

tr Cheatsheet

By Dejan Panovski • Updated on Oct 28, 2026

tr command reference for translating, deleting, and squeezing characters, with character classes, escape sequences, case conversion, and pipeline recipes.

The `tr` command changes text one character at a time. It reads only from standard input, so you pipe text into it or redirect a file. Every example below behaves the same with GNU `tr` and with the Rust-based `tr` that ships with Ubuntu 25.10 and later.

Basic Syntax

Input comes from a pipe or a redirect, never a file argument.

<code>tr SET1 SET2</code>	Replace each character in SET1 with the matching one in SET2
<code>tr -d SET1</code>	Delete characters in SET1
<code>tr -s SET1</code>	Squeeze repeated characters in SET1 to one
<code>tr -c SET1 SET2</code>	Act on characters NOT in SET1 (also <code>-C</code> , <code>--complement</code>)
<code>tr -t SET1 SET2</code>	Truncate SET1 to the length of SET2 first
<code>tr -ds SET1 SET2</code>	Delete SET1, then squeeze SET2
<code>command tr SET1 SET2</code>	Translate the output of another command
<code>tr SET1 SET2 < in.txt > out.txt</code>	Read a file and write the result to a new one

Translate Characters

Characters map by position; the last character of SET2 repeats if it is shorter.

<code>echo 'linuxize' tr 'lin' 'red'</code>	Map <code>l</code> to <code>r</code> , <code>i</code> to <code>e</code> , <code>n</code> to <code>d</code> ; prints <code>reduxeze</code>
<code>tr ' ' '_'</code>	Replace spaces with underscores
<code>tr ':' '\n'</code>	Replace colons with newlines
<code>tr '{}' '()'</code>	Replace curly braces with parentheses
<code>tr 'a-f' 'A-F'</code>	Ranges map in order
<code>echo 'abcde' tr 'abcde' '12'</code>	Short SET2 reuses its last character; prints <code>12222</code>
<code>echo 'abcde' tr -t 'abcde' '12'</code>	Only <code>a</code> and <code>b</code> change; prints <code>12cde</code>

Change Case

Convert, swap, or rotate letter case.

<code>tr '[:lower:]' '[:upper:]'</code>	Lowercase to uppercase
<code>tr '[:upper:]' '[:lower:]'</code>	Uppercase to lowercase
<code>tr 'a-z' 'A-Z'</code>	Convert ASCII lowercase letters to uppercase
<code>tr 'a-zA-Z' 'A-Za-z'</code>	Swap case; Hello becomes hELLO
<code>tr 'A-Za-z' 'N-ZA-Mn-za-m'</code>	ROT13; run it twice to get the original text

Delete Characters

Remove every occurrence of the characters in SET1.

<code>tr -d 'aeiou'</code>	Delete the listed characters
<code>tr -d '[:digit:]'</code>	Delete all digits
<code>tr -d '[:punct:]'</code>	Delete punctuation
<code>tr -d '\r' < win.txt > unix.txt</code>	Convert Windows line endings to Unix
<code>tr -d '\n'</code>	Join all lines into one
<code>tr -d '[:blank:]'</code>	Delete spaces and tabs, keep newlines
<code>tr -d '[:space:]'</code>	Delete all whitespace, including newlines

Keep Only (Complement)

`-c` flips SET1. Add `\n` to SET1 to keep line breaks.

<code>echo 'tel 123-456' tr -cd '[:digit:]'</code>	Keep only digits; prints 123456 with no newline
<code>tr -cd '[:alnum:]\n'</code>	Keep letters, digits, and newlines
<code>tr -cd '[:print:]\n'</code>	Strip control characters and other non-printable bytes
<code>tr -c 'a-z\n' '_'</code>	Replace everything except a-z and newlines with _
<code>tr -cs '[:alnum:]' '\n'</code>	Print one word per line

Squeeze Repeats

`-s` collapses a run of the same character into one.

<code>tr -s ' '</code>	Collapse repeated spaces
<code>tr -s '\n'</code>	Remove blank lines between lines of text
<code>tr -s ' '\n'</code>	Squeeze spaces and newlines in one pass
<code>tr -s ' ' '_'</code>	Translate spaces to _, then squeeze the _ runs
<code>echo 'hello' tr -s 'a-z'</code>	Squeeze any repeated lowercase letter; prints he1o

Character Classes

Quote character classes. When translating, SET2 accepts only `[ :lower: ]` or `[ :upper: ]` , paired with the opposite class at the same position in SET1. With `-ds` , SET2 accepts any character class.

[:alnum:]	Letters and digits
[:alpha:]	Letters
[:digit:]	Digits 0–9
[:lower:]	Lowercase letters
[:upper:]	Uppercase letters
[:space:]	Space, tab, newline, carriage return, vertical tab, form feed
[:blank:]	Space and tab only
[:punct:]	Punctuation
[:print:]	Printable characters, including space
[:graph:]	Printable characters, excluding space
[:cntrl:]	Control characters
[:xdigit:]	Hex digits 0–9, A–F, a–f

Escapes and Repeats

Special characters and fill patterns inside a set.

<code>\n</code>	Newline
<code>\t</code>	Tab
<code>\r</code>	Carriage return
<code>\\</code>	Backslash
<code>\0</code>	NUL byte
<code>\NNN</code>	Byte with octal value NNN (1 to 3 digits)
<code>[c*]</code>	In SET2, repeat c until SET2 is as long as SET1
<code>[c*n]</code>	In SET2, n copies of c

Common Pipelines

Practical combinations with other commands.

<code>printf '%s\n' "\$PATH" tr ':' '\n'</code>	One <code>PATH</code> directory per line
<code>tr '\0' '\n' < /proc/\$\$/environ</code>	Print a NUL-separated file one entry per line
<code>tr '\n' ',' < list.txt</code>	Join lines with commas (leaves a trailing comma)
<code>tr -dc 'A-Za-z0-9' < /dev/urandom head -c 16; echo</code>	Random 16-character string
<code>tr -cs '[:alpha:]' '\n' < file.txt grep -v '^\$' sort uniq -c sort -rn</code>	Count how often each word appears (case-sensitive)

Troubleshooting

Quick checks for common `tr` errors and surprises.

extra operand 'file.txt'	<code>tr</code> does not accept file names; use <code>tr ... < file.txt</code>
extra operand with <code>-d</code> and two sets	<code>-d</code> takes one set; use <code>-ds</code> to delete and squeeze
missing operand after 'x'	Translation needs SET1 and SET2; add SET2 or use <code>-d</code>
Output file is empty after <code>tr ... < f > f</code>	The shell truncates <code>f</code> first; write to a temp file, then <code>mv</code> it
A set starting with <code>-</code> is read as an option	Put the dash last (<code>'_ - '</code>) or add <code>--</code> before the sets
<code>-c</code> output has one extra character	The trailing newline is in the complement; add <code>\n</code> to SET1
Accented letters turn into garbage	<code>tr</code> works on bytes; use <code>sed</code> for multibyte UTF-8 text

Related Guides

Use these guides for full text-processing workflows.

tr Command in Linux	Full <code>tr</code> guide with examples
sed Find and Replace	Replace words and patterns
Bash String Manipulation	Change case and replace text inside the shell
cut Command in Linux	Extract fields after squeezing spaces
uniq Command in Linux	Count repeated lines and words
awk Command in Linux	Pattern-based field processing