NAME

od — dump files in numeric and readable formats

SYNOPSIS

od [**−v**] [**−j** *skip*] [**−N** *max*] [**−w**[*width*]] [**−A n**|**x**|**d**|**o**] [**−t** *type*...]...
[**−−endian**=**little**|**big**] [*file*]...

Where *type* is a string of
{**a**|**c**} [**z**[**Z**]]
{**x**|**u**|**d**|**o**}[**1**|**2**|**4**|**8**|**C**|**S**|**I**|**L**][**z**[**Z**]]
f [**4**|**8**|**16** |**F**|**D**|**L**] [**z**[**Z**]]

DESCRIPTION

Discards *skip* bytes of *file*s (standard input stream if "**-**", the default), then formats *max* (unlimited) bytes, *width* (**16**) per line, with an address, interpreted as *type*s.

type	**Type**	**Bytes**	**Notes**
a	ASCII	**1**	Strip top bit, format non-printable bytes and space as in ascii(7) (in lower-case), others verbatim.
c	Character	≥ **1**	Non-printables and invalid sequences written as octal bytes. Control characters written as C escapes. Multi-byte characters are written at their start, remaining bytes are "**". Not affected by block boundaries.
x	Hexadecimal integer	**1**, **2**, **4**, **8**	Respectively: **C** (*char*), **S** (*short*), **I** (*int*), **L** (*long*). **xo** zero-padded, **ud** space-padded. Defaults to **I** (**4**).
u	Unsigned decimal integer		
d	Signed decimal integer		
o	Octal integer		
f	Floating-point number	**4**, **8**, **16**	Respectively: **F** (*float*), **D** (*double*), **L** (*long double*), to the recovery precision. Defaults to **D** (**8**).

These may be suffixed with a **z** to provide a dump of printable bytes on the right margin, with non-printables replaced with a '**.**', or with a **zZ** to do the same for characters. Many *type*s may be pasted together or passed to multiple **−t**s.

The input is taken to be as-if **cat** [*file*].... If this doesn't evenly divide a *type*, then it's filled out with zero bytes. With multiple *type*s, output is right-aligned to their respective boundaries. Multi-byte *type*s are cast directly into this system's their native representation (least significant byte first).

Data for each line is preceded by an address, governed by **−A**:
- **n** (empty)
- **x** hexadecimal
- **d** decimal
- **o** octal

then a space. With multiple *type*s, only the first one gets an address — the rest's is padded with spaces. After *file*s are exhausted, the final line gives the would-be address of the next byte (and, thus, the total input size), unless **−An**.

skip, *max*, and *width* are in the mostly-case-insensitive format:
[*base*][**b**|**KMGTPEZY**[**B**]] (with at least one of {*base*, **b**, **KMGTPEZY**, **B**})
Where *base* is an optionally-floating-point number of bytes, defaulting to **1**, which is then optionally multiplied by the relevant unit. **B** sets the unit multiplier to **1000** (from **1024**). **b**(lock) is a unit of **512**.
skip|*max*|*width* is equal to $base \cdot unit^{mult}$, if any, or *base*.

OPTIONS

−v, **−−output−duplicates**	By default, consecutive identical lines are replaced with a line of just "*"; write them all, instead.
−j, **−−skip−bytes**=*skip*	Seek over (or read and discard) *skip* bytes of the input.
−N, **−−read−bytes**=*max*	Consume at most *max* bytes.
−w, **−−width**	**−w***32*
−w*width*, **−−width**=*width*	Each output line contains *width* bytes. Rounded down to a multiple of the widest *type*.
−A, **−−address−radix**=**n**|**x**|**d**|**o**	See above. Defaults to **o**.
−t, **−−format**=*type*...	Cast input data as *type*s. See above. Defaults to "**oS**" (**o2**).
−−endian=little	No effect.
−−endian=big	Reverse each value's bytes before casting. The values are prefix-matched (**−−endian**=*b* is equivalent to **−−endian=big**, &c.).

EXIT STATUS

1 if a *file* couldn't be opened or read or *skip* could not be satisfied.

EXAMPLES

```
$ seq 10 | od
0000000 005061 005062 005063 005064 005065 005066 005067 005070
0000020 005071 030061 000012
0000025
$ seq 10 | od -t x1z -Ax -j 0x2 -N 022                  # like hexdump(1)
000002 32 0a 33 0a 34 0a 35 0a 36 0a 37 0a 38 0a 39 0a  >2.3.4.5.6.7.8.9.<
000012 31 30                                            >10<
000014
$ seq 10 | od -t azf -An
  1 nl  2 nl  3 nl  4 nl  5 nl  6 nl  7 nl  8 nl  >1.2.3.4.5.6.7.8.<
 1.6292135911574872e-259 1.9544134620527668e-259
  9 nl  1  0 nl                                   >9.10.<
      2.16194199887e-313
$ echo Żupan z 雨港 | od -t cz -t czZ -t d2 -Ad -N 14
0000000  Ż  **  u   p  a   n      z     雨  **  ** 346 270  >..upan z .....<
         Ż  **  u   p  a   n      z     雨  **  ** 346 270  >Ż*upan z 雨*..<
        -17467  28789  28257  31264  -5856  -22373  -18202
0000014
```

SEE ALSO

hexdump(1), ascii(7)

STANDARDS

Conforms to IEEE Std 1003.1-2024 ("POSIX.1"). The default address base and *width* are unspecified, but all implementations agree on **o** and **16**. *skip* and *max* are guaranteed to take **0x**/**0X** and **0** prefixes for base-16 and base-8; only *skip* is required to accept suffixes, and only **k**, **m**, and **b**. **1**-, **2**-, **4**-, and **8**-byte **xudo**s are required to be present, even if no native sizes or alphabetic names for them exist. The sizes for **f** are accurate for IEEE Std 754-1985 systems — alphabetic names are guaranteed to exist to map to their respective C types; numeric names must correspond to those. **−A n** is allowed to still produce a final, empty, line where the file size *would* have been.

There are obsolete XSI-shaded usages to watch out for:

od [**−xdsbo**] ... [*file*] ... **+**[[**0**]**x**]*skip*[**.**][**b**][**B**]
od [**−xdsbo**] ... *file* [**+**|**0x**]*skip*[**.**][**b**][**B**] (|starts with a digit)

these are not part of this implementation, but are prevalent on others. XSI **−xdsbo** can be translated to the standard format as such:

−c Like **−tc**, but with LC_CTYPE=**C** and not "**\a**" or "**\v**".

−x = **−t** x2 **−d** = **−t** u2 **−s** = **−t** d2 **−b** = **−t** o1 **−o** = **−t** o2

These contradict values on some historical systems. Strictly, the POSIX usage requirements contradict *all* historical systems (and the synopses above are more representative). Avoid them, guard against them by always specifying at least one of the standard flags (**−Ao** is well-suited).

The **z** *type* suffix, **−w**, and **−−endian** are extensions, originating from the GNU system. Its *width* is a plain number and always reset to the smallest — instead of the closest acceptable — value. The **zZ** suffix is an extension.

The **c** mode formats NUL (**0**) as "**\0**", the bell (**0x7**) as "**\a**", the backspace (**0x8**) as "**\b**", the form-feed (**0xC**) as "**\f**", the new-line (**0xA**) as "**\n**", the carriage return (**0xD**) as "**\r**", the tab (**0x9**) as "**\t**", and the vertical tab (**0xB**) as "**\v**". The **a** mode is allowed to format the new-line (**0xA**) as either "**lf**" (the proper "LINE FEED"-derived short-hand) or "**nl**" — all implementations except the illumos gate use the latter for compatibility with 4.2BSD.

BUGS

Strictly, same-line folding ought to apply to output lines, not input blocks — there are pathological scenarios where this could affect **c** and **zZ** output.

HISTORY

Research UNIX

Appears in the first edition of the UNIX Programmer's Manual as od (I):

```
NAME        od  --  octal dump
SYNOPSIS    od name [ origin ]
SEE ALSO    db
```

indicating that it "dumps a file in octal, eight words per line with the origin of the line on the left." – as present-day, including the would-be-next address at the end, except that the output words ("**o2**"-equivalent) are faux-signed with the top bit used as the sign bit and the remaining fifteen bits formatted verbatim. The **BUGS** confirm that each "block" is 512 bytes and written in its entirety, rounded up with "garbage" (old data). In many ways the **BUGS** down-play the role of this, saying that this only happens at end of the file, but it happens every time read(2) returns short.

origin — octal and rounded down to the closest multiple of the line size (**16**) — functions like *skip*, and discards (parts of the) read blocks.

Version 2 AT&T UNIX elaborates on the **db** recommendation thusly:

```
Since od does not seek, but reads to the desired starting point,
od (rather than db) should be used to dump special files.
```

with a mirrored stanza — though citing byte-wise reading — in db (I). Indeed, the debugger is both likely to be more familiar to most users and has many more output formats: **/** like "**o2**", \ like "**o1**", **"** like "**c**" (with a backslash prepended for non-printable octal formats) but for two bytes at the cursor, **'** likewise but for one, **`** like "**o2**" but multiplied by **2** (quoting B programs), and **?** to fully disassemble at the cursor, and is thus much more useful for what one'd use **od** for today.

The **BUGS** are removed.

Version 3 AT&T UNIX sees

```
SYNOPSIS    od [ -abcdho ] [ file ] [ [+]offset[.][b] ]
```

which is more accurately transcribed as

od [**−abcdho**] *file* [[**+**]*offset*[**.**][**b**]]
od [**−abcdho**] [**+***offset*[**.**][**b**]]

(**+** only required if no *file*), matching the present-day XSI semantics — *offset* is octal by default, but **.** makes it decimal; the **b** multiplies it by **512** (also, '**8**' and '**9**' are allowed in octal mode the base of the *offset* is used as the base the address; this is like **−jA** were welded together).

file defaults to the standard input stream if not specified; **−dob** are as present-day XSI, **−x** is equivalent to present-day **−t x2** (**−x**), **−c** is like **−c** except only "**\0**", "**\t**", and "**\n**" are recognised and all other non-printables are written as "**\?**", and **−a** disassembles (but just the opcodes), with unknown formatted as "**???**". Multiple formats are allowed, but are always written in the order **−odahcb**. They do a pretty good job of being aligned, and not over-aligned.

The no-seek stanza is removed, and *`offset`* first *sets* the file position to *`offset`*/**512** (with no error checking), then eats the remainder. The position is untouched without *`offset`*.

An inscrutable deduplication scheme appears at some point in [Version 3 AT&T UNIX, Version 5 AT&T UNIX], discarding consecutive identical *words*. Note how these files differ only by one or two bytes (and never-mind the **`printf`** anacrhonism):

```
$ printf '%09999d' | od -c +0.
0000000  0  0
0009984  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 \0
0009999
$ printf 'a%09999d' | od -c +0.
0000000  a  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0010000
$ printf 'aa%09999d' | od -c +0.
0000000  a  a  0  0  0  0  0  0  0  0  0  0  0  0  0  0
0000016  0  0
0010000  0 \0
0010001
```

Note also how even in byte modes, input is still padded to two-byte boundaries. This is undocumented.

Version 7 AT&T UNIX sees a **SYNOPSIS** of

od [**−bcdox**] [file] [[**+**]offset[**.**][**b**]]

`-x` was renamed from **`-h`** and matches present-day XSI (**`-h`** is still accepted). **`-c`** is as present-day XSI. With multiple types, addresses are continued with a tab instead of eight spaces.

The word-squeezing scheme is replaced with the present-day line-based one.

Undocumentedly, *`offset`* may start with a **`x`** or **`0x`** for lower-case hexadecimal (this, again, propagates as-if **`-Ax`**; thankfully hexadecimal characters are only allowed in hexadecimal mode), this is still overridden by `.`. **`B`** is accepted as well as **`b`**. To this end, the **SYNOPSIS** may be better-served as

od [**−bcdoshx**] *`file`* [[**+**][[**0**]**x**]*`offset`*[**.**][**b**|**B**]]
od [**−bcdoshx**] [**+**[[**0**]**x**]*`offset`*[**.**][**b**|**B**]]

The final offset is arrived at with a single `fseek`(3) call (error again unchecked, but at least it's consistent now).

The BSD

3BSD naturally sees Version 7 AT&T UNIX **`od`**, but re-adds **`-a`** to the **SYNOPSIS** (and only there).

4.2BSD upgrades the **NAME** to "od − octal, decimal, hex, ascii dump", which is still somehow not enough, and sees a **SYNOPSIS** of

od [−format] [file] [[+]offset[**.**][**b**] [label]]

a usage string of

```
usage: od [-abcdfhilopswvx] [file] [[+]offset[.][b] [label]]
```

and a header comment with

```
usage:  od [-abBcdDefFhHiIlLopPs[n]vw[n]xX] [file]
           [[+]offset[.][b] [label]]
```

which may be better served as

od [−{**a**[**p**|**P**]**bBoOcdDefFhxHXiIlLs**[*`min`*...]**vw**[*`width`*...]}...] *`file`*
[[**+**][[**0**]**x**]*`offset`*[**.**][**b**][**B**] [[**+**][[**0**]**x**]*`label`*[**.**][**b**][**B**]]]
od [−{**a**[**p**|**P**]**bBoOcdDefFhxHXiIlLs**[*`min`*...]**vw**[*`width`*...]}...] [**+**[[**0**]**x**]*`offset`*[**.**][**b**][**B**]
[[**+**][[**0**]**x**]*`label`*[**.**][**b**][**B**]]]

(except **`pP`** may be at any point in the flag). If *`label`*, then the address base is that of *`label`* instead of *`offset`* and an additional address is written, following the normal one, but starting at *`label`* instead of *`offset`* (**0**); this may correspond to the single line of **`od -ap +`**_`100 x20`_:

```
0000040 (0000020)    k dc3   Z   4  bs   M dc3 etx   a   ?  si   e
```

`b` multiplies by **512** *and* **`B`** multiplies by **1024**. The address(ses) are followed by two spaces, not one.

If *offset*, *file* has links, and is not a teletype, it's sought "in multiples of a physical block" (**512**) "in case we're accessing a raw disk", then read, otherwise just read. The no-links check is commented "/*!pipe*/", but it'll also (falsely) succeed if *file* is simply removed after being opened, but (correctly) succeed for sockets. The file position is still set instead of advanced, but *if* reading to skip, premature end-of-file is met with an error, similar to present-day. Thus, **+xfffffe00** on a seekable *file* is all-but-guaranteed to produce "fffffe00" as the sole output but **+xfffffe01** – to error.

−a is like present-day "**a**"; additionally, if **−p** is also specified, the cells for bytes with even parity (even number of set bits before stripping) are underlined by formatting them as underscore-backspace-character, if **−P** – odd parity, whichever's last.

−w may be succeeded by a decimal string and defaults to **32** if it isn't or it's just zeroes – much like present-day **−w** – but is always accepted, and types that aren't evenly divided are zero-extended like at the end of input. **−v** is invented as present-day. Single-byte formats (**abc**) no longer have a phantom zero byte at the end of odd-length inputs.

−HX are both equivalent to present-day "**x4**", **−D** – "**u4**", **−IlL** – "**d4**", and **−O** – "**o4**"; **−B** is an alias for **−o**; **−i** is equivalent to present-day **−s** (and the formatter name suggests that it was simply renamed from Version 7 AT&T UNIX **−s**, and quite late). **−f** is equivalent to present-day "**fF**", and both of **−eF** – "**fD**", though **−f** uses a fixed "**%.7e**" format and **−F** – "**%.14e**" (fixed-precision scientific notation). On the VAX, which uses some cursed non-IEEE Std 754-1985 float implementation and would otherwise SIGILL in this case, if the second-least significant byte of the first four bytes is **0x80**, they're instead formatted as-if **x4**.

−s implies **−v** and is a strings(1)-like mode, finding strings of at least *min* (default **3**) consecutive "ascii graphic" (actually printable + the non-NUL ones from **−c**) bytes, followed by a NUL (**0**), and outputting them each on their own line, unsuppressably preceded by the address(es) of their first byte.

min and *width* greater than **1024** overrun the buffer, Up to *31* types may be specified (more overrun the buffer), and their order is respected.

The manual specs a scarce few of the *format*s: **a**[**p**|**P**] (displaying "with their ACSII names"), **bcdfhilox**, **v** (though you wouldn't call it a format), **s**[*min*...], and **w**[*width*...]. This leaves **BDeFHXIL** undocumented, which represent three unique formats (**DFX**). Uncharacteristically, the *offset* and *label* are fully-documented (except that the "radix" (base) of *label* overrides that of *offset*, and *offset*-past-end behaviour).

The **BUGS** are plenty: "It is an historical botch to require specification of object, radix, and sign representation in a single character argument." — foreshadowing the genericised **−t** — "A hexadecimal offset can't be a block count." — strictly false: it can't be a **512**-byte block count, since the **b** is consumed as the number; it can be a **1024**-byte block count since the *offset* and *label* only parse lower-case hexadecimal — &c.

4.3BSD–Reno sees a **SYNOPSIS** of

od [**−aBbcDdeFfHhIiLlOoPpvXx**]

(any amount of files may be specified to be concatenated, as present-day) and implements **od** as a basename-selected parser effectively injecting **−e** arguments to hexdump(1) (except, of course, **−s**, whose error points to strings(1) — which, notably, had the same base-line capability even in 4.2BSD — **−Pp** which are in the **SYNOPSIS** by accident and yield an error, and **−w**, which just yields an error). **BUGS** are re-assessed as "Quite a few.". The old implementation is distributed in source form in /usr/src/old until 4.4BSD–Lite.

4.4BSD sees a **SYNOPSIS** of

od [**−aBbcDdeFfHhIiLlOovXx**] [[**+**]offset[**.**][**Bb**]]*file*

which really ought to be

od [**−aBbcDdeFfHhIiLlOovXx**] [*file*] [[**+**][**0**[**x**]]*offset*[**.**][**Bb**]] (if not **+**, then *offset* must start with a digit or '**x**' and a hexadecimal digit)

od [**−aBbcDdeFfHhIiLlOovXx**] [[**+**][**0**[**x**]]*offset*[**.**][**Bb**]]

if the *offset* argument isn't consumed, all arguments are considered to be files; if it is, the argument list terminates just before it. *offset* is parsed with strtol(3), so it accepts both letter cases for hexadecimal arguments, and — in a self-fulfilling prophecy — consumes both **Bb**s, which are now exclusive.

Invalid digits are now silently ignored and the argument is unconsumed. **x** requires that the second character is a hexadecimal digit; **0x** doesn't. **.** excludes, instead of overrding, [**0**]**x**. Garbage at either end causes the argument to be left unconsumed.

System V

AT&T System III UNIX sees v7 **od** except it exits **2** if *file* failed to open.

AT&T System V Release 1 UNIX adds **−s**, as present-day (ordered as **−odas**[**hx**]**cb** on output). **b**|**B** multiplies by BSIZE (which depends on the filesystem configured when building — **512** for the "original" (and when built with dual-filesystem support) and **1024** for the new one), instead of **512** (the manual, naturally, does not reflect this).

AT&T System V Release 3 UNIX uses **512** again.

AT&T System V Release 4 UNIX shoe-horns the 4.2BSD format-order code into a minimally-changed implementation with a **SYNOPSIS** of

od [**−bcDdFfOoSsvXx**] [*file*] [[**+**]*offset*[**.** | **b**]]

with **−v** as present-day, **−fF** equivalent to 4.2BSD (sans the VAX bullshit), and **−XDSO** equivalent to their lower-case present-day XSI variants but for four-byte integers (similar, again, to 4.2BSD but without the **s**→**i** rename).

Standards

X/Open Portability Guide Issue 2 (“XPG2”) includes AT&T System V Release 3 UNIX od(1) — shaded OF ("Output format incompletely specified" – it isn't at all) — with editorial differences and **−s** shaded PI ("The behaviour cannot be guaranteed to be consistent"), no doubt in reference to 4.2BSD's.

IEEE Std 1003.2-1992 (“POSIX.2”) invents the modern

od [-v] [-A *address_base*] [-j *skip*] [-n *count*] [-t *type_string*] ... [*file* ...]

Synopsis as "od — Dump files in various formats" and leaves the behaviour unspecified if no flags (except **−v**) were passed and the first argument starts with a digit or a ‘**+**’.

4.3BSD–Reno's hexdump(1) cites "POSIX 1003.2" compatibility (and bases purported **od** obsolescence thereon). The **POSIX.2 Change History** indicates that in Draft 10, "hexdump" was "renamed" (entirely re-invented) to "od". This is further lamp-shaded in the **History of Decisions Made** for **od**: "The hexdump description was much more complex than needed for a simple dump utility.".

X/Open Portability Guide Issue 4 (“XPG4”) adds the AT&T System V Release 3 UNIX usage, shaded EX (equivalent to present-day XSI), on top of the IEEE Std 1003.2-1992 (“POSIX.2”) implementation. **−s** is still also shaded PI, even though it's defined as "equivalent to **−t d2**". *offset* is additionally only recognised if there are up to two arguments. **FUTURE DIRECTIONS** warn that all EX-shaded features may be withdrawn in the future. Oh, what a world this would be!

The Single UNIX Specification (“SUS”) defines **−c** to "interpret bytes as characters specified by the current setting of the LC_CTYPE category" (instead of the POSIX locale, which is as-if ASCII) — which doesn't really match what implementations do at all (AT&T System V Release 4 UNIX says it does so, but that's not at all true and depending on the locale it takes a longer or shorter path to format bytes with the high bit set as three octal digits) — and that it's "equivalent to **−t c**", which it isn't, not only due to the aforementioned, but also because it doesn't catch "**\a**" and "**\v**".

Version 2 of the Single UNIX Specification (“SUSv2”) removes the mistaken **−c** ↔ **−t c** equivalency stanza.

Version 3 of the Single UNIX Specification (“SUSv3”) unshades **−s** PI and changes *offset* recognition to present-day: last argument starts with a **+** or two arguments and the second starts with a "numeric" byte.

IEEE Std 1003.1-2008 (“POSIX.1”) allows the "**-**"-as-standard-input-stream behaviour and also suppresses *offset* handling if **−v** was passed, as present-day.