Palingol Grammar

rules

```
progr     :  optblk startblk befblk progblk aftblk endblk

optblk    : <empty>
          | optblk %pragma OPTNAM OPTVAL

startblk : <empty>
          |  %start { <empty> }
          |  %start { boolexp }

befblk    :  <empty>
          |  %before { <empty> }
          |  %before { boolexp }

progblk   :  %program { palblk spanblk crossblk }

aftblk    :  <empty>
          |  %after { <empty> }
          |  %after { boolexp }

endblk    :  <empty>
          |  %end { <empty> }
          |  %end { boolexp }

palblk    : <empty>
          | palblk palunit

spanblk   :  span(s) { spandesc }

crossblk  : <empty>
          | cross(es) { boolexp }

palunit   :  helix | pal(indrom) { boolexp }

spandesc  :  <empty>
          | spandesc spanunit

spanunit  : (SPANTYPE) vdigit

vdigit    : (+|-)DIGITS

palexp    : PALELT
          | PALELT #DIGITS

boolexp   :   VBOOL
          | ( VBOOL )
          | ( BOOLPHYSICAL palexp )
          | ( BOOLMONADIC boolexp )
          | ( BOOLDYADIC boolexp boolexp )
          | ( NUMCOMPMONADIC numdft )
          | ( NUMCOMPDYADIC numdft numdft )
          | ( STRCOMPMONADIC strdft )
          | ( STRCOMPDYADIC strdft strdft )
          | ( PATDYADIC strdft numdft )
          | ( PATTRYADIC strdft numdft boolexp )
          | ( BOOLPAT strdft strdft numdft numdft boolexp )
          | ( SETMONADIC varnamgenexp )
          | ( VARCOMP varnam)
          | ( PRINTDYADIC VSTRING varnam)
          | ( PRINTDYADIC VSTRING genexp )
```

```
           | ( IFMONADIC ( boolexp ) boolexp )
           | ( IFMONADIC ( boolexp ) boolexp ELSEMONADIC boolexp )
           | ( WHILEMONADIC ( boolexp ) boolexp )
           | ( PRAGMA OPTNAM )
           | ( PRAGMA OPTNAM boolexp )
           | varnam
           | boolexp boolexp

varnam     : $LETTERS(DIGITS)


numexp     : vdigit
           | ( NUMPHYSICAL palexp )
           | ( NUMMONADIC numdft )
           | ( NUMDYADIC numdft numdft )
           | ( NUMSTRMONADIC strdft )
           | ( NUMSTRDYADIC strdft strdft )
           | ( NUMPAT )
           |   NUMPAT
           |   varnam

numdft     :    numexp
           |    palexp

strexp     :  VSTRING
           | ( STRPHYSICAL palexp )
           | ( STRMONADIC strdft )
           | ( STRDYADIC strdft strdft )
           | ( STRTRYADIC strdft numdft numdft )
           | ( STRPAT )
           |   STRPAT
           | varnam

strdft     : strexp
           | palexp

genexp     : boolexp
           | numdft
           | strdft
```

tokens

```
DIGITS              : [0-9]+

VSTRING             : "[^\"\n]*"
                    | '[^\"\n]*'

LETTERS             : [a-zA-Z]+


OPTNAM              : debug
                    | prune
                    | volatil

OPTVAL              : on
                    |  off

SPANTYPE            : %s(tart)_h(ead) | %e(nd)_h(ead)
                    | %s(tart)_t(ail) | %e(nd)_t(ail)

PALELT              : head | tail | loop
                    | fullseq(uence) | seqnam(e)  | seqinf(o)
```

```
VBOOL               : true | false

BOOLPHYSICAL        : direct(strand)

BOOLMONADIC         : not

BOOLDYADIC          : and | or | equiv(alent)

NUMCOMPMONADIC      : zero

NUMCOMPDYADIC       : eq | ne | gt | ge | lt | le

STRCOMPMONADIC      : str(ing)empty

STRCOMPDYADIC       : str(ing)beg(ins)
                    | str(ing)end(s)
                    | str(ing)c(o)mp(are)

PATDYADIC           : pat(tern)match

PATTRYADIC          : pat(tern)compil(e)

BOOLPAT             : pat(tern)search

SETMONADIC          : set

VARCOMP             : exist(s)

PRINTDYADIC         : print(f)

IFMONADIC           : if

WHILEMONADIC        : while

PRAGMA              : pragma


NUMPHYSICAL         : start | end
                    | length | energ(y)

NUMMONADIC          : opp(osite) | sq(uare)

NUMDYADIC           : add | sub(s)(tract)
                    | mul(tiply) | div(ide) | mod(ulo)

NUMSTRMONADIC       : str(ing)len

NUMSTRDYADIC        : scoremat(rix)

NUMPAT              : matchpos(ition)
                    | matcherr(or)(s)

STRPHYSICAL         : seq(uence)

STRMONADIC          : complem(ent)
                    | inver(se)
                    | complinv(erse)

STRDYADIC           : str(ing)cat

STRTRYADIC          : s(ub)str(ing)
```

STRPAT : **matchfrag**(**ment**)