Identification.

Erase and Kill Character Conventions
J. H. Saltzer, C. Strachey

Purpose.

Experience has shown that even with sophisticated editor
programs available, two minimal editing conventions at
the earliest possible level are very useful for human
input to a computer system. These two conventions give
the typist these editing capabilities at the instant he
is typing:

1. ·Ability to delete the last character or characters
    typed.

2. Ability to delete all of the current line typed up
    to this point.

(More complex editing capabilities must also be available,
but they fall in the domain of editing programs which
can work with lines previously typed as well as the current
input stream.) By framing these two editing conventions
in the language of the canonical form, it is possible
to preserve the ability to interpret unambiguously a typed
line image despite the fact that editing was required.

Erase convention

The first editing convention is that one graphic, the
number sign (#), is reserved as the _erase_ character.
When this character appears as the only graphic in a print
position, it erases itself and the contents of the previous
print position. When it is not the only graphic in a
print position, it erases the print position in which
it appears. If the erase follows simple carriage motion,
the carriage motion is erased. Several successive erase
characters will erase an equal number of preceding print
positions or simple carriage motions. Since erase processing
occurs _after_ the transformation to canonical form, there
is no ambiguity as to which print position has been erased;
the printed line image is always the guide. Whenever
a print position is erased, the carriage motion (if any)
on the two sides of the erased print position are combined
into a single carriage motion.

Kill convention.

The second editing convention reserves another graphic,
the at sign (@), as the kill character. When this character
appears as the only graphic in a print position, the contents
of that line up to and including the kill character are
discarded. Again, since the kill processing occurs after
the conversion to canonical form, there can be no ambiguity
about which characters have been discarded. By convention,
kill is done before erase, so that it is not possible
to erase a kill character.

Examples.

Example 1:

Typist: abcx#de⟨sp⟩⟨BS⟩fzz#g⟨NL⟩
Printed line: abcx#defzz#g
Canonical form: abcx#defzz#g⟨NL⟩
Final input: abcdefg⟨NL⟩

This example represents the primary use of the erase character,
correcting typing errors the moment they are noted. Note
that the erroneous space between e and f was not erased,
it was undone.

Example 2:

Typist: get⟨HT⟩x#lda⟨HT⟩⟨SP⟩⟨SP⟩word⟨NL⟩
Printed line: get   x#lda   word
Canonical form: get   x#lda   word⟨NL⟩
Final input: get   lda   word⟨NL⟩

The typist places two spaces after his second horizontal
tab because his program has a fixed format input and demands
7 spaces, and the fixed horizontal tab only moves 5 spaces
now. This example demonstrates that programs should not
generally require fixed format input.

Example 3:

Typist:
This @The off⟨BS⟩⟨BS⟩⟨BS⟩___#n⟨BS⟩_ state⟨NL⟩
Printed line:
This @The off#n state
Canonical form:
This @The _⟨BS⟩o_⟨BS⟩f_⟨BS⟩f#_⟨BS⟩n state⟨NL⟩
Final input:
The _⟨BS⟩o_⟨BS⟩n state⟨NL⟩
Printed appearance of final form:
The on state