a place of mind
THE UNIVERSITY OF BRITISH COLUMBIA

CPSC 259

C strings

String functions

Strings

- What is a C string?
  - an array (string) of `char` that terminates in a null character (`'\0'`)

- Different ways to create strings
  - `char an_array[6] = {'H', 'e', 'l', 'l', 'o', '\0'};`
  - `char str[SOMESIZE] = "A string of char";`
    - This automatically gives a null char at the end
  - `char* another_string = "A string of char";`
    - This automatically gives a null char at the end

String length

How long is a piece of string?

- C provides a group of functions for string processing, declared in a header `string.h`

```
#include <string.h>
```

- Calculating length

```
size_t strlen(const char* s);
```

  - `size_t` is an unsigned data type defined by several C/C++ standards

not the same as the array length

```
char mystring[] = "Hello there";
int length;
length = strlen(mystring);
printf("The string '%s' has %d letters\n", mystring, length);
```

11

String comparison

"Hello"
"there"

- Comparing integers and characters: use ==

```c
int a = 6;
int b = 7;
if (a == b) { ... }
```

```c
char a = 'a';
char b = 'b';
if (a == b) { ... }
```

- To compare strings in C:

```c
int strcmp(const char* str1, const char* str2);
int strncmp(const char* str1, const char* str2, size_t num);
```

\0

```c
char string1[] = "Hello";
char string2[] = "Hello there";
int length;

length = strlen(string1);
if (strncmp(string1, string2, length) == 0) {
  printf("The first %d letters of %s and
          %s are the same\n", length, string1, string2);
} else {
  printf("Not the same\n");
}
```

\0

Return values:
< 0: first non-match is smaller in str1
== 0: contents are equal
> 0: first non-match is greater in str1

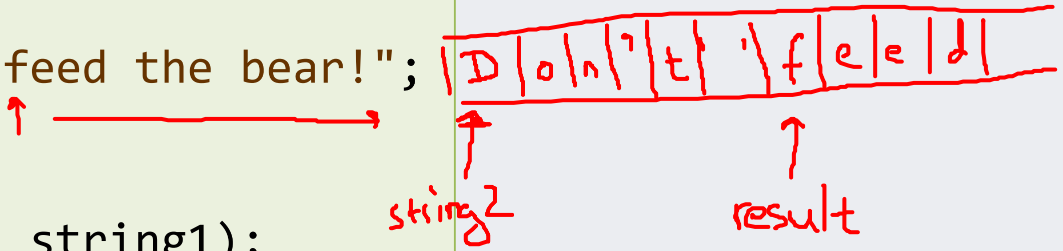
String searching

- Searching – check if a string contains another string

```
char* strstr(const char* search_in, const char* search_for);
```

  - locates the first occurrence of the entire `search_for` string within the `search_in` string, or NULL if not found

```
char string1[] = "feed";
char string2[] = "Don't feed the bear!";
char* result = NULL;

result = strstr(string2, string1);
printf("%s\n", result);

result = strstr(string2, "Please");
if (result == NULL) {
  printf("Not found\n");
}
```



```
feed the bear!
Not found
```

Exercise 3.1

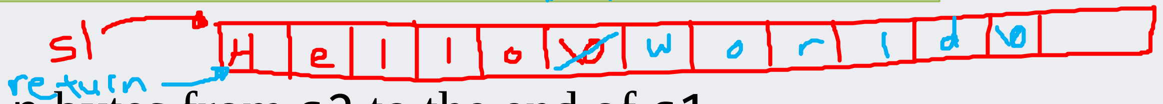
- Suppose we have a long text string stored as `char* hpatps`, and a short search string stored as `char* hwmnbn`.
- How can we find the address of the *second* occurrence of `hwmnbn`?

```
char* first_occurrence = strstr(hpatps, hwmnbn);

char* second_occurence = strstr ( ~~first_occurrence~~,     hwmnbn);

                                  first_occurrence + strlen(hwmnbn)

                                  first_occurrence + 1
```

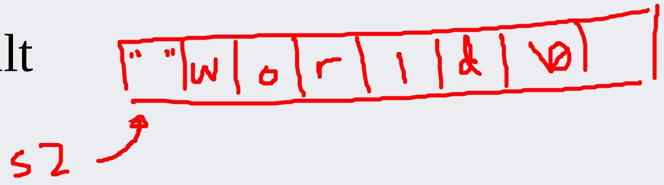
String concatenation

- Concatenation

```
char* strncat(char* s1, const char* s2, size_t n);
```



  - appends no more than `n` bytes from `s2` to the end of `s1`
  - The initial byte of `s2` overwrites the null byte of `s1`
  - A terminating null byte is appended to the result
  - returns `s1` (with `s2` concatenated)



```
char* strcat(char* s1, const char* s2);
```

```
char* empty_string;
char a_long_string[128] = "These ";
strcat(a_long_string, "strings ");
strcat(a_long_string, "are ");
empty_string = strcat(a_long_string, "concatenated!");
printf("%s\n", empty_string);
```

String copying

- Copying

```
char* strncpy(char* dest, const char* src, size_t n);
```

  - copies not more than `n` bytes from the string pointed to by `src` to the string pointed to by `dest`
  - returns `dest`

```
char* strcpy(char* dest, const char* src);
```

```
char a_str[] = "Make news!";
int length = strlen(a_str);
char* other_str = (char*) malloc(length+1); // why +1?
strcpy(other_str, a_str);
a_str[0] = 'F';
printf("a_str = %s\notherstr = %s\n", a_str, other_str);
```

iClicker 3.1

- Suppose we have a `char* str_a` containing `"this is a long string"`, and a `char* str_b` containing `"a short string"`.
- What character is found at `str_a[16]` after performing `strcpy(str_a, str_b);`?

A. `'\0'` (the null character)

[✓ B.] `'t'`

C. The index is out of bounds

D. Participation credit :3

```
str_a
  ↘
  t h i s   i s   a   l o n g   s t r i n g \0
  0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 . — -
  a   s h o r t   s t r i n g \0 s t r i n g \0 . — -
```

Readings for this lesson

- Thareja
  - Chapter 4

- Next class:
  - Thareja, Chapter 5