

Timothy Lytkine
CSCI 2113

Module Exercises for C Module 3

***Exercise 3.1:** Type up, compile and execute the above program. Add a second pointer variable that points to *i* and modifies *i* using only the second pointer variable. Can two pointers point to the same thing?*

Two pointers can not point to the same thing because attempting to do so results in a compiler error. (Segmentation fault: 11)

***Exercise 3.2:** In addition to the value in *j* also print the address in *charPtr* in the first *printf* statement.*

```
Timothys-MacBook-Pro:Module3 timothylytkine$ pico Exercise3p2.c
Timothys-MacBook-Pro:Module3 timothylytkine$ gcc -o Exercise3p2 -
std=c99 Exercise3p2.c
Timothys-MacBook-Pro:Module3 timothylytkine$ ./Exercise3p2
First byte: 5
Address in charPtr: 0x7fff58ac7b20
Second byte: 0
Third byte: 0
Fourth byte: 0
```

Exercise 3.4: Draw the "memory picture" for arrays B and B2 in the first arrays example above (arrays.c) just before the free() function calls are executed. That is, draw a picture with sample memory addresses that shows how these arrays are located in memory. Put this in the "answers" PDF for this module.

C Module 3
Exercise 3.4

Software Engineering
Timothy Lytkine

Memory Picture for Arrays B and B2

memory address		pointer = memory address + 1
5,999	6000	B
6,000	0	B[0][0] (B[1][2] = i)
	⋮	
6000 + (8 × 20) = 6,160	361	B[19][19]
7,000	8,000	B2
	⋮	
8,000	9,000	B2[0] ∈ 160 bits (20 × 8)
	(160 for each B[i] from index 1 to 9)	160 × 15 = 3040
	12,040	B2[19]
		B2[0][0] B2[i][j] = i
		B2[0][19]
		B2[1][0]
		B2[1][19]
		B2[2][0]
		B2[2][19]
		(Keeps going)

8 bytes
20 #'s

functions part of the
- B and B2 are stacks which hold pointers to the addresses where malloc manages the memory outside the stack and holds 8-bit values (doubles)

malloc manages 9,000

9,160

9,312

9,320

9,480

