Arrays and Functions

Arrays & Memory

- This model isn't quite accurate…

test[0]	test[1]	test[2]	test[3]	test[4]	test[5]
1	45	7	1000	-105	42
0x42	0x43	0x44	0x45	0x46	0x47

Arrays & Memory

- This model isn't quite accurate…
  - The memory of an integer is larger than one bit
  - Therefore, there has to be more space between the elements of this array

test[0]	test[1]	test[2]	test[3]	test[4]	test[5]
1	45	7	1000	-105	42
0x42	0x43	0x44	0x45	0x46	0x47

Arrays & Memory

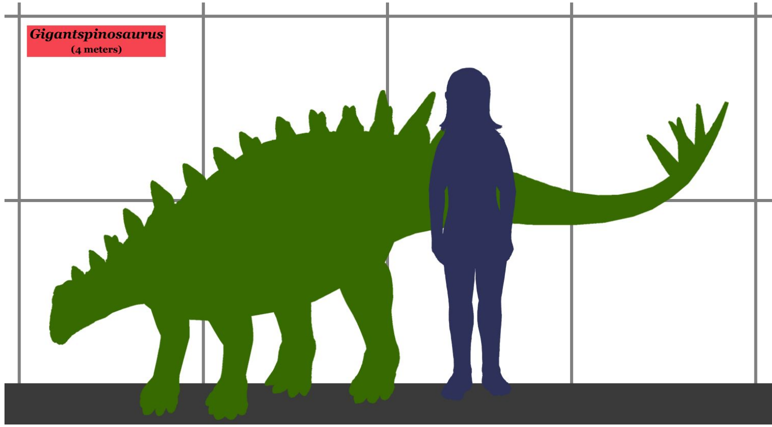
- How much space? It depends on integer memory…
  - Use the sizeof() function to determine this

test[0]	test[1]	test[2]	test[3]	test[4]	test[5]
1	45	7	1000	-105	42
0x42	0x43	0x44	0x45	0x46	0x47

Sizeof()

- Included in <stdio.h>
- Determines the size of variables, in bytes

*See array_sizeof.c in github to see it in action



Arrays and Functions

- Arrays can be passed into functions…
  - As long as the function knows 1) the start location, and 2) the size of each element within the array

  - Start location = name of the array
  - Size = given by sizeof(), OR the type of the array

Arrays and Functions

- Then the function can directly modify the elements
    - Similar to pointers, but without the * / &

Arrays and Functions

- Passing arrays into functions requires specific syntax

- <u>Function Declaration</u>

*> type name(**int test[]**, int max_size);*

/ The array has to include empty square brackets [] to show that it is an array */*

Arrays and Functions

- Passing arrays into functions requires specific syntax

- Function Declaration / Definition

*> type name(int test[]**, int max_size**);*

/ It is usually extremely helpful to include the maximum size of the array as well */*

Arrays and Functions

- Passing arrays into functions requires specific syntax

- <u>Function Invocation</u>

```
> name(test, size);

/*ONLY need to give the name of the
array*/
```