



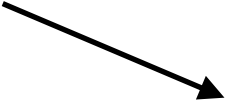
**C\*Characters and Strings**

- C\* characters are ASCII-encoded.
- C\* character literals are characters in code like 'c'.
- C\* strings are stored as null-terminated sequences of characters.  
*Alternatively the end of a string could be identified by storing the number of characters at its beginning.*
- C\* string literals are strings in code like "this".
- The difference between 'a' and "a".

ascii representation  
(**numerical value**)  
in memory

**pointer** to first  
word of where the  
string is stored





# C\* versus C Strings

- C\* strings are arrays of unsigned 64-bit integers
- C strings are arrays of characters, that is, of type `char`
- For example, `"Hello World!"` is equal to `0x6F57206F6C6C6548` in C\* but `0x48` in C
- Note that `0x48` is ASCII for `H` while `0x65`, `0x6C`, `0x6F`, `0x20`, and `0x57` are ASCII for `e`, `l`, `o`, `space`, and `w`, respectively.



# C\* Characters and Strings

- C\* characters are ASCII-encoded.
- C\* character literals are characters in code like 'c'.
- C\* strings are stored as null-terminated sequences of characters.  
*Alternatively the end of a string could be identified by storing the number of characters at its beginning.*
- C\* string literals are strings in code like "this".
- The difference between 'a' and "a".

ascii representation  
(**numerical value**)  
in memory

**pointer** to first  
word of where the  
string is stored