StreamLengthObjNr = nil {
       filterName := name.String()
       o, found := dict.Find("DecodeParms")
                                                                                                                                            sd.StreamLength, err = int640bject(ctx, *sd.StreamLengthObjNr)
      if !found {
                                                                                                                                           if err ≠ nil {
                                                                                                                                                return nil, err
          log.Read.Println("pdfFilterPipeline: end w/o decode parms")
          return append(filterPipeline, PDFFilter{Name: filterName, DecodeParms:
      d, ok := o.(Dict)
                                                                                                                                        newOffset := sd.StreamOffset
                                                                                                                                        rd, err := newPositionedReader(ctx.Read.rs, &newOffset)
          ir, ok := o.(IndirectRef)
                                                                                                                                        if err \neq nil {
          if !ok {
             return nil, errors.Errorf("pdfFilterPipeline: corrupt Dict: %s\n", o
          d, err = dereferencedDict(ctx, ir.ObjectNumber.Value())
          if err ≠ nil {
             return nil, err
                                                                                                                                        rawContent, err := readContentStream(rd, int(*sd.StreamLength))
                                                                                                                                           return nil, err
      log.Read.Println("pdfFilterPipeline: end with decode parms")
       return append(filterPipeline, PDFFilter{Name: filterName, DecodeParms: d}),
                                                                                                                                       sd.Raw = rawContent
   filterArray, ok := o.(Array)
      return nil, errors.Errorf("pdfFilterPipeline: Expected filterArray corrupt,
                                                                                                                                        return rawContent, nil
```

var decodeParmsArr Array

decodeParmsArr, ok = decodeParms.(Array)

log.Read.Println("pdfFilterPipeline: end")

streamLength, streamLengthRef := d.Length()

filterPipeline, err := pdfFilterPipeline(ctx, d)

return filterPipeline, err

**if** streamInd ≤ 0 {

return sd, err

streamOffset += offset

if ctx.EncKey ≠ nil {

**if** err ≠ nil {

int, streamOffset int64, err error) {

return nil, 0, 0, 0, err

**return** nil, 0, 0, 0, err

} **else if** streamInd < 0 { // dict

l = line[:streamInd]

l = line[:endInd]

var objectNr, generationNr \*int

return nil, 0, 0, 0, err

o, err = parseObject(&l)

**if** err  $\neq$  nil {

switch o := obj.(type) {

**return** d, err

if ctx.EncKey ≠ nil {

if ctx.EncKey ≠ nil {

**if** err ≠ nil

if ctx.EncKey ≠ nil {

if err ≠ nil {

entry, ok := ctx.Find(objectNumber)

return o, nil

if entry.Compressed {

**if** err ≠ nil {

if entry.Object = nil {

**if** err  $\neq$  nil {

%d", objectNumber)

entry.Object = o

return entry.Object, nil

return nil, err

i, ok := o.(Integer)

**if** err ≠ nil {

**if** !ok {

localhost:49203

**if** o = nil {

return nil, err

return o, ni

case HexLiteral:

return nil, err

return StringLiteral(\*s1), nil

return StringLiteral(string(bb)), nil

func dereferencedObject(ctx \*Context, objectNumber int) (Object, error) {

return nil, errors.New("pdfcpu: dereferencedObject: unregistered object")

log.Read.Printf("dereferencedObject: dereferencing object %d\n", objectNumber

return nil, errors.Wrapf(err, "dereferencedObject: problem dereferencin

return nil, errors.New("pdfcpu: dereferencedObject: object is nil")

93 **func dereferencedInteger**(ctx \*Context, objectNumber **int**) (\*Integer, error) {

return nil, errors.New("pdfcpu: dereferencedInteger: corrupt integer")

o, err := dereferencedObject(ctx, objectNumber)

o, err := ParseObject(ctx, \*entry.Offset, objectNumber, \*entry.Generation)

err := decompressXRefTableEntry(ctx.XRefTable, objectNumber, entry)

x.AES4Strings, ctx.E.R); err  $\neq$  nil {

**return** nil, err

objectNr, generationNr, err = parseObjectAttributes(&l)

if objNr ≠ \*objectNr || genNr ≠ \*generationNr {

return o, endInd, streamInd, streamOffset, err

} else { // did

} else if streamInd < endInd { // streamdict</pre>

**if** err ≠ nil {

var buf []byte

if err ≠ nil {

line := string(buf)

var l string

localhost:49203

22/05/2020

offset int64) (sd StreamDict, err error) {

if found {

**if** !ok {

/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go

return nil, errors.New("pdfcpu: pdfFilterPipeline: expected decodeParms

filterPipeline, err = buildFilterPipeline(ctx, filterArray, decodeParmsArr,

func streamDictForObject(ctx \*Context, d Dict, objNr, streamInd int, streamOffset,

return sd, errors.New("pdfcpu: streamDictForObject: stream object without

sd = NewStreamDict(d, streamOffset, streamLength, streamLengthRef, filterPipeline

log.Read.Printf("streamDictForObject: end, Streamobject #%d\n", objNr)

func dict(ctx \*Context, d1 Dict, objNr, genNr, endInd, streamInd int) (d2 Dict, err

if endInd  $\geq$  0 & (streamInd < 0 || streamInd > endInd) {

log.Read.Printf("dict: end, #%d\n", objNr)

rd, err = newPositionedReader(ctx.Read.rs, &offset)

buf, endInd, streamInd, streamOffset, err = buffer(rd)

**if** endInd < 0 { // & streamInd ≥ 0, streamdict

\_, err := decryptDeepObject(d1, objNr, genNr, ctx.EncKey, ctx.AES4Strings,

/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go

func object(ctx \*Context, offset int64, objNr, genNr int) (o Object, endInd, streamIn

log.Read.Println("object: big stream, we parse object until stream")

log.Read.Println("object: small object w/o stream, parse until endobj")

log.Read.Println("object: small stream within buffer, parse until stream")

log.Read.Println("object: small obj w/o stream, parse until endobj")

return nil, 0, 0, 0, errors.Errorf("object: non matching objNr(%d) or rationNumber(%d) tags found.", \*objectNr, \*generationNr)

func ParseObject(ctx \*Context, offset int64, objNr, genNr int) (Object, error) {

log.Read.Printf("ParseObject: begin, obj#%d, offset:%d\n", objNr, offset)

d, err := dict(ctx, o, objNr, genNr, endInd, streamInd)

obj, endInd, streamInd, streamOffset, err := object(ctx, offset, objNr, genNr)

return streamDictForObject(ctx, o, objNr, streamInd, streamOffset, offset)

if \_, err = decryptDeepObject(o, objNr, genNr, ctx.EncKey,

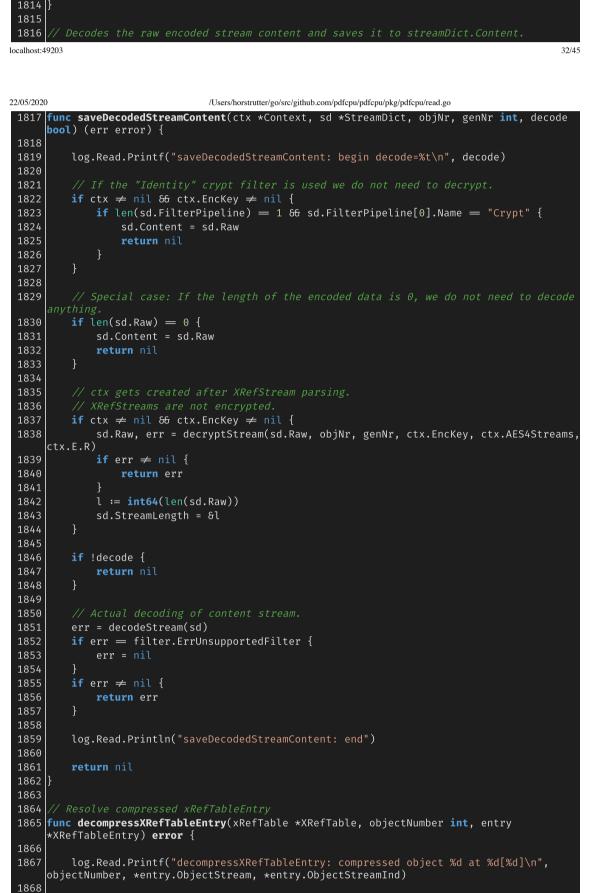
s1, err := decryptString(o.Value(), objNr, genNr, ctx.EncKey,

bb, err := decryptHexLiteral(o, objNr, genNr, ctx.EncKey, ctx.AES4String

/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go

localhost:49203

```
63 func loadEncodedStreamContent(ctx *Context, sd *StreamDict)([]byte, error) {
         log.Read.Println("LoadEncodedStreamContent: end, already in memory.")
             return nil, errors.New("pdfcpu: loadEncodedStreamContent: missing
          log.Read.Printf("LoadEncodedStreamContent: new indirect streamLength:%d\n"
      log.Read.Printf("LoadEncodedStreamContent: seeked to offset:%d\n", newOffset)
      log.Read.Printf("LoadEncodedStreamContent: end: len(streamDictRaw)=%d\n",
                             /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
      <u>c saveDecodedStreamContent(ctx *Co</u>ntext, sd *StreamDict, objNr, genNr int, decode
     ol) (err error) {
```



objectStreamXRefTableEntry, ok := xRefTable.Find(\*entry.ObjectStream)

```
if !ok {
localhost:49203
                                /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
           am %d, no xref table entry", *entry.ObjectStream)
          sd, ok := objectStreamXRefTableEntry.Object.(ObjectStreamDict)
            return errors.Errorf("decompressXRefTableEntry: problem dereferencing objec
          eam %d, no object stream", *entry.ObjectStream)
          o, err := sd.IndexedObject(*entry.ObjectStreamInd)
            return errors.Wrapf(err, "decompressXRefTableEntry: problem dereferencing
            t stream %d", *entry.ObjectStream)
          entry.Object = o
          entry.Generation = &g
         entry.Compressed = false
      log.Read.Printf("decompressXRefTableEntry: end, Obj %d[%d]:\n<%s>\n",
*entry.ObjectStream, *entry.ObjectStreamInd, o)
          return ni
  .899 func logStream(o Object) {
          switch o := o.(type) {
          case StreamDict:
             if o.Content = nil {
                  log.Read.Println("logStream: no stream content")
             if o.IsPageContent {
           case ObjectStreamDict:
              if o.Content = nil {
                  log.Read.Println("logStream: no object stream content")
              } else {
                  log.Read.Printf("logStream: objectStream content = %s\n", o.Content)
              if o.ObjArray = nil {
                  log.Read.Println("logStream: no object stream obj arr")
              } else {
                  log.Read.Printf("logStream: objectStream objArr = %s\n", o.ObjArray)
```

```
log.Read.Println("logStream: no ObjectStreamDict")
935 func decodeObjectStreams(ctx *Context) error {
      log.Read.Println("decodeObjectStreams: begin")
      for k := range ctx.Read.ObjectStreams {
         keys = append(keys, k)
      for _, objectNumber := range keys {
         entry := ctx.XRefTable.Table[objectNumber]
            return errors.Errorf("decodeObjectStream: missing entry for obj#%d\n",
         log.Read.Printf("decodeObjectStreams: parsing object stream for obj#%d\n",
         o, err := ParseObject(ctx, *entry.Offset, objectNumber, *entry.Generation)
         if err \neq nil || o = nil {
             return errors.New("pdfcpu: decodeObjectStreams: corrupt object stream")
         sd, ok := o.(StreamDict)
         if !ok {
             return errors.New("pdfcpu: decodeObjectStreams: corrupt object stream")
         if _, err = loadEncodedStreamContent(ctx, &sd); err ≠ nil {
             return errors.Wrapf(err, "decodeObjectStreams: problem dereferencing
       ct stream %d", objectNumber)
         if err = saveDecodedStreamContent(ctx, &sd, objectNumber, *entry.Generation
              log.Read.Printf("obj %d: %s", objectNumber, err)
             return err
```

```
if !sd.IsObjStm()
              return errors.New("pdfcpu: decodeObjectStreams: corrupt object stream"
          log.Read.Printf("decodeObjectStreams: object stream #%d\n", objectNumber)
          ctx.Read.UsingObjectStreams = true
          osd, err ≔ objectStreamDict(&sd)
             return errors.Wrapf(err, "decodeObjectStreams: problem dereferencing
       ct stream %d", objectNumber)
         log.Read.Printf("decodeObjectStreams: decoding object stream %d:\n",
          if err = parseObjectStream(osd); err ≠ nil {
             return errors.Wrapf(err, "decodeObjectStreams: problem decoding object
       am %d\n", objectNumber)
          if osd.ObjArray = nil {
             return errors.Wrap(err, "decodeObjectStreams: objArray should be set!")
         log.Read.Printf("decodeObjectStreams: decoded object stream %d:\n",
         entry.Object = *osd
      log.Read.Println("decodeObjectStreams: end")
      return nil
021 <mark>func handleLinearizationParmDict</mark>(ctx *Context, obj Object, objNr int) error {
      if ctx.Read.Linearized {
         return nil
      if d, ok := obj.(Dict); ok & d.IsLinearizationParmDict() {
          ctx.Read.Linearized = true
          ctx.LinearizationObjs[objNr] = true
          log.Read.Printf("handleLinearizationParmDict: identified linearizationObj
         a := d.ArrayEntry("H")
```

localhost:49203

```
ctx.Read.BinaryTotalSize += *o.StreamLength
localhost:49203
                                  /Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
              ctx.Read.BinaryTotalSize += *o.StreamLength
          case XRefStreamDict:
              ctx.Read.BinaryTotalSize += *o.StreamLength
   102 <mark>func dereferenceObject</mark>(ctx *Context, objNr int) error {
          xRefTable := ctx.XRefTable
          xRefTableSize := len(xRefTable.Table)
          log.Read.Printf("dereferenceObject: begin, dereferencing object %d\n", objNr)
          entry := xRefTable.Table[objNr]
           if entry.Free {
              log.Read.Printf("free object %d\n", objNr)
              return nil
           if entry.Compressed {
              err := decompressXRefTableEntry(xRefTable, objNr, entry)
                  return err
               //log.Read.Printf("dereferenceObject: decompressed entry,
sed=%v\n%s\n", entry.Compressed, entry.Object)
              return nil
          log.Read.Printf("in use object %d\n", objNr)
          if entry.Offset = nil || *entry.Offset = 0 {
             log.Read.Printf("dereferenceObject: already decompressed or used object w/o
            t \rightarrow ignored")
              return ni
          o := entry.Object
         if o ≠ nil {
              updateBinaryTotalSize(ctx, o)
             log.Read.Printf("handleCachedStreamDict: using cached object %d of
           <%s>\n", objNr, xRefTableSize, entry.Object)
              return nil
          log.Read.Printf("dereferenceObject: dereferencing object %d\n", objNr)
```

/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pdfcpu/read.go

return errors.Errorf("handleLinearizationParmDict: corrupt linearizat

return errors.Errorf("handleLinearizationParmDict: corrupt linearizatio

return errors.Errorf("handleLinearizationParmDict: corrupt linearization
t at obj:%d - corrupt array entry H, needs Integer values", objNr)

return errors.Errorf("handleLinearizationParmDict: corrupt

return errors.Wrapf(err, "dereferenceObject: problem dereferencing stream %d"

err = saveDecodedStreamContent(ctx, sd, objNr, genNr, ctx.DecodeAllStreams)

nearization dict at obj:%d - corrupt array entry H, needs Integer values", objNr)

t at obj:%d - corrupt array entry H, needs length 2 or 4", objNr)

at obj:%d - missing array entry H", objNr)

**if** len(a)  $\neq$  2 & len(a)  $\neq$  4 {

offset, ok := a[0].(Integer)

offset64 := **int64**(offset.Value())

ctx.OffsetPrimaryHintTable = &offset64

offset, ok := a[2].(Integer)

offset64 := **int64**(offset.Value()) ctx.OffsetOverflowHintTable = &offset64

if \_, err = loadEncodedStreamContent(ctx, sd); err ≠ nil {

ctx.Read.BinaryTotalSize += \*sd.StreamLength

!085 **func updateBinaryTotalSize**(ctx \*Context, o Object) {

**if** !ok {

**return** ni

**var** err error

return err

localhost:49203

localhost:49203

**switch** o := o.(**type**) {

```
/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
   o, err := ParseObject(ctx, *entry.Offset, objNr, *entry.Generation)
     return errors.Wrapf(err, "dereferenceObject: problem dereferencing object %d
   err = handleLinearizationParmDict(ctx, o, objNr)
   if err ≠ nil {
       return err
   if _, ok := o.(ObjectStreamDict); ok {
     return errors.Errorf("dereferenceObject: object stream should already be
      erenced at obj:%d", objNr)
   if _, ok := o.(XRefStreamDict); ok {
      return errors.Errorf("dereferenceObject: xref stream should already be
     erenced at obj:%d", objNr)
   if sd, ok := o.(StreamDict); ok {
       err = loadStreamDict(ctx, &sd, objNr, *entry.Generation)
           return err
       entry.Object = sd
log.Read.Printf("dereferenceObject: end obj %d of %d\n<%s>\n", objNr,
xRefTableSize, entry.Object)
   logStream(entry.Object)
   return ni
   nc processDictRefCounts(xRefTable *XRefTable, d Dict) {
   for _, e := range d {
       switch o1 := e.(type) {
       case IndirectRef:
           entry, ok := xRefTable.FindTableEntryForIndRef(&o1)
           if ok {
               entry.RefCount++
        case Dict:
          processRefCounts(xRefTable, o1)
       case Array:
           processRefCounts(xRefTable, o1)
```

```
/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
for _, e := range a {
         switch o1 := e.(type) {
          case IndirectRef:
             entry, ok := xRefTable.FindTableEntryForIndRef(&o1)
             {f if} ok \{
                 entry.RefCount++
          case Dict:
             processRefCounts(xRefTable, o1)
          case Array:
             processRefCounts(xRefTable, o1)
220 func processRefCounts(xRefTable *XRefTable, o Object) {
      switch o := o.(type) {
      case Dict:
         processDictRefCounts(xRefTable, o)
         processDictRefCounts(xRefTable, o.Dict)
      case Array:
          processArrayRefCounts(xRefTable, o)
 35 func dereferenceObjects(ctx *Context) error {
      log.Read.Println("dereferenceObjects: begin")
      xRefTable := ctx.XRefTable
      var keys []int
      for k := range xRefTable.Table {
          keys = append(keys, k)
      sort.Ints(keys)
      for _, objNr := range keys {
         err := dereferenceObject(ctx, objNr)
          if err ≠ nil {
             return err
      for _, objNr := range keys {
         entry := xRefTable.Table[objNr]
          if entry.Free || entry.Compressed {
              continue
          processRefCounts(xRefTable, entry.Object)
```

```
log.Read.Println("dereferenceObjects: end")
      return ni
   func identifyRootVersion(xRefTable *XRefTable) error {
      log.Read.Println("identifyRootVersion: begin")
      rootVersionStr, err := xRefTable.ParseRootVersion()
          return err
       if rootVersionStr = nil {
      rootVersion, err := PDFVersion(*rootVersionStr)
         return errors.Wrapf(err, "identifyRootVersion: unknown PDF Root version:
      xRefTable.RootVersion = &rootVersion
      if *xRefTable.HeaderVersion < V14 {</pre>
          log.Info.Printf("identifyRootVersion: PDF version is %s - will ignore root
             xRefTable.HeaderVersion, *rootVersionStr)
      log.Read.Println("identifyRootVersion: end")
      return nil
B06 func dereferenceXRefTable(ctx *Context, conf *Configuration) error {
      log.Read.Println("dereferenceXRefTable: begin")
      xRefTable := ctx.XRefTable
     err := checkForEncryption(ctx)
      if err \neq nil {
```

```
/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
         if err ≠ nil {
         err = dereferenceObjects(ctx)
              return err
         err = identifyRootVersion(xRefTable)
          log.Read.Println("dereferenceXRefTable: end")
         return nil
   345 <mark>func handleUnencryptedFile</mark>(ctx *Context) error {
         if ctx.Cmd = DECRYPT || ctx.Cmd = SETPERMISSIONS {
              return errors.New("pdfcpu: this file is not encrypted")
          if ctx.Cmd ≠ ENCRYPT {
              return nil
         if ctx.OwnerPW = "" {
            return errors.New("pdfcpu: please provide owner password and optional user
          return nil
  :364 func idBytes(ctx *Context) (id []byte, err error) {
         if ctx.ID = nil {
             return nil, errors.New("pdfcpu: missing ID entry")
         hl, ok := ctx.ID[0].(HexLiteral)
         if ok {
             id, err = hl.Bytes()
              if err ≠ nil {
localhost:49203
```

```
/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
       } else {
        sl, ok := ctx.ID[0].(StringLiteral)
             return nil, errors.New("pdfcpu: ID must contain hex literals or string
         id, err = Unescape(sl.Value())
             return nil, err
      return id, nil
return cmd = CHANGEOPW || cmd = CHANGEUPW || cmd = SETPERMISSIONS
395 func handlePermissions(ctx *Context) error {
      ok, err ≔ validatePermissions(ctx)
     if err ≠ nil {
      if !ok {
         return errors.New("pdfcpu: corrupted permissions after upw ok")
      if !hasNeededPermissions(ctx.Cmd, ctx.E) {
         return errors.New("pdfcpu: insufficient access permissions")
      return nil
15 func setupEncryptionKey(ctx *Context, d Dict) (err error) {
     ctx.E, err = supportedEncryption(ctx, d)
      if err ≠ nil
        return err
      ctx.E.ID, err = idBytes(ctx)
      if err ≠ nil {
         return err
      var ok bool
      ok, err = validateOwnerPassword(ctx)
```

```
/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go
             return err
        if !ok & needsOwnerAndUserPassword(ctx.Cmd) {
            return errors.New("pdfcpu: please provide the owner password with -opw")
         if ok & !needsOwnerAndUserPassword(ctx.Cmd) {
            ok, err = validatePermissions(ctx)
             if err ≠ nil {
                 return err
             if !ok {
                 return errors.New("pdfcpu: corrupted permissions after opw ok")
         ok, err = validateUserPassword(ctx)
         if err ≠ nil {
             return err
            return errors.New("pdfcpu: please provide the correct password")
         return handlePermissions(ctx)
    71 func checkForEncryption(ctx *Context) error {
         ir := ctx.Encrypt
            return handleUnencryptedFile(ctx)
          log.Read.Printf("Encryption: %v\n", ir)
        if ctx.Cmd = ENCRYPT {
             return errors.New("pdfcpu: this file is already encrypted")
localhost:49203
```

```
d, err := dereferencedDict(ctx, ir.ObjectNumber.Value())
   return err
log.Read.Printf("%s\n", d)
return setupEncryptionKey(ctx, d)
```

localhost:49203

/Users/horstrutter/go/src/github.com/pdfcpu/pdfcpu/pkg/pdfcpu/read.go