

6.S077
(aka 6.0004)

C Strings
And other things

Spring 2022

strtok

```
char * strtok ( char * str, const char * delimiters );
```

- **strtok** is a “stateful” function that you can call repeatedly to chop up a string into substrings
- It does this by scanning **str** for *any* chars that are in the **delimeters** c-string and replacing them with a NULL before returning a pointer to the beginning of the sub-c-string terminated by that NULL
- Calling repeatedly with NULL for value of **str** tells it to keep moving through the string

*in C99, compiler is smart enough to determine if thing being looked at is variable length such as a Variable Length Array (but that is beyond the scope of this class)

strtok in action

```
#include <stdio.h>
#include <string.h>
int main (void){
    char message[22] ="This is a string.";
    char* pch; //pointer
    printf ("Splitting string \"%s\" into tokens:\n",message);
    pch = strtok (message," ,.-");
    while (pch != NULL){
        printf ("%s\n",pch);
        pch = strtok (NULL, " ,.-");
    }
    return 0;
}
```

Prints: This
is
a
string

strtok in action

CHAR ARRAY CREATION:

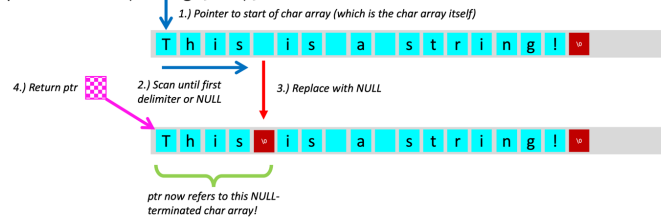
```
char message[22] = "This is a message!";
```

Array (grey) stores values (green)

T h i s i s a s t r i n g ! \0

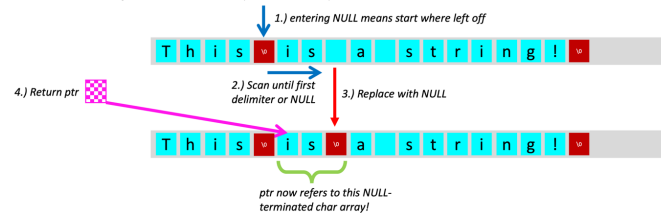
FIRST CALL:

```
char* ptr = strtok(message, " ");
```



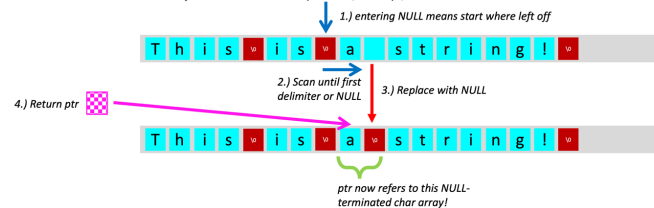
SECOND CALL:

```
char* ptr = strtok(NULL, " ");
```



THIRD CALL:

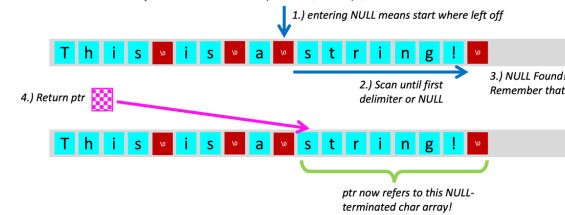
```
char* ptr = strtok(NULL, " ");
```



■ = '\0' //NULL
■ = some non-NULL char
■ → = returned pointer
→ = assignment/overwrite
→ = path of scan/checking

FOURTH CALL:

```
char* ptr = strtok(NULL, " ");
```



FIFTH, ETC... CALL:

```
char* ptr = strtok(NULL, " ");
```

