COMP1511 PROGRAMMING FUNDAMENTALS

LECTURE 9

Some awesome functions for your characters and finally introducing STRINGS!



AST WERK.

- Looked at 2D arrays (which make up

 - a grid and allow us to do some pretty cool stuff)
- Got introduced to pointers

• Learnt about 1D arrays

- Revisit pointers, by solving a problem with pointers
- Learn two new functions available to
 - us: getchar() and putchar()
- Look at strings





Live lecture code can be found here:

HTTPS://CGI.CSE.UNSW.EDU.AU/~CS1511/22T1/LIVE/WEEK05/

WHERE IS THE CODE?

POINTERS RECAP

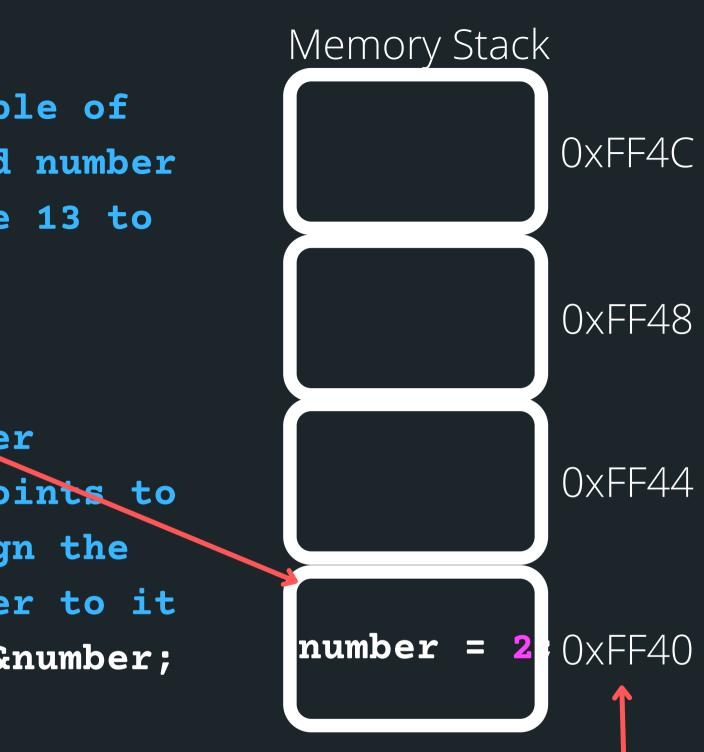
- A pointer is another variable that stores a memory address of a variable
- This is very powerful, as it means you can modify things at the source (this also has certain implications for functions which we will look at in a bit)
- To declare a pointer, you specify what type the pointer points to with an asterisk:
 - type_pointing_to *name_of_ variable;
 - For example, if your pointer points to an int:
- int *pointer;

VISUALLY WHAT IS **HAPPENING?**

// Declare a variable of // type int. called number // Assign the value 13 to // box int number = 2;

// Declare a pointer // variable that points to // an int and assign the // address of number to it int *number_ptr = &number;

AND



- // So now:
- number = 13
- number ptr = 0xFF40

POINTERS

1) Declare a pointer with a * - this is where you will specify what type the pointer points to. For example, a pointer that stores the address of an int type variable: int *number ptr; 2) Initialise a pointer - assign the address to the variable with & number_ptr = &number; 3) Dereference a pointer - using a *, go to the address that this pointer variable is assigned and find what is at that address *number ptr

POINTERS RECAP

THERE ARE **THREE PARTS TO A POINTER**

1. Declare a pointer with a * - this is where you will specify what type the pointer points to

#indlude <stdio.h>

int main (void) {

```
//Declare a variable of type int, called box.
//Assign value 6 to box
int box = 6;
//Declare a pointer variable that points to an int.
/Assign the address of box to it
int *box_ptr = &box;
```

, box_ptr, *box ptr);

return 0:

3. Dereference a pointer -Using a *, go to the address that this pointer variable is assigned and find what is at that address

2. Initialise a pointer - assign the address to the variable with &

printf("The value of the variable 'box' located at address %p is %d\n'

POINTERS

1) Declare a pointer with a * - this is where you will specify what type the pointer points to. For example, a pointer that stores the address of an int type variable: int *number ptr; 2) Initialise a pointer - assign the address to the variable with & number_ptr = &number; 3) Dereference a pointer - using a *, go to the address that this pointer variable is assigned and find what is at that address *number ptr

COMMON **MISTAKES**/ SYNTAX

Let me know in the chat - will this work or not? (yay or nay) int number; int *number ptr;

number ptr = number;

*number_ptr= &number;

number_ptr= &number;

*number_ptr= number;

CODE CODE CODE

A SIMPLE POINTERS EXAMPLE

pointers_simple.c

• A simple pointers example

CODE CODE CODE

ARRAYS AND POINTERS AND FUNCTIONS - LET'S BRING IT ALL TOGETHER...

shufflin.c

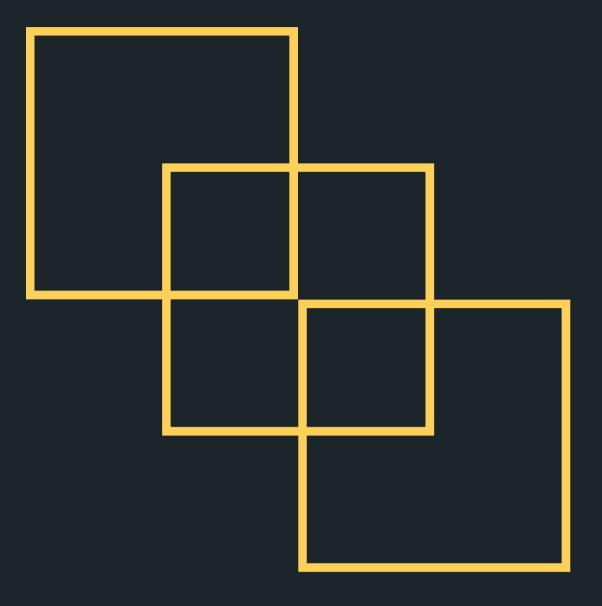
can't return an array*

• The problem is this: Read in an array of numbers (user will specify how many numbers they plan to read in). Then the first number and the last number in the array will be swapped, and the modified array printed out again.

 Let's see and use some pointers. Now remember that you can only return one thing back to main and you

• So without using pointers, can you have a swapping function that swaps out two things? How would you return both of those things back to the main?

REAK TIME



Can you reproduce this figure using just one line, without lifting the pen and without going back over an already drawn line?

C LIBRARIES

GOOD FOR BORROWING A LOT OF FUNCTIONS

GOOD REFERENCE IF YOU ARE INTERESTED IN LEARNING MORE ABOUT EACH LIBRARY:

HTTPS://WWW.TUTORIALSPOINT .COM/C_STANDARD_LIBRARY/IN DEX.HTM



- C has a number of standard libraries available to us
- Libraries are usually .h files (header files)
- We can use these libraries whenever we want to borrow some functions by: #include <library_name.h>
- So far we have used
 - <stdio.h> Standard Input/Output Library
- Other useful libraries we may have seen:
 - <stdlib.h>Standard Library
 - <math.h> Mathematics Library
- Sometimes we can just borrow functions instead of writing them from scratch, like printf, scanf etc.

HELPFUL LIBRARY FUNCTIONS FOR CHARS

GETCHAR()

- getchar() is a function that reads a character from input (a single character) • Reads one byte of input
 - Usually returns an int (ASCII code of that
 - character that it read)
 - Can return -1 (EOF), which is useful for knowing when to finish input
- - will not get its input until enter is pressed at the end of the line (it keeps filling up a buffer until enter is pressed)

HELPFUL LIBRARY FUNCTIONS FOR CHARS

PUTCHAR()

- - to standard output
- Similar to printf("%c", character);

1	<pre>#include <stdio.h></stdio.h></pre>
2	
3	<pre>int main (void) {</pre>
4	
5	<pre>//Declare a variable int</pre>
6	<pre>int character;</pre>
7	<pre>//Use the getchar() func</pre>
8	//Remember that this fun
9	<pre>character = getchar();</pre>
10	-
11	//When you press Ctrl+D
12	//be exited
13	<pre>while (character != EOF)</pre>
14	printf("You entered
15	<pre>//Using the function</pre>
16	<pre>putchar(character);</pre>
17	<pre>printf("\n");</pre>
18	//Get the next chara
19	character = getchar(
20	}
21	return 0;
22	}
22	

• **putchar()** is a function that prints out one character

```
called character
```

```
tion to read one character at a time
nction will take char when a new line is entered
```

```
to signal EOF (end of file) - the while loop will
the character: ");
n putchar to show output one character at a time
```

```
acter from the buffer
);
```

WHY USE **GETCHAR() OVER** SCANF()

- scanf() is a formatted way of reading input from terminal, whereas getchar() reads a single character at a time
- **scanf()** reads a character according to the format specified (%d, %lf, %c), whereas getchar()just reads a single character at a time
- **scanf()** takes in the format and variable address,
 - whereas **getchar()** does not take any input.
- So **scanf()** can do many things and is easy to make mistakes with, if you need one character at a time, it is better to use **getchar()**

WHY USE **PUTCHAR() OVER PRINTF()**

character at a time

• **printf()** is a formatted way of outputting to terminal, whereas **putchar()** outputs a single

SOME OTHER INTERESTING CHARACTER FUNCTIONS

<CTYPE.H> **STANDARD LIBRARY**

CHECK OUT THE REST OF THE FUNCTIONS: HTTPS://WWW.TUTORIALSPOINT.COM/C_ST ANDARD_LIBRARY/CTYPE_H.HTM



Some other useful functions for characters: • **isalpha()** will determine if the character is a letter • **isdigit()** will determine if the character is a

- number
- case letter
- case letter
- letter
- case letter

• **islower()** will determine if the character is a lower

• **isupper()** will determine if the character is an upper

• **tolower()** will convert the character to a lower case

• toupper() will convert the character to an upper

USING SOME OF THESE FUNCTIONS

<pre>//Remember that this fun character = getchar(); //When you press Ctrl+D //be exited while (character != EOF) printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(}</pre>	1	<pre>#include <stdio.h></stdio.h></pre>
<pre>4 int main (void) { 5 6 //Declare a variable int cal 7 int character; 8 9 printf("Enter your name; 10 //Use the getchar() func 11 //Remember that this func 12 character = getchar(); 13 14 //When you press Ctrl+D f 15 //be exited 16 while (character != EOF) 17 printf("You entered f 18 //Using the function 19 putchar(character); 10 printf("\n"); 11 //Check if the chara 12 //Islower() found in 13 if (islower(characte 14 //If it is, then 15 //If it is, then 16 //function toupp 17 character = toup 17 printf("Your new 18 putchar(characte 19 printf("Your new 19 printf("Your new 19 printf("Your new 19 printf("Your new 10 character = toup 17 printf("Your new 18 putchar(characte 19 printf("\n"); 19 } 10 } 11 //Get the next chara 12 character = getchar(13 } 13 return 0; 14 //Get the next chara 15 character = getchar(16 } 17 character = getchar(17 character = ge</pre>	2	<pre>#include <ctype.h></ctype.h></pre>
<pre>5 6 //Declare a variable int cal 7 int character; 8 9 printf("Enter your name a 10 //Use the getchar() func 11 //Remember that this func 12 character = getchar(); 13 14 //When you press Ctrl+D a 15 //be exited 16 while (character != EOF) 17 printf("You entered 18 //Using the function 19 putchar(character); 19 printf("\n"); 10 //Check if the chara 11 //Islower() found in 12 if (islower(characte 12 //If it is, then 13 //function toupper 14 //Get the next chara 15 //Get the next chara 16 character = getchar(17</pre>	3	
<pre>6 //Declare a variable int cal int character; 8 9 printf("Enter your name //Use the getchar() func //Remember that this func character = getchar(); 13 14 //When you press Ctrl+D 15 //be exited 16 while (character != EOF) 17 printf("You entered 18 //Using the function 19 putchar(character); 19 printf("\n"); 10 //Check if the chara 11 //Slower() found in 12 if (islower(character 12 //If it is, then 13 if (islower(character 14 //If it is, then 15 //function toupp 16 character = toup 17 printf("Your new 18 putchar(characte 19 printf("Your new 19 putchar(characte 10 printf("Your new 10 printf("Your new 10 printf("Your new 11 printf("\n"); 12 //Get the next chara 13 character = getchar(14 } 15 return 0;</pre>	4	<pre>int main (void) {</pre>
<pre>7 int character; 8 9 printf("Enter your name; 10 //Use the getchar() func: 11 //Remember that this func: 12 character = getchar(); 13 14 //When you press Ctrl+D 15 //be exited 16 while (character != EOF) 17 printf("You entered 18 //Using the function 19 putchar(character); 19 printf("\n"); 10 printf("\n"); 11 //Check if the chara 12 //Islower() found in 13 if (islower(character) 14 //If it is, then 15 //function toupp 17 character = toup 18 printf("Your new 19 putchar(character) 19 printf("Your new 19 putchar(character) 19 printf("Your new 19 printf("Your new 10 printf("Your new 10 printf("Your new 11 putchar(character) 12 //Get the next chara 13 character = getchar(14 } 15 return 0;</pre>	5	
<pre>8 9 printf("Enter your name i 10 //Use the getchar() func 11 //Remember that this func 12 character = getchar(); 13 14 //When you press Ctrl+D 15 //be exited 16 while (character != EOF) 17 printf("You entered 18 //Using the function 19 putchar(character); 19 printf("\n"); 10 //Check if the chara 19 //If it is, then 19 //function toupp 17 character = toup 18 //If it is, then 19 putchar(characte 19 printf("Your new 19 putchar(characte 19 printf("Your new 10 putchar(characte 10 printf("Your new 11 putchar(characte 12 //If it is, then 13 //Get the next chara 14 //Get the next chara 15 //Get the next chara 16 character = getchar(17 printf("Your new 18 putchar(characte 19 printf("\n"); 19 printf("\n"); 10 printf("\n"); 10 printf("\n"); 10 printf("\n"); 11 printf("\n"); 12 printf("\n"); 13 printf("\n"); 14 printf("\n"); 15 printf("\n"); 15 printf("\n"); 16 printf("\n"); 17 printf("\n"); 18 printf("\n"); 19 printf("\n"); 19 printf("\n"); 10 printf("\n"); 10 printf("\n"); 11 printf("\n"); 12 printf("\n"); 13 printf("\n"); 14 printf("\n"); 15 printf("\n"); 15 printf("\n"); 16 printf("\n"); 17 printf("\n"); 18 printf("\n"); 19 printf("\n"); 19 printf("\n"); 10 print</pre>	6	//Declare a variable int cal
<pre>9 printf("Enter your name a //Use the getchar() funct //Remember that this funct character = getchar(); 3 //When you press Ctrl+D //be exited while (character != EOF) printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	7	<pre>int character;</pre>
<pre>//Use the getchar() func //Remember that this func character = getchar(); //When you press Ctrl+D //be exited while (character != EOF) printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	8	
<pre>//Use the getchar() funct //Remember that this funct character = getchar(); //When you press Ctrl+D //be exited while (character != EOF) printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	9	printf("Enter your name
<pre>//Remember that this fun character = getchar(); //When you press Ctrl+D //be exited while (character != EOF) printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	LΘ	· · · ·
<pre>character = getchar(); //When you press Ctrl+D * //be exited while (character != EOF) printf("You entered * //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower() found in //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	11	//Remember that this fun
<pre>//When you press Ctrl+D //be exited //be exited while (character != EOF) printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	12	
<pre>//be exited while (character != EOF) printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	13	<u> </u>
<pre>//be exited while (character != EOF) printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	4	//When you press Ctrl+D
<pre>printf("You entered //Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	15	
<pre>17 printf("You entered 18 //Using the function 19 putchar(character); 19 printf("\n"); 20 //Check if the chara 21 //Islower() found in 22 //islower() found in 23 if (islower(characte 24 //If it is, then 25 //function toupp 26 character = toup 27 printf("Your new 28 putchar(characte 29 printf("\n"); 20 } 21 //Get the next chara 22 character = getchar(23 } 24 //Get the next chara 25 character = getchar(26 } 27 printf("\n"); 28 putchar(characte) 29 printf("\n"); 20 } 20 } 21 //Get the next chara 23 character = getchar(24 } 25 printf("\n"); 26 printf("\n"); 27 printf("\n"); 28 putchar(characte) 29 printf("\n"); 20 } 20 } 21 //Get the next chara 22 character = getchar() 23 } 24 } 25 printf("\n"); 26 printf("\n"); 27 printf("\n"); 28 printf("\n"); 29 printf("\n"); 20 } 20 } 21 //Get the next chara 22 character = getchar() 23 printf("\n"); 24 printf("\n"); 25 printf("\n"); 26 printf("\n"); 27 printf("\n"); 28 printf("\n"); 29 printf("\n"); 20 printf</pre>	16	
<pre>//Using the function putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	L7	
<pre>putchar(character); printf("\n"); //Check if the chara //islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	18	
<pre>20 printf("\n"); 21 //Check if the chara 22 //islower() found in 23 if (islower(characte 24 //If it is, then 25 //function touppon 26 character = toup 27 printf("Your new 28 putchar(characte 29 printf("\n"); 30 } 31 32 //Get the next chara 33 character = getchar(34 } 35 return 0;</pre>	19	
<pre>//Check if the chara //islower() found in if (islower(characte //If it is, then //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	20	
<pre>//islower() found in if (islower(characte //If it is, then //function toupp character = toup printf("Your new putchar(characte printf("\n"); } } //Get the next chara character = getchar(} return 0;</pre>	21	
<pre>//If it is, then //function touppon character = toup printf("Your new putchar(characte printf("\n"); } } //Get the next characte character = getchar(} return 0;</pre>	22	
<pre>//If it is, then //function touppon character = toup printf("Your new putchar(characte printf("\n"); } } //Get the next characte character = getchar(} return 0;</pre>	23	<pre>if (islower(characte</pre>
character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;	24	
character = toup printf("Your new putchar(characte printf("\n"); } //Get the next chara character = getchar(} return 0;	25	//function toupp
<pre>printf("Your new putchar(characte printf("\n"); } } //Get the next chara character = getchar(} return 0;</pre>	26	
<pre>printf("\n"); printf("\n"); } //Get the next chara character = getchar(} return 0;</pre>	27	printf("Your new
<pre>30 } 31 32 //Get the next chara 33 character = getchar(34 } 35 return 0;</pre>	28	putchar(characte
<pre>31 32 //Get the next chara 33 character = getchar(34 } 35 return 0;</pre>	29	<pre>printf("\n");</pre>
<pre>32 //Get the next character 33 character = getchar(34 } 35 return 0;</pre>	30	}
<pre>character = getchar(} return 0;</pre>	31	-
34 } 35 return θ;	32	<pre>//Get the next chara</pre>
35 return 0;	33	character = getchar(
	34	}
36 }	35	return 0;
	86	}

led character

```
as an example of getchar() and press Enter: ");
tion to read one character at a time
ction will take char when a new line is entered
to signal EOF (end of file) - the while loop will
{
the character: ");
putchar to show output one character at a time
cter is a lower case letter by using the function
<ctype.h> standard library
er)){
convert it to upper case letter by using the
er() found in <ctype.h> standard library
per(character);
character is: ");
er);
cter from the buffer
);
```

STRINGS

WHAT ARE THEY?

- together
 - an array of characters!
- character
- It is a placeholder to indicate that this array of characters is a string

• Strings are a collection of characters that are joined

 There is one very special thing about strings in C - it is an array of characters that finishes with a \0 • This symbol is called a null terminating character • It is always located at the end of an array, therefore an array has to always be able to accomodate this

It is not displayed as part of the string

It is very useful to know when our string has come to

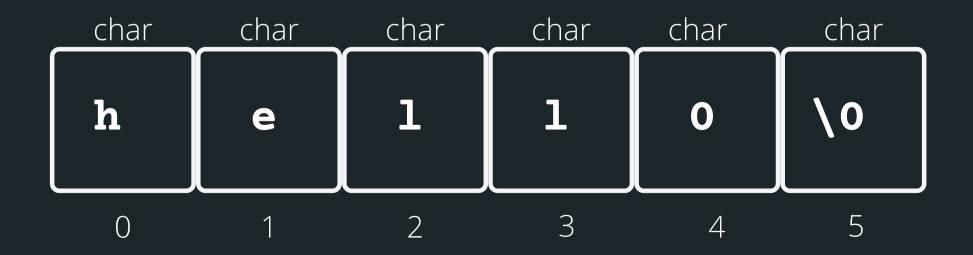
an end, when we loop through the array of characters

HOW DO WE **DECLAREA STRING?**

WHAT DOES IT LOOK **LIKE VISUALLY?**

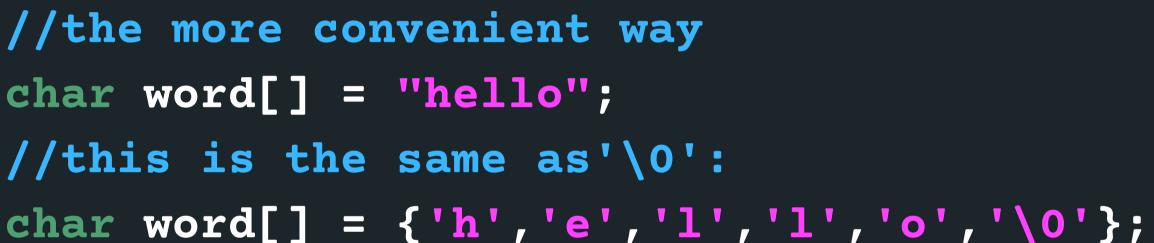
- type is char.
- methods:

//the more convenient way char word[] = "hello"; //this is the same as'\0':



• Because strings are an array of characters, the array

To declare and initialise a string, you can use two



HELPFUL LIBRARY FUNCTIONS FOR STRINGS

FGETS()

There is a useful function for reading strings: fgets(array[], length, stream) The function needs three inputs:

- terminal) char array[MAX_LENGTH]; MAX_LENGTH from terminal input

• array[] - the array that the string will be stored into

• length - the number of characters that will be read in

stream - this is where this string is coming from - you

don't have to worry about this one, in your case, it will always be stdin (the input will always be from

// Declare an array where you will place the string that you read from somewhere // Read in the string into array of length fgets(array, MAX_LENGTH, sdin)

HOW DO KEEP READING **STUFF IN OVER AND** OVER AGAIN?

Using the **NULL** keyword, you can continuously get string input from terminal until Ctrl+D is pressed • fgets() stops reading when either length-1 characters are read, newline character is read or an end of file is reached, whichever comes first 1 #include <stdio.h> 2 3 #define MAX LENGTH 15 5 int main (void) { 6 //1. Declare an array, where you will place the string 7 char array[MAX LENGTH]; 8 10 printf("Type in a string to echo: "); 11 //2. Read a string into the array until Ctrl+D is pressed, which is indicated by NULL keyword 1 11 while (fgets(array, MAX LENGTH, stdin) != NULL) { printf ("The string is: \n"); 15 printf("%s", array); 16 printf("Type in a string to echo: "); 17 18 return 0; 19 }

HELPFUL LIBRARY FUNCTIONS FOR STRINGS

FPUTS()

Another useful function to output strings:

fputs(array[], stream)

The function needs two inputs:

- array[] the array that the string is be stored in
- stream this is where this string will be output to, you
 - don't have to worry about this one, in your case, it will
 - always be stdout (the output will always be in
 - terminal)
- // Declare an array where you will place the string that you read from somewhere
- char array[MAX_LENGTH]; // Read in the string into array of length
- MAX LENGTH from terminal input fgets(array, MAX_LENGTH, sdin) //Output the array now
- fputs(array, stdout)

SOME OTHER INTERESTING STRING FUNCTIONS

<STRING.H> **STANDARD LIBRARY**

CHECK OUT THE REST OF THE FUNCTIONS: HTTPS://WWW.TUTORIALSPOINT.COM/ C_STANDARD_LIBRARY/STRING_H.HTM



Some other useful functions for strings:

- the '\0'
- (concatenate)
- **strcmp()** compare two strings
- character

• **strlen()** gives us the length of the string (excluding)

• **strcpy()** copy the contents of one string to another • **strcat()** attach one string to the end of another

• **strchr()** find the first or last occurance of a

USING SOME OF THESE FUNCTIONS

STRINGS

1	<pre>#include <stdio.h></stdio.h></pre>
2	<pre>#include <string.h></string.h></pre>
3	
4	#define MAX_LENGTH 15
5	
6	<pre>int main (void) {</pre>
7	
8	//Declare an origin
9	char word[MAX_LENGT
10	
11	<pre>//Example using str</pre>
12	<pre>//Example using str //to another (desti</pre>
13	strcpy(word, "Sasha
14	printf("%s\n", word
15	
16	<pre>//Example using str</pre>
17	// '\0':
18	<pre>int length = strlen</pre>
19	printf("The size of
20	
21	<pre>//Example using str</pre>
22	//this function wil
23	<pre>//other int if not</pre>
24	<pre>int compare_string1</pre>
25	printf("The two str
26	
27	compare_string1 = s
28	printf("The two str
29	
30	return 0;
31	}

```
nal array
ΓΗ];
cpy to copy from one string
ination, source):
:");
i);
rlen to find string length (returns int not including
("Sasha");
the string Sasha is: %d\n", length);
rcmp to compare two strings character by character:
ll return 0 if strings are equal
the same
= strcmp("Sasha", "Sashha");
rings are the same: %d\n", compare string1);
strcmp(word, "Sasha");
rings are the same: %d\n", compare string1);
```



Feedback please!

I value your feedback and use to pace the lectures and improve your overall learning experience. If you have any feedback from today's lecture, please follow the link below. Please remember to keep your feedback constructive, so I can action it and improve the learning experience.

https://www.menti.com/1qehxnzw39

WHAT DID WE LEARN TODAY?

POINTERS RECAP

pointers_basic.c shuffling.c

CHAR **FUNCTIONS**

get_char.c char_functions.c

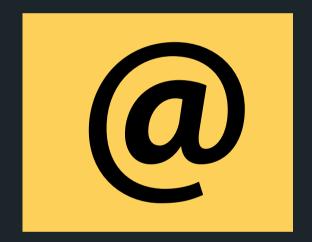


STRINGS

string_functions.c

REACH OUT





CONTENT RELATED QUESTIONS

Check out the forum

ADMIN QUESTIONS cs1511@cse.unsw.edu.au