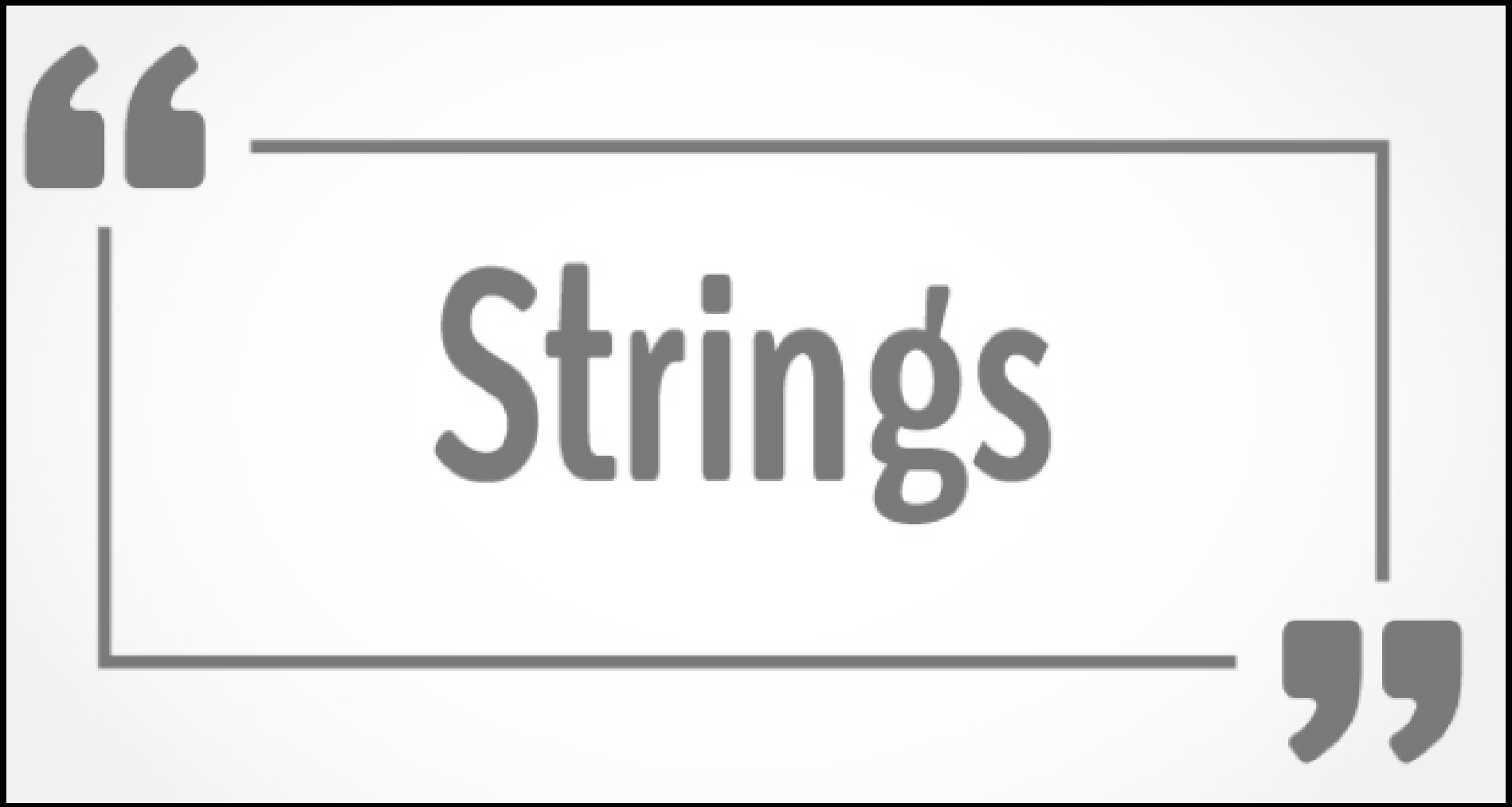
CSE102
Computer Programming



Credits: www.programiz.com

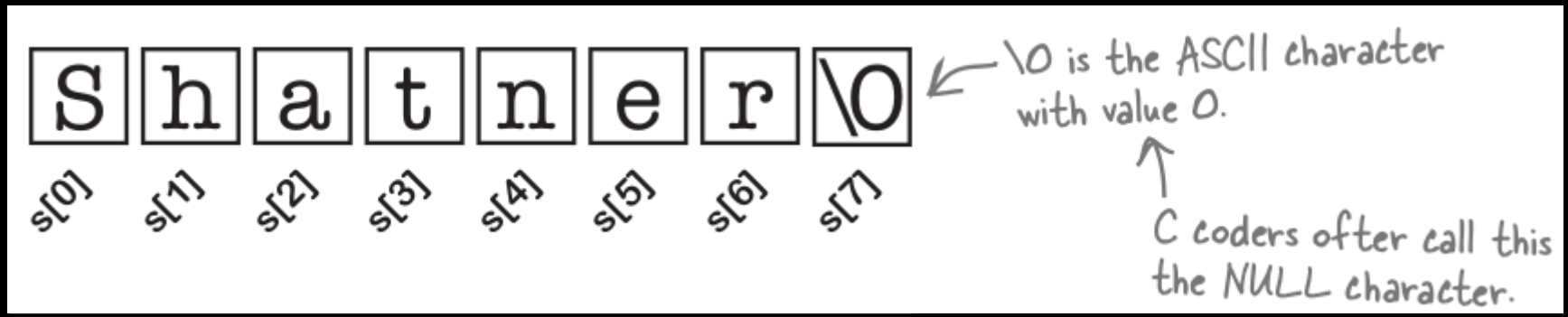
Strings - Character Arrays

```
s = "Shatner"
```

C sees/reads strings as character arrays

```
s = {'S', 'h', 'a', 't', 'n', 'e', 'r'}
```

and stores is memory like this



Credits: Head First C

Initialize Strings

```
char color[] = “blue”;
char color[] =
{‘b’, ‘l’, ‘u’, ‘e’,‘\0’};

char color[10];
scanf(“%9s”,color);
```

Credits: How to Program C

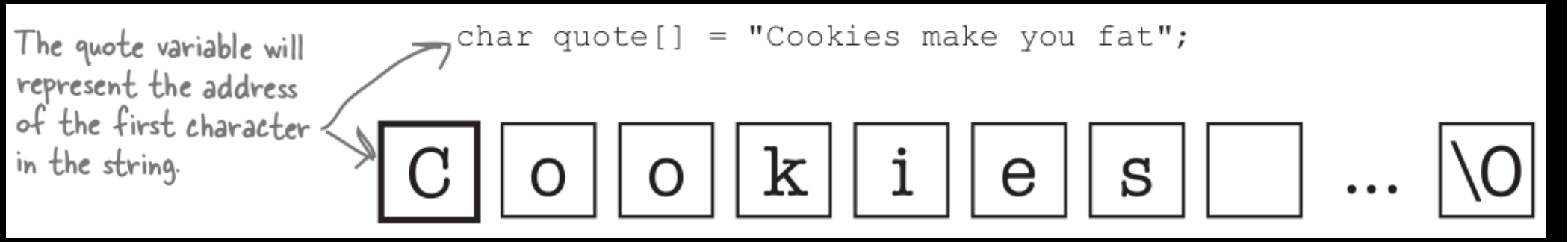
Initialize Strings

```
char c[] = "abcd";
       OR,
char c[50] = "abcd";
       OR,
char c[] = {'a', 'b', 'c', 'd', '\0'};
       OR,
char c[5] = {'a', 'b', 'c', 'd', '\0'};
```

```
char *c = "abcd";
```

Credits: How to Program C

If Strings are Arrays?!




```
void fortune_cookie(char msg[])
{
  printf("Message reads: %s\n", msg);
  printf("msg occupies %i bytes\n", sizeof(msg));
}
```

msg is actually a pointer variable.

msg points to the message.

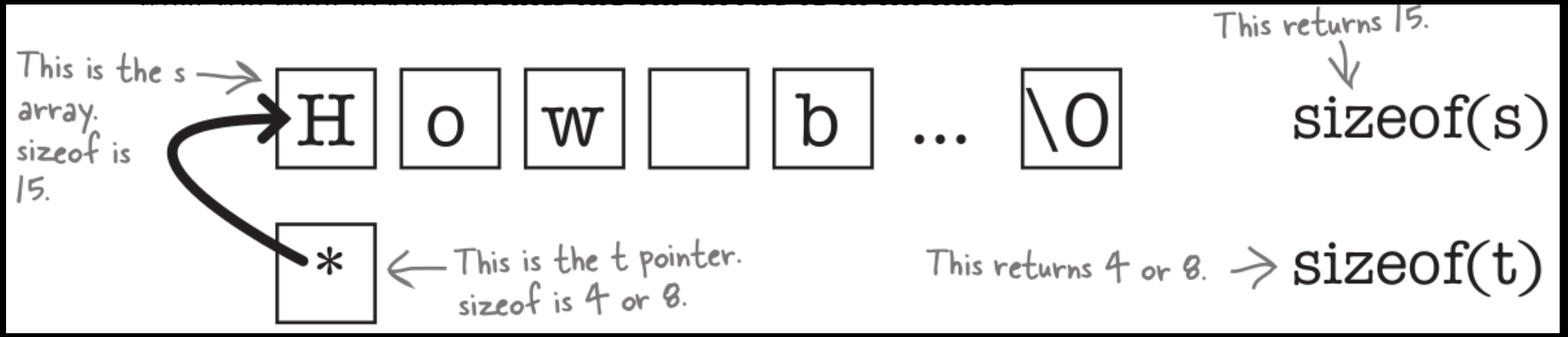
sizeof(msg) is just the size of a pointer.

Credits: Head First C

An Array isn't a Pointer!

```
char s[] = "How big is it?";

char *t = s;
```

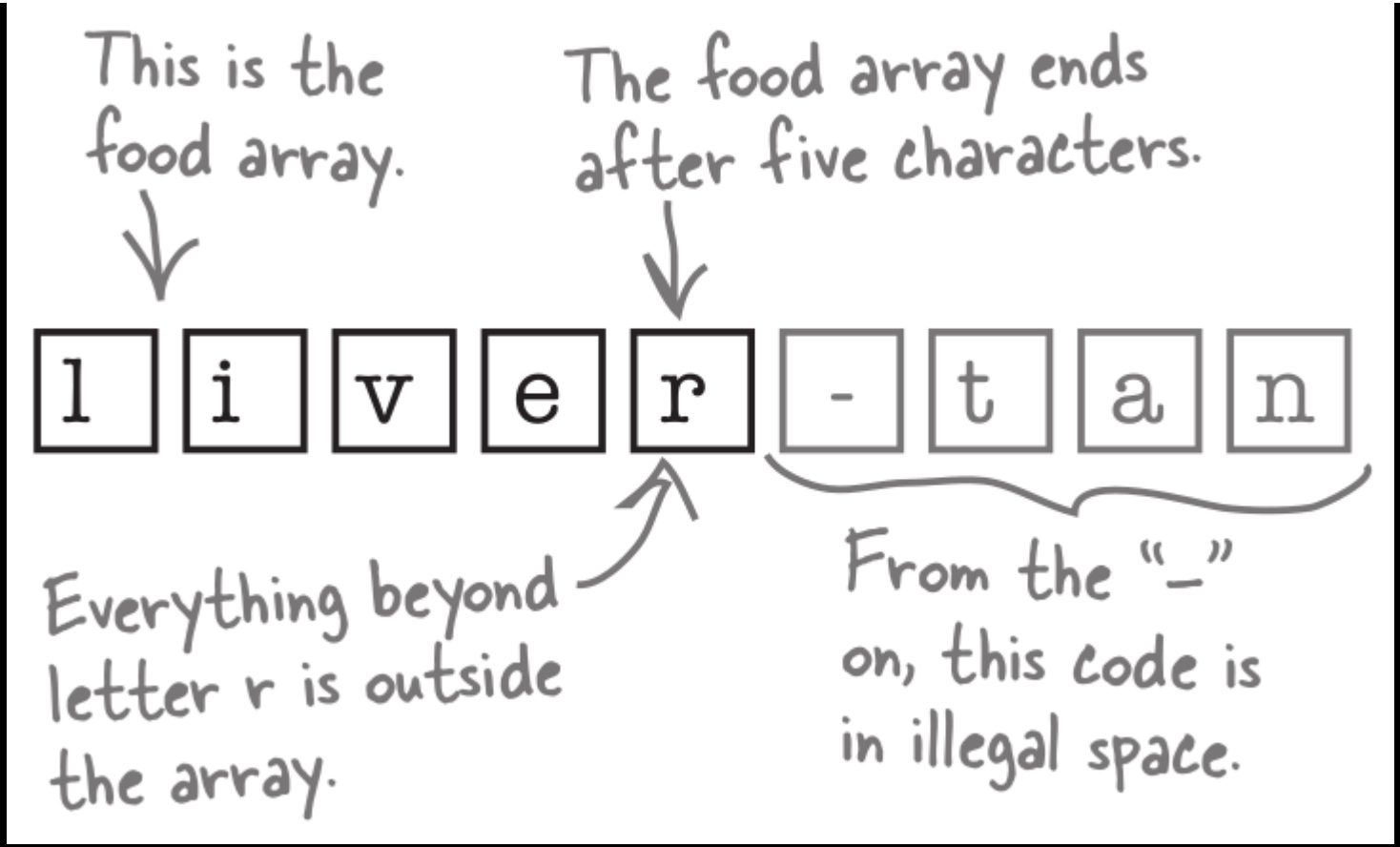


&s == s **&t != t**

Credits: Head First C

Scanf Strings Watchout!?

```
char food[5];
printf("Enter favorite food: ");
scanf("%s", food);
printf("Favorite food: %s\n", food);
```



Credits: Head First C

Remember fgets()

This is the same program as before.

```
char food[5];
printf("Enter favorite food: ");
fgets(food, sizeof(food), stdin);
```

First, it takes a pointer to a buffer.

Next, it takes a maximum size of the string ('/0' included).

stdin just means the data will be coming from the keyboard.

Credits: Head First C

Three-Card Monte

```
#include <stdio.h>


int main()
{
   char *cards = "JQK";
   char a_card = cards[2];
   cards[2] = cards[1];
   cards[1] = cards[0];
   cards[0] = cards[2];
   cards[2] = cards[1];
   cards[1] = a_card;
   puts(cards);
   return 0;
}
```

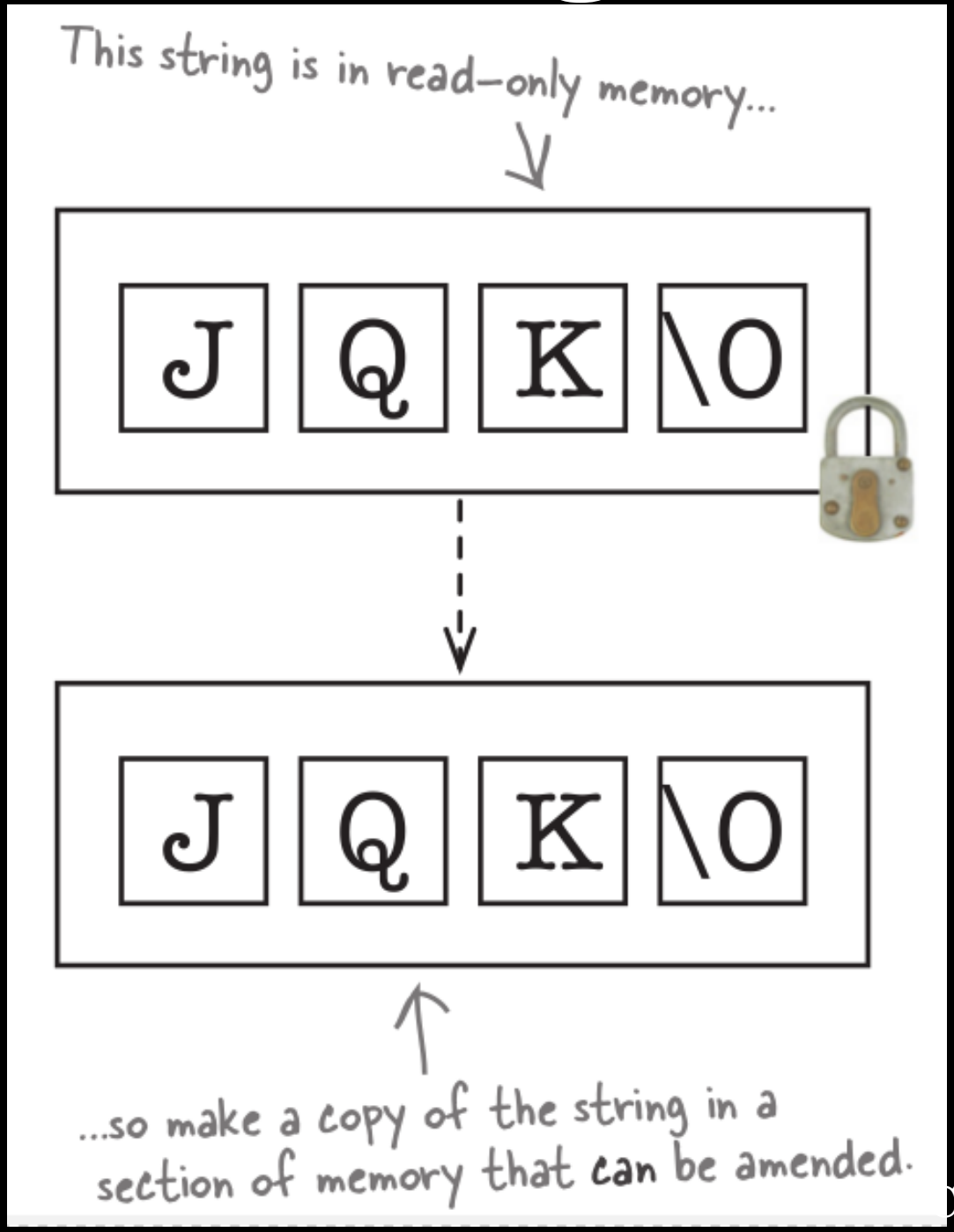
Credits: Head First C

Update String Literals

```
// This can't modify the string
char *cards = "JQK";

// This can modify the string!!
char cards[] = "JQK";
```

Update String Literals



Credits: Head First C

cards[] or *cards

If you see a declaration like this, what does it *really* mean?

```
char cards[]
```

Well, it **depends on where you see it**. If it's a normal variable declaration, then it means that `cards` is an array, and you have to set it to a value immediately:

```
int my_function()
{
  char cards[] = "JQK";
  ...
}
```

cards is an array.

There's no array size given, so you have to set it to something immediately.

But if cards is being declared as a *function argument*, it means that `cards` is a **pointer**:

```
void stack_deck(char cards[])
{
  ...
}

void stack_deck(char *cards)
{
  ...
}
```

cards is a char pointer.

These two functions are equivalent.

Credits: Head First C

Best Practice

One way to avoid this problem in the future is to never write code that sets a simple `char` pointer to a string literal value like:

```
char *s = "Some string";
```

There's nothing wrong with setting a pointer to a string literal—the problems only happen when you try to *modify* a string literal. Instead, if you want to set a pointer to a literal, always make sure you use the `const` keyword:

```
const char *s = "some string";
```

That way, if the compiler sees some code that tries to modify the string, it will give you a compile error:

```
s[0] = 'S';
monte.c:7: error: assignment of read-only location
```

Credits: Head First C

String.h

This first set of brackets is for the array of all strings.

The second set of brackets is used for each individual string.

You know that track names will never get longer than 79 characters, so set the value to 80.

The compiler can tell that you have five strings, so you don't need a number between these brackets.

Each string is an array, so this is an array of arrays.

```
char tracks[][80] = {
  "I left my heart in Harvard Med School",
  "Newark, Newark - a wonderful town",
  "Dancing with a Dork",
  "From here to maternity",
  "The girl from Iwo Jima",
};
```

Compare two strings to each other

Make a copy of a string

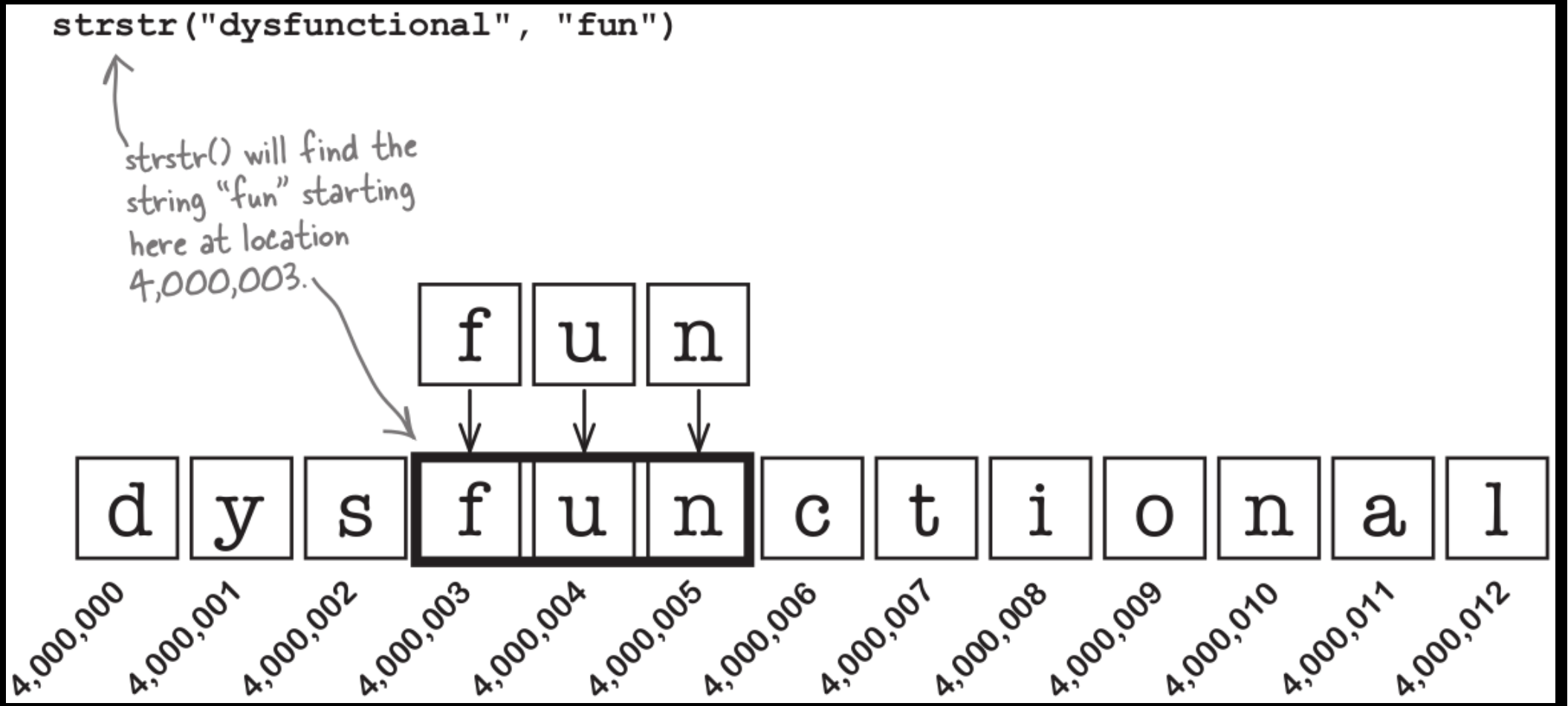
Search for a string

Slice a string into little pieces

There are plenty of other exciting things in string.h for you to play with; this is just for starters.

Credits: Head First C

String Manipulation



Credits: Head First C

String Manipulation

```
char s0[] = "dysfunctional";
char s1[] = "fun";
if (strstr(s0, s1))
  puts("I found the fun in dysfunctional!");
```

Credits: Head First C

String Manipulation

Function prototype	Function description
`char *strcpy( char *s1, const char *s2 )`	*Copies* string `s2` into array `s1`. The value of `s1` is returned.
`char *strncpy( char *s1, const char *s2, size_t n )`	*Copies at most `n` characters* of string `s2` into array `s1`. The value of `s1` is returned.
`char *strcat( char *s1, const char *s2 )`	*Appends* string `s2` to array `s1`. The first character of `s2` *overwrites the terminating null character* of `s1`. The value of `s1` is returned.
`char *strncat( char *s1, const char *s2, size_t n )`	*Appends at most `n` characters* of string `s2` to array `s1`. The first character of `s2` *overwrites the terminating null character* of `s1`. The value of `s1` is returned.

Credits: How to Program C

String Manipulation

Function prototype	Function description
`int strcmp( const char *s1, const char *s2 );`	*Compares* the string `s1` with the string `s2`. The function returns `0`, less than `0` or greater than `0` if `s1` is equal to, less than or greater than `s2`, respectively.
`int strncmp( const char *s1, const char *s2, size_t n );`	*Compares up to `n` characters* of the string `s1` with the string `s2`. The function returns `0`, less than `0` or greater than `0` if `s1` is equal to, less than or greater than `s2`, respectively.

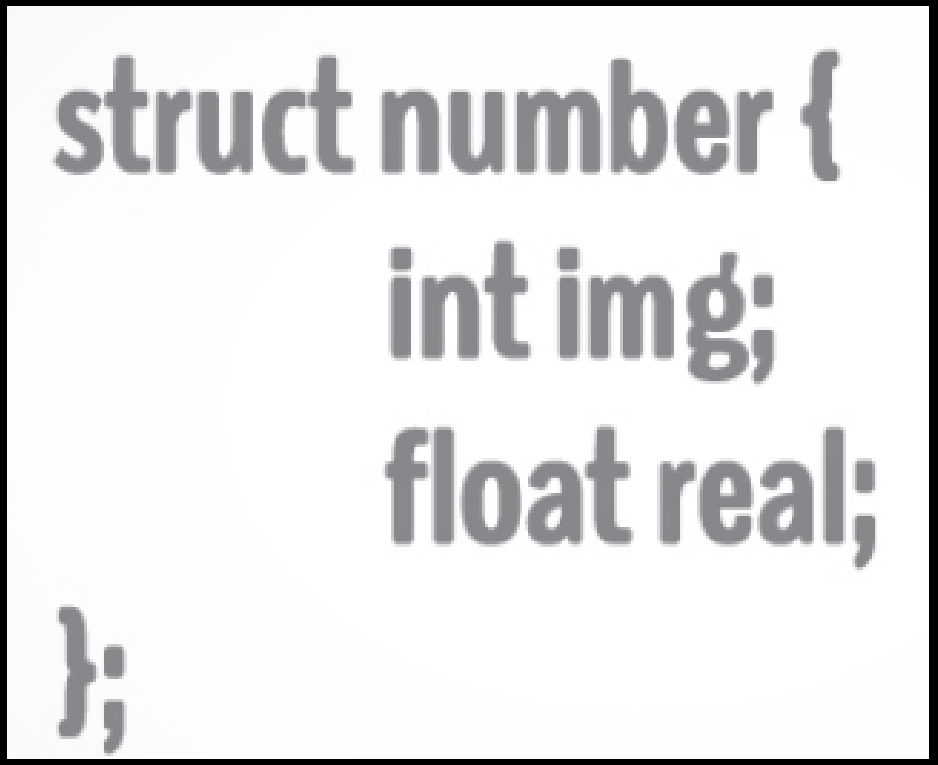
Credits: How to Program C

String Manipulation

Function	Work of Function
strlen()	Calculates the length of string
strcpy()	Copies a string to another string
strcat()	Concatenates(joins) two strings
strcmp()	Compares two string
strlwr()	Converts string to lowercase
strupr()	Converts string to uppercase

Credits: www.programiz.com

CSE102
Computer Programming
(Next Topic)



Credits: www.programiz.com