1. Addresses vs Contents of Strings

In C, a string in a sequence of chars stored in memory with a nul char ('\0') at the end to indicate the end of the string. The block of memory that contains that string has an address.

In C, as in any other language, a string has contents and a string has a location. In C, unlike some other languages, you can decide to work with the address or with the contents. Understanding the difference is essential to effective use of the language.

The diagram below shows how assigning an address instead of copying the contents can lead to odd, confusing results.

2. The Wrong Way to Keep a Copy of a String

utl

main()

utbuf

(1) Records are copied
one-by-one from the
file on the disk into
utbuf, a struct in main.

(2) You pass the address of
utbuf into calc_connect.

In that calc_connect, the address
of utbuf is stored in the pointer
variable called utm.

This picture shows that utm
points to the struct in main().

calc_connect(&utbuf, av[1]);

calc_connect()

utm

(3) In calc_connect, you set
  utl = utm->ut_line
which sets the global variable
utl to POINT TO the ut_line member of the utbuf variable in main.

Each time you read the next
record from the disk, that
member contains the ut_line
for that record.

(4) Problem: You have not made a
copy of what is in ut_line. Instead,
you have pointed to the member
where it is copied from the disk.

(5) Solution: create an array of
chars and COPY the string from
ut_line into that array.