NAME

basictool − tokenise, de-tokenise, pack, unpack and analyse BBC BASIC programs

SYNOPSIS

basictool [*OPTION*]... INFILE [*OUTFILE*]

DESCRIPTION

basictool converts BBC BASIC programs between ASCII text and the tokenised form used by (6502) BBC BASIC. It can also pack programs (making them shorter but less readable), unpack them (to partially reverse the effects of packing) and generate variable and line number references. Behind the scenes, it is really a specialised BBC Micro emulator which uses the BBC BASIC and Advanced BASIC Editor ROMs to operate on BBC BASIC programs.

basictool expects at least one argument, the name of the input file. The input may be either tokenised or text BBC BASIC; **basictool** will auto-detect the type of input, although this can be overridden using *−−input−tokenised* if necessary. **basictool** does not care about file extensions; auto-detection is based on the file contents. To read from standard input, specify the input file as − . A second argument, the name of the output file, may be given; standard output will be used by default.

Text input to **basictool** can use the standard BBC BASIC abbreviations (such as "P." for "PRINT"). Line numbers in text input are optional; lines with no number will be automatically assigned the next line number in sequence.

OPTIONS

−h, **−−help**
: Show a description of **basictool** and list the available options.

−−roms
: Show information about the BBC BASIC and Advanced BASIC Editor ROMs built into **basictool**.

−v, **−−verbose**
: Increase the verbosity of **basictool**'s output; this can be specified twice for extra-verbose output.

−−show−all−output
: Show all output from the hidden emulated BBC Micro. **basictool** normally processes this output internally but does not show it to the user. This is mostly useful for debugging or to see what's going on behind the scenes.

−2, **−−basic−2**
: Use the BASIC 2 ROM. This is mainly useful to avoid BASIC 4's behaviour of always stripping spaces at the end of lines; see *−−strip−spaces−end* below for more on this.

−4, **−−basic−4**
: Use the BASIC 4 ROM; this is the default anyway, so this option just makes that explicit.

−−input-tokenised
: Assume that the input file is tokenised BASIC instead of trying to auto-detect whether it is text or tokenised BASIC. This should not normally be necessary. There is no way to explicitly force the input to be treated as text BASIC and it shouldn't be necessary to do this.

−s, **−−strip−spaces**
: Shorthand for *−−strip−spaces−start −−strip−spaces−end*.

−−strip−spaces−start
: By default **basictool** will preserve spaces at the start of lines in text input; this option causes them to be stripped.

−−strip−spaces−end
: By default **basictool** will try to preserve spaces at the end of lines in text input; this option causes them to be stripped. BASIC 4 always strips spaces at the end of lines whether this option is specified or not; a warning will be given if this happens and this option is not specified. Use *−−basic−2* if it is important to preserve spaces at the end of lines.

−p, **−−pack**
Pack the program using the Advanced BASIC Editor's pack utility; this will reduce its size and improve its performance, at the cost of significantly reducing its readability. By default the pack options which give the largest possible size reduction are used; the following options allow individual pack options to be disabled if they are inappropriate.

−−pack−rems−n
Disable removal of REM statements (BASIC comments) when packing. This option implies *−−pack*.

−−pack−spaces−n
Disable removal of unnecessary spaces when packing. This option implies *−−pack*.

−−pack−comments−n
Disable removal of assembly language comments when packing. This option implies *−−pack*.

−−pack−variables−n
Disable shortening of variable, procedure and function names when packing. This option implies *−−pack*.

−−pack−singles−n
Disable changing the first character of variable, procedure and function names when packing. This option implies *−−pack*.

−−pack−concatenate−n
Disable combining separate lines of code together when packing. This option implies *−−pack*.

−r, **−−renumber**
Renumber the program, by default starting with line 10 and incrementing the line number in steps of 10 for subsequent lines; the following options allow these defaults to be overridden. Simple line number references are fixed up during renumbering, but programs using computed line numbers are likely to be broken by renumbering.

−−renumber−start=*N*
Renumber the program using N as the first line number. This option implies *−−renumber*.

−−renumber−step=*N*
Renumber the program using gaps of N between line numbers. This option implies *−−renumber*.

−l, **−−listo**=*N*
Use BASIC's LISTO option N to control the formatting of text output. This option is only relevant when using the *−−ascii* output type option.

The following options control the type of output produced by **basictool**. Only one of these options can be given; if no output type option is given, *−−ascii* will be used.

−a, **−−ascii**
Output the program as ASCII text suitable for viewing or editing with a text editor.

−t, **−−tokenise**
Output the program as tokenised BASIC suitable for LOADing or CHAINing into (6502) BBC BASIC.

−f, **−−format**
Output the program as formatted text using the Advanced BASIC Editor's format utility. This splits up multi-statement lines without introducing extra line numbers, which may be more readable but means the output cannot be re-used as input to **basictool**.

−u, **−−unpack**
Output the program as unpacked text using the Advanced BASIC Editor's unpack utility. This will partially undo the effects of *−−pack* but of course cannot restore comments or variable, procedure and function names. The output is human-readable and should also be suitable for use as input to **basictool**.

Unpacking splits apart multi-statement lines and will fail if the line numbers in the input program do not have sufficient gaps to allocate numbers for the new lines; if this happens, try using *--renumber-step* in combination with *--unpack*.

--line-ref
Output a table of line number references in the program using the Advanced BASIC Editor's "Table Line References" utility.

--variable-xref
Output a table of variable, procedure and function names used in the program and the line numbers they are used on using the Advanced BASIC Editor's "Variables Cross Reference Tables" utility.

EXIT STATUS

basictool will exit with a zero exit status if no errors occur; errors are indicated by a non-zero exit status.

EXAMPLES

Tokenise a text BASIC program so it can be loaded by BBC BASIC.

```
$ basictool -t prog.txt prog.tok
```

Display a tokenised BASIC program as human-readable text.

```
$ basictool prog.tok
```

Tokenise and pack a text BASIC program so it can be loaded by BBC BASIC and use less memory.

```
$ basictool -t -p prog.txt prog.tok
```

AUTHORS

basictool was written by Steven Flintham. It builds on several other existing pieces of code, most notably the BBC BASIC and Advanced BASIC Editor ROMs, which actually do most of the work, and lib6502, which allows the 6502 code in those ROMs to be executed.

BBC BASIC was originally published by Acorn.

The BASIC editor and utilities were originally published separately by Altra. The Advanced BASIC Editor ROMs used here are (C) Baildon Electronics. Thanks to Dave Hitchins for his support for developing **basictool** using these ROMs.

lib6502 was originally written by Ian Piumarta, although **basictool** uses the version from PiTubeClient (https://github.com/hoglet67/PiTubeClient).

Cross-platform command line parsing is performed using cargs (https://github.com/likle/cargs).