10/24/24

Arrays of type char have some special features.

The stuff in this lecture is not about the string class. Before there was a string class, or any classes, there was a desire to treat arrays of char in a special way that facilitated treating them like Text

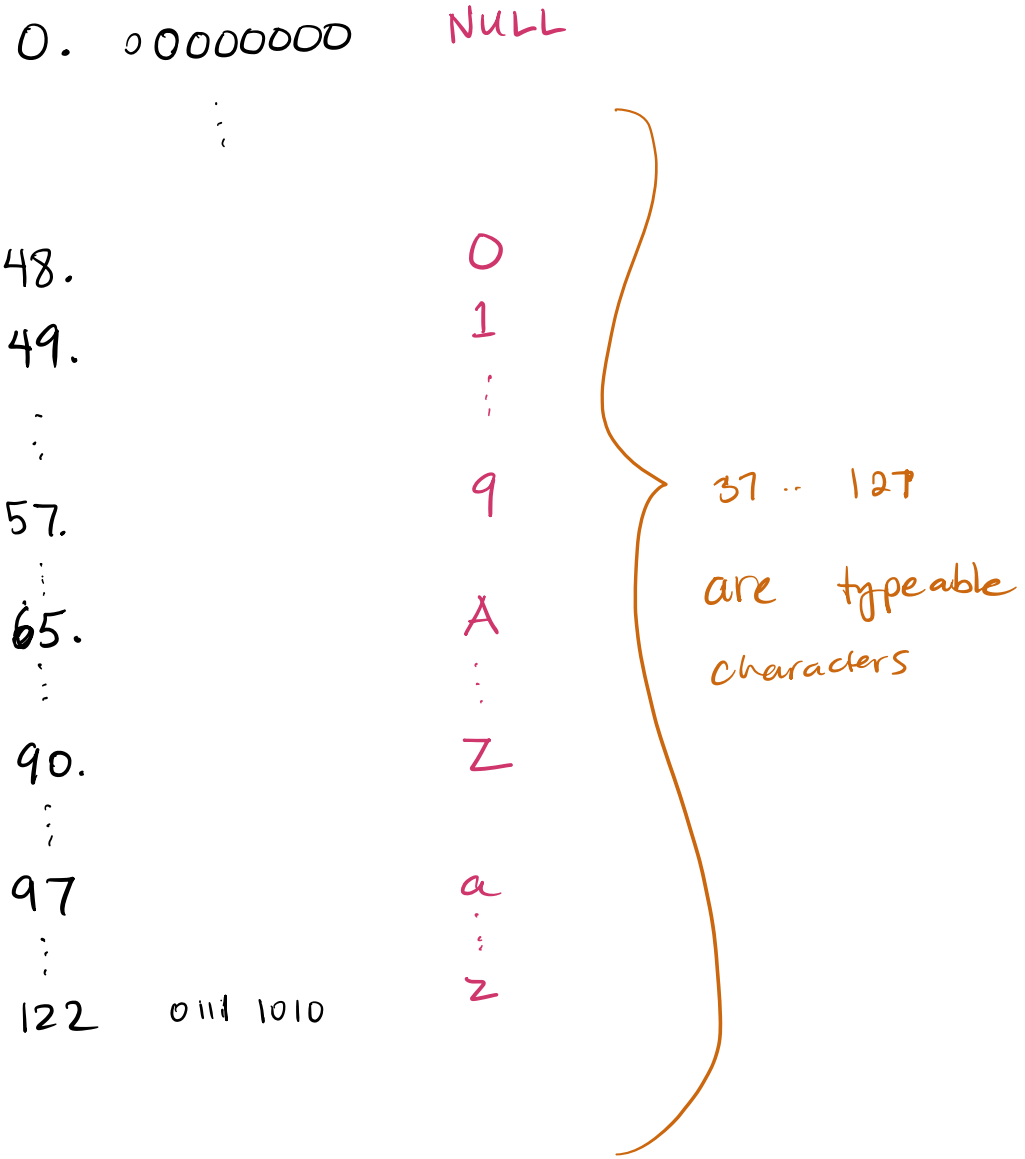
Need to reserve space for text, but don't know how many characters will be in the text.

The following does not apply to arrays of other types.

... not on arrays of ints, floats, doubles, bools, ...

The NULL terminator

Characters in C++ are represented by ASCII code (www.asciitable.com)



The NULL terminator

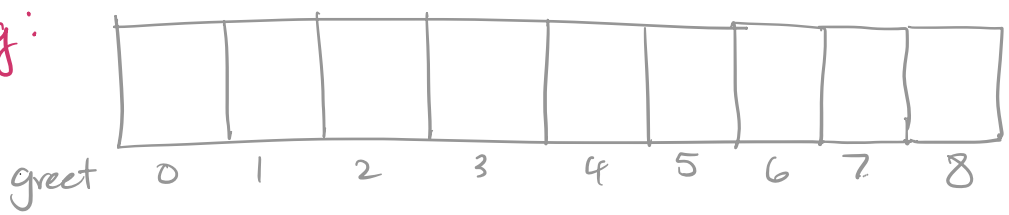
Characters in C++ are represented by ASCII code (www.asciitable.com)

0.	00000000	NULL	'\0' = NULL
	⋮		'\32' = space
48.		0	
49.		1	
⋮		⋮	
57.		9	
⋮			
65.		A	'\65' = 'A'
⋮		⋮	
90.		Z	
⋮			
97		a	
⋮		⋮	
122	0111 1010	z	

cin – (and another input-reading function, scanf) can read a word into a char array

– puts '\0' as first character after end of read word.

Eg:

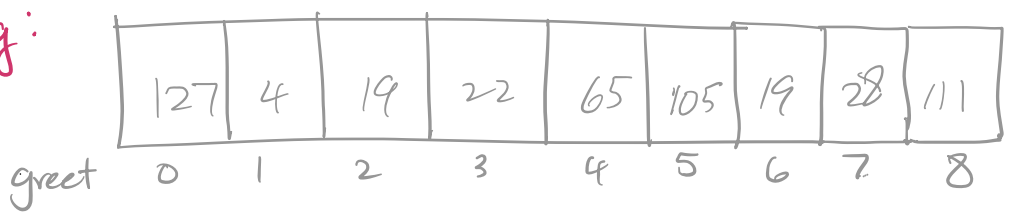



```
char greet[9];
```

cin – (and another input-reading function, scanf) can read a word into a char array

- puts '\0' as first character after end of read word.

Eg:



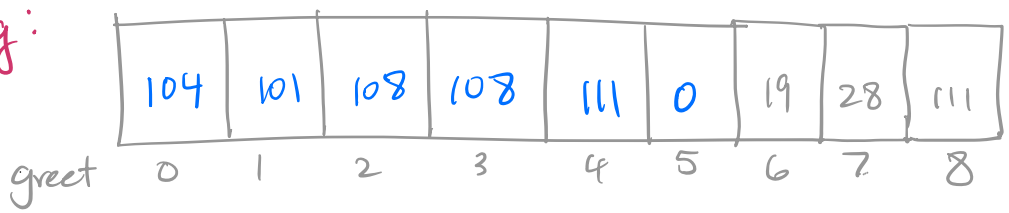

```
char greet[9];

cin >> greet;  // user enters "hello"
```

cin – (and another input-reading function, scanf) can read a word into a char array

- puts '\0' as first character after end of read word.

Eg:




```
char greet[9];

cin >> greet;  // user enters "hello"
```

Note: cin does not check that there is enough space for the string.

Sometimes we want the whole line
(to 'newline' '\12')

ie. including ' ' '\32'

... Then we use getline

```
cin.getline(text);
```

↑ char text[100];

Or, if we are using C++ strings, we have a slightly different call for getline:

```
getline(cin, s);
```

↑ #include <string>
string s;

To read a character, even if it is white space:

```
#include <cstdio>
:
char ch = getc(stdin);
```

<iostream> can give you a char including WS (white space) as follows:

```
#include <iostream>
:
cin >> noskipws >> ch;
```

cout (and printf) can display contents of a character array. It stops at first '\0' (null)

```
cout << text;
printf("%s", text); // We won't cover these functions, but if you see them in code, know they are input/output functions
```

printf	scanf
fprintf	fscanf

Risk: if no '\0', will keep printing until it encounters one (a lot of garbage text).

```
arr[0] = 'h';
arr[1] = 'i';
arr[2] = ' ';
arr[3] = 'X';
arr[4] = '\0';
arr[5] = 'Z';
cout << arr;
arr[0] = 0;
cout << arr;
arr[0]++;
cout << arr;
arr[0] = 104;
```

You can use ++ -- on chars;
you get the next char.

#include <cstring>

- gives you many useful functions on null-terminated character arrays.

```
char s1[sz], s2[sz], s3[sz];

strlen(s1); // returns #of chars before leftmost '\0'

strcpy(s1, s2) // copies s1 into s2

strcat(s1, s2) // copies s2 onto end of s1

strncpy(s1, s2, N) // strcpy, but ≤ N chars

strncat(s1, s2, N) // strcat, but ≤ N chars.
```