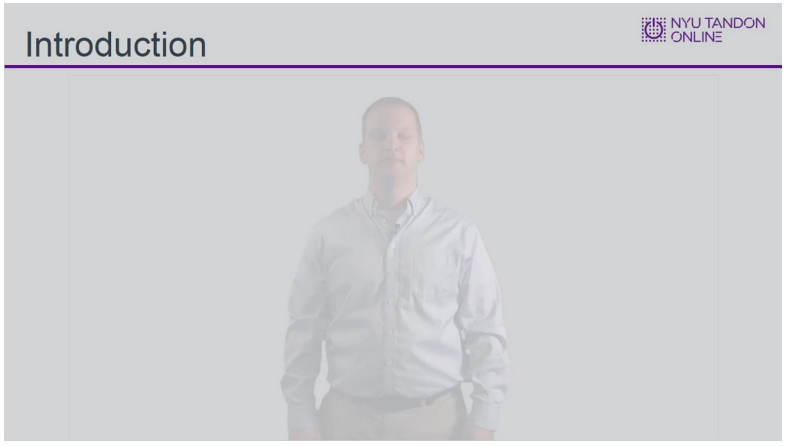
CS Bridge Module 11 Pointers and Dynamic Storage

1. Pointers and Dynamic Storage

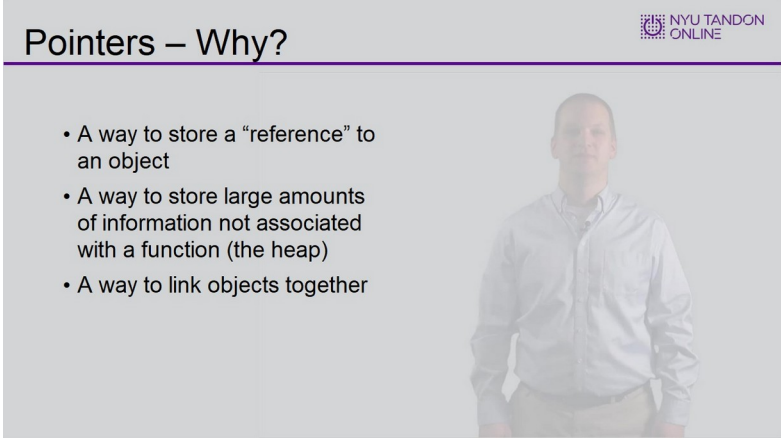
1.1 Title Slide

Notes:

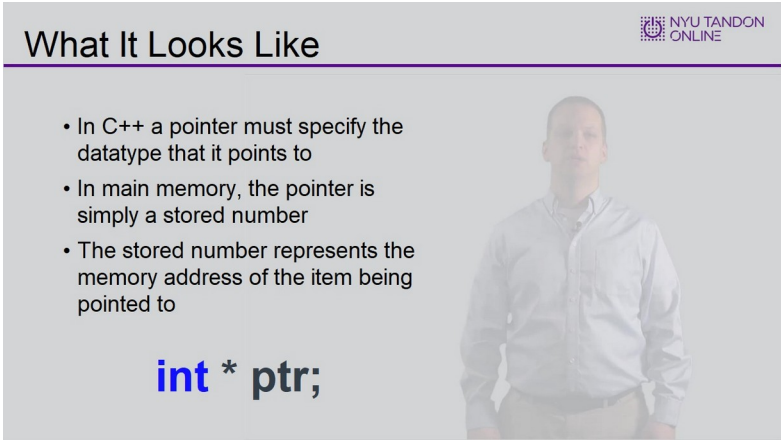
1.2 Introduction



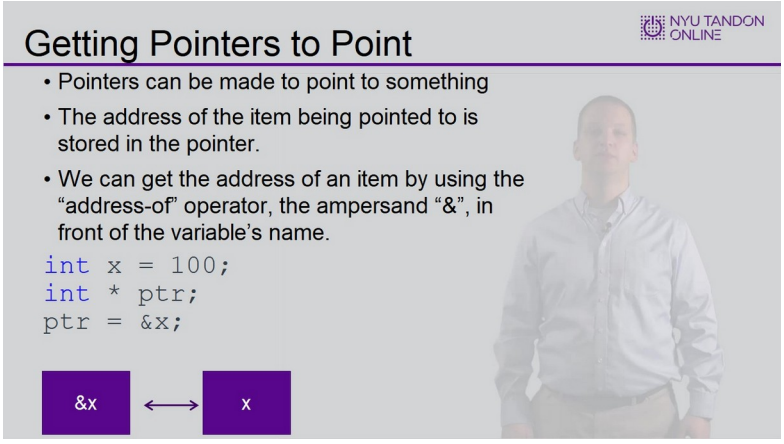
1.3 Pointers – Why?



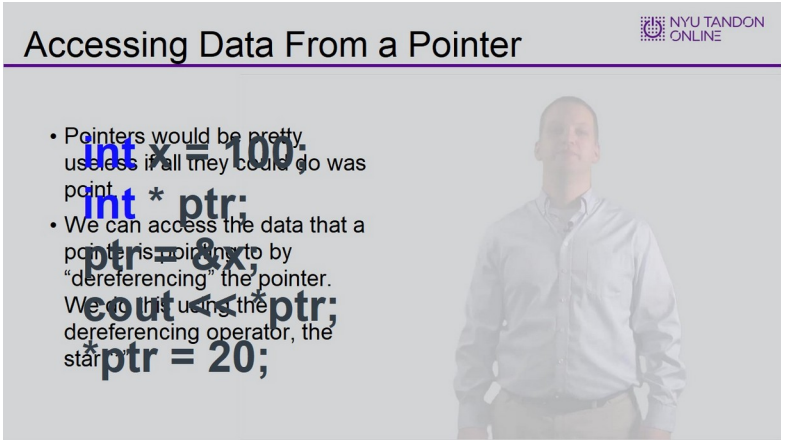
1.4 What it looks like



1.5 Getting pointers to point

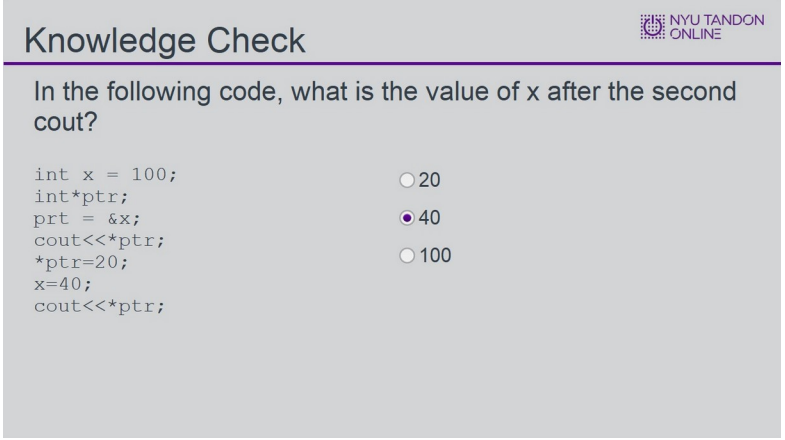


1.6 Accessing data from a pointer



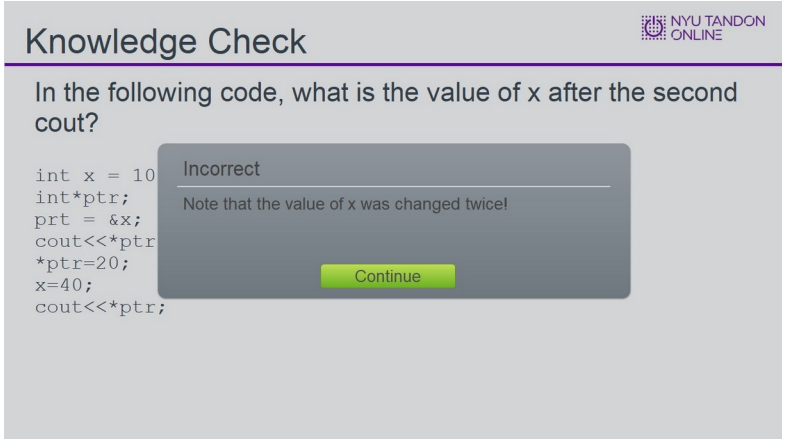
1.7 Knowledge Check

(Multiple Choice, 10 points, 1 attempt permitted)

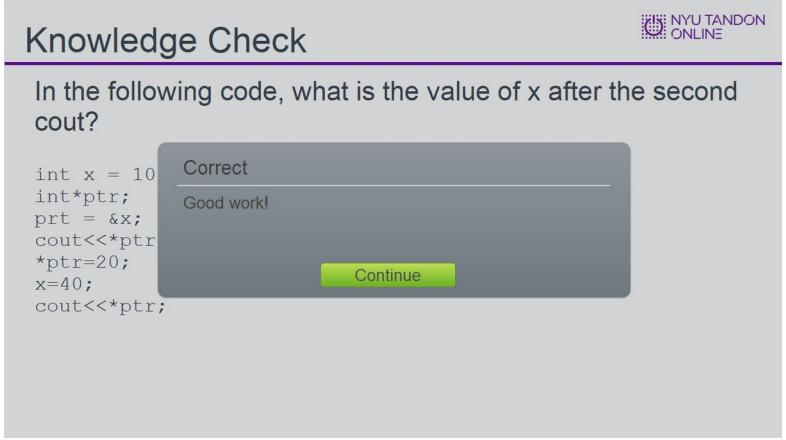


Correct	Choice	Feedback
	20	Note that the value of x was changed twice!
X	40	Good work!
	100	Note that the value of x has changed twice!

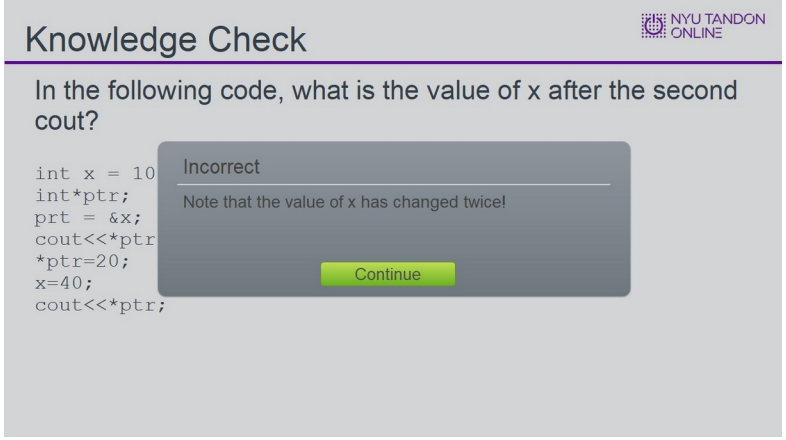
20 (Slide Layer)



40 (Slide Layer)

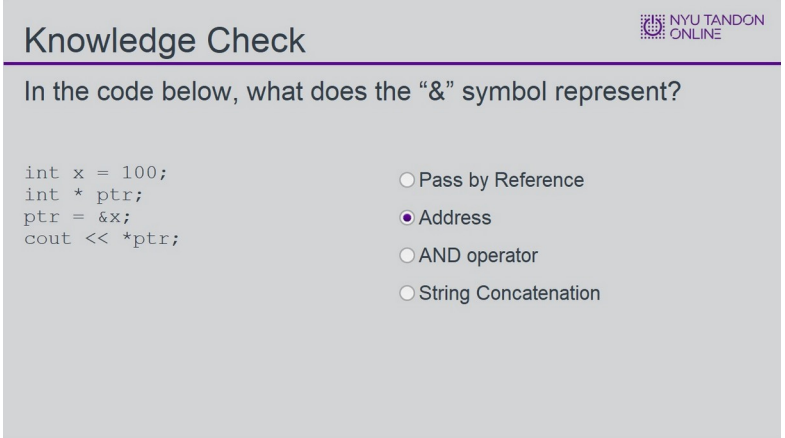


100 (Slide Layer)



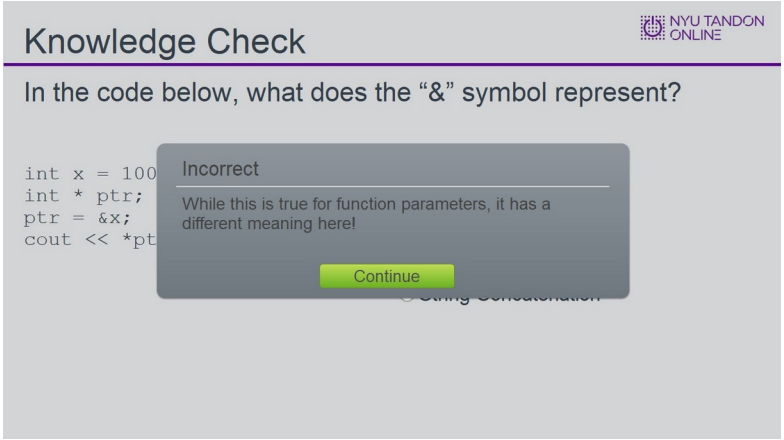
1.8 In the code below, what does the “&” symbol represent?

(Multiple Choice, 10 points, 4 attempts permitted)

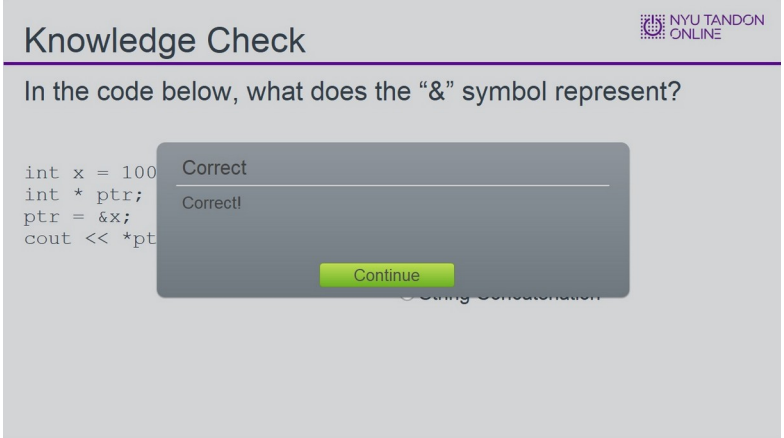


Correct	Choice	Feedback
	Pass by Reference	While this is true for function parameters, it has a different meaning here!
X	Address	Correct!
	AND operator	The AND operator is represented by “&&”
	String Concatenation	String concatenation can be done with the “+” symbol

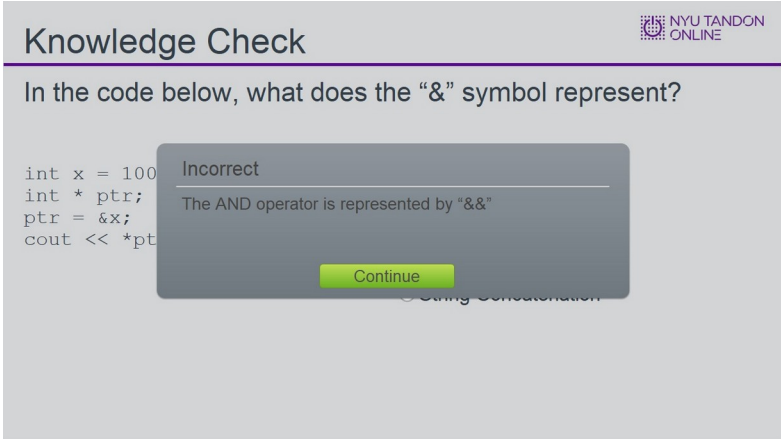
Pass by Reference (Slide Layer)



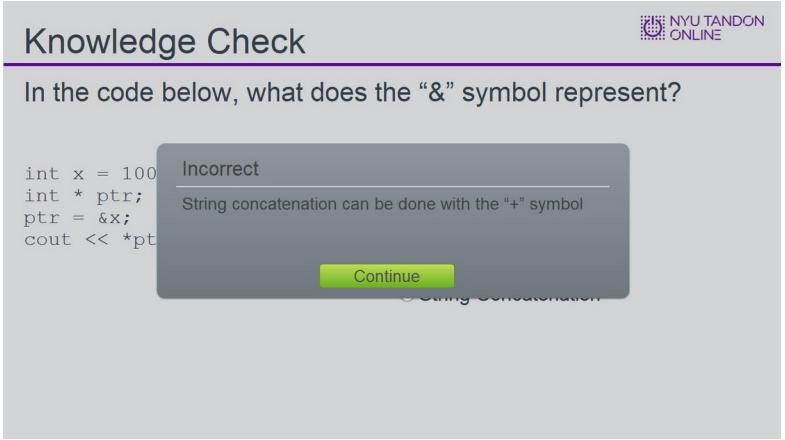
Address (Slide Layer)



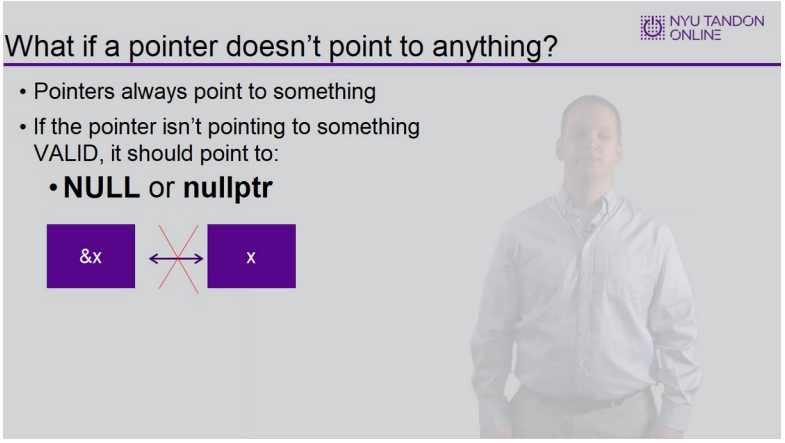
AND operator (Slide Layer)



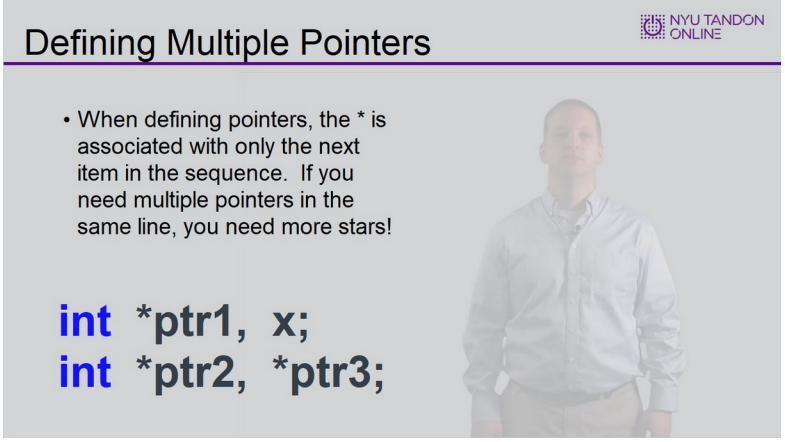
String Concatenation (Slide Layer)



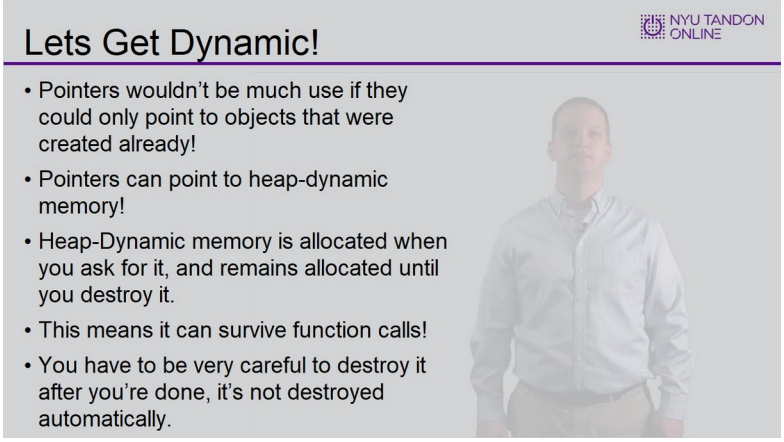
1.9 What if a pointer doesn’t point to anything?



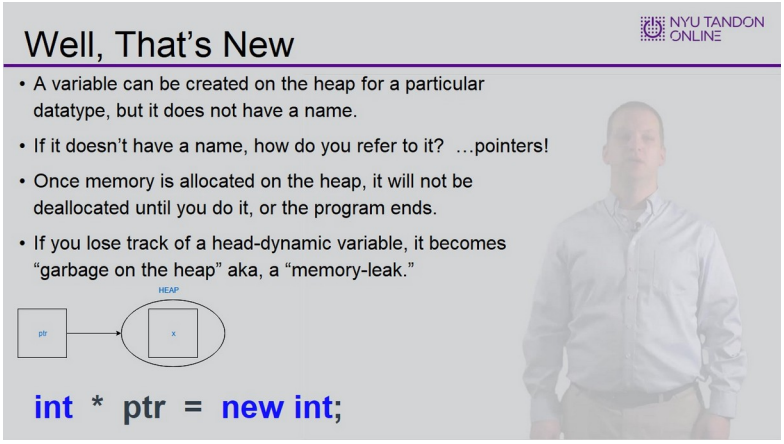
1.10 Defining multiple pointers



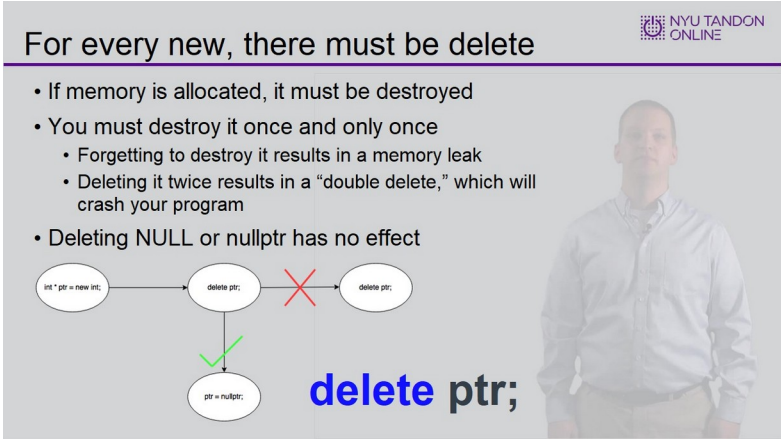
1.11 Lets get dynamic!



1.12 Well, that's new

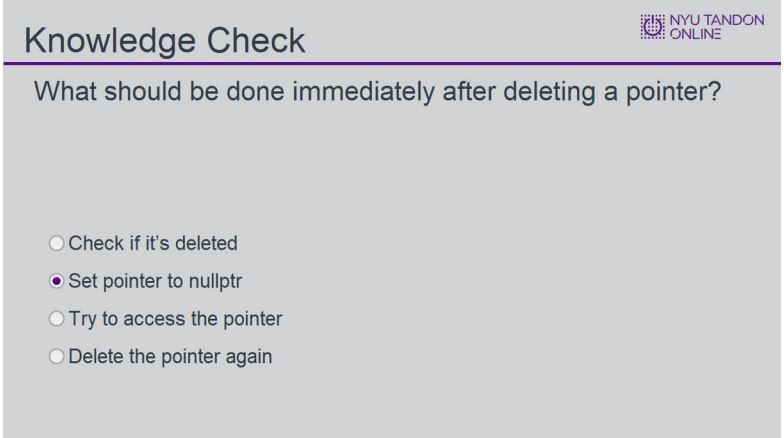


1.13 For every new, there must be delete



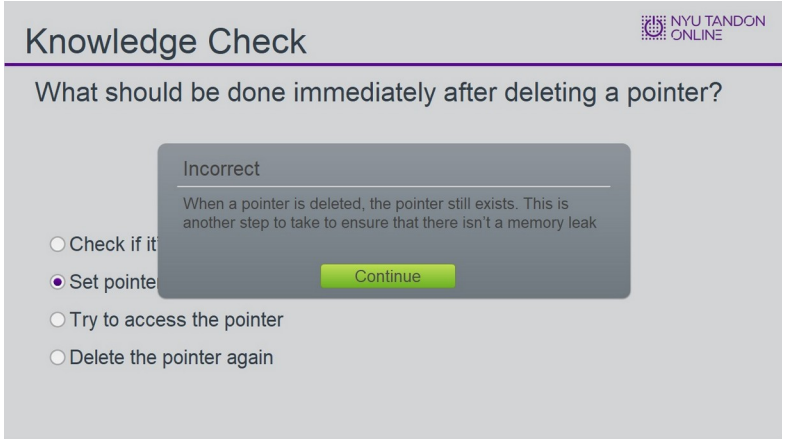
1.14 Knowledge Check

(Multiple Choice, 10 points, 1 attempt permitted)

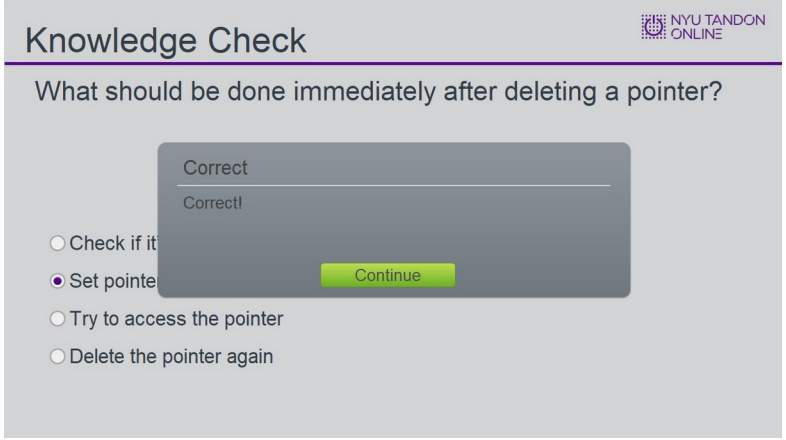


Correct	Choice	Feedback
	Check if it's deleted	When a pointer is deleted, the pointer still exists. This is another step to take to ensure that there isn't a memory leak
X	Set pointer to nullptr	Correct!
	Try to access the pointer	Once a pointer is deleted, there is no way to access it
	Delete the pointer again	Deleting a pointer after it has already been deleted will crash the program!

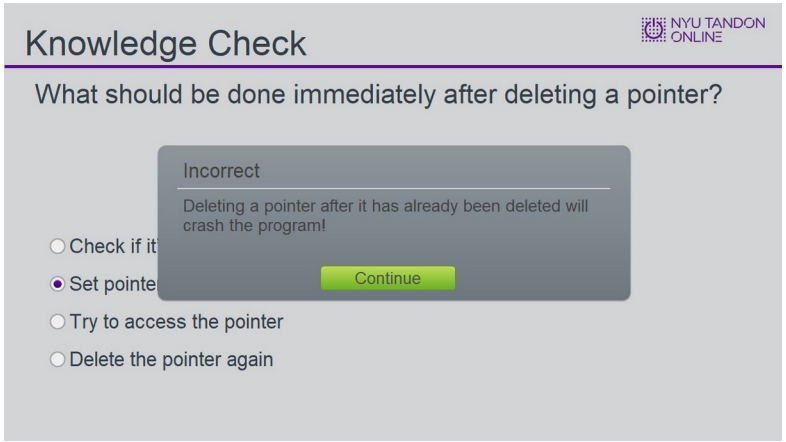
Check if it's deleted (Slide Layer)



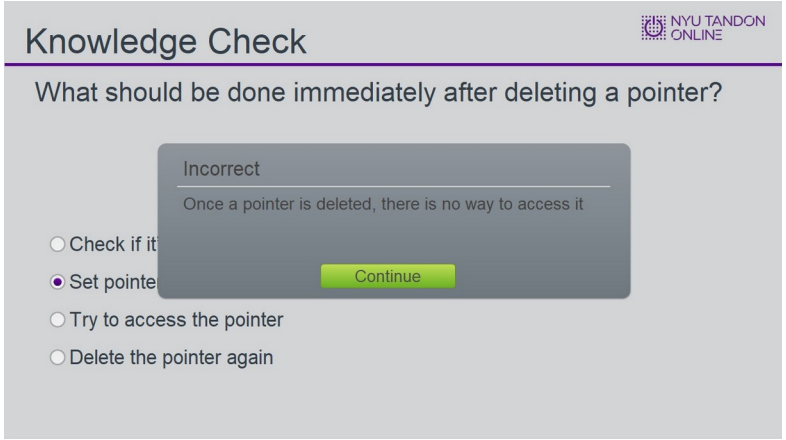
Set pointer to nullptr (Slide Layer)



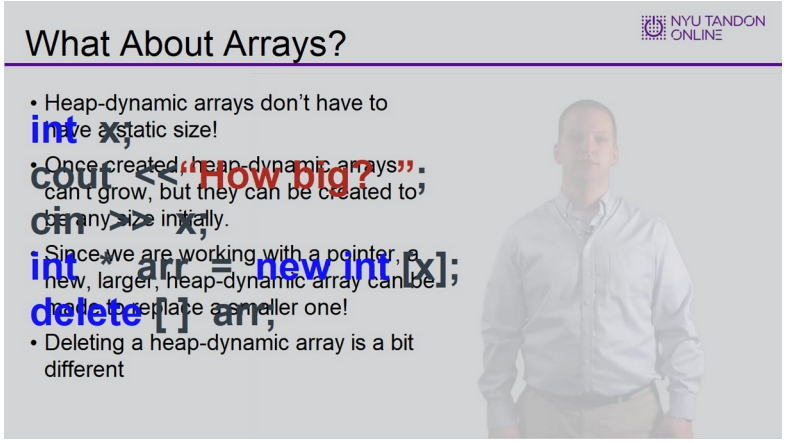
Delete the pointer again (Slide Layer)



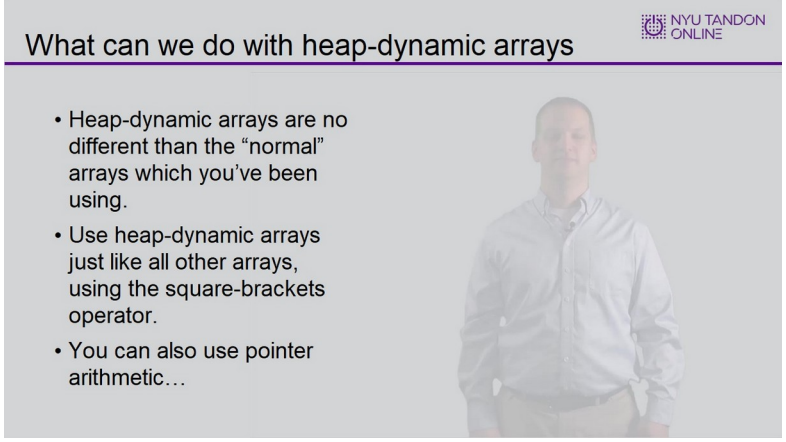
Try to access the pointer (Slide Layer)



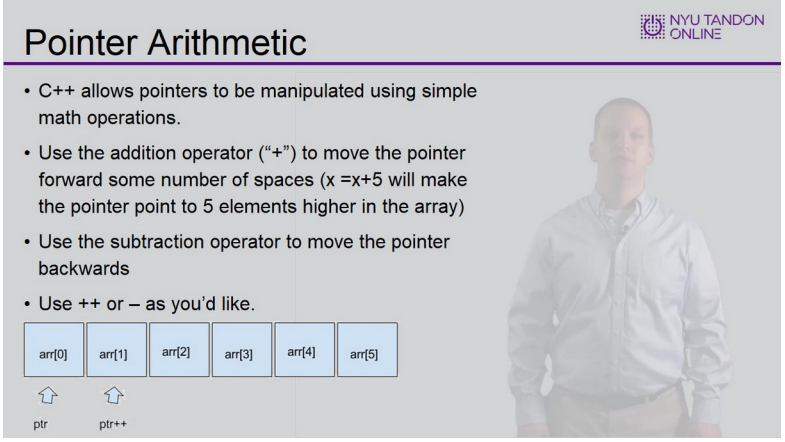
1.15 What about arrays?



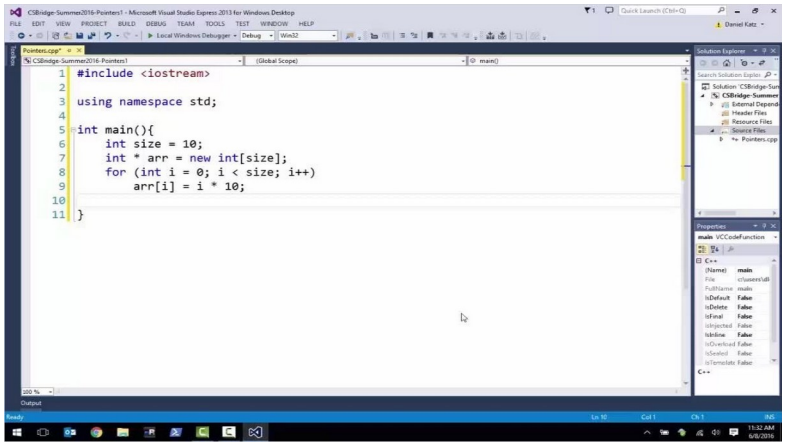
1.16 What can we do with heap-dynamic arrays



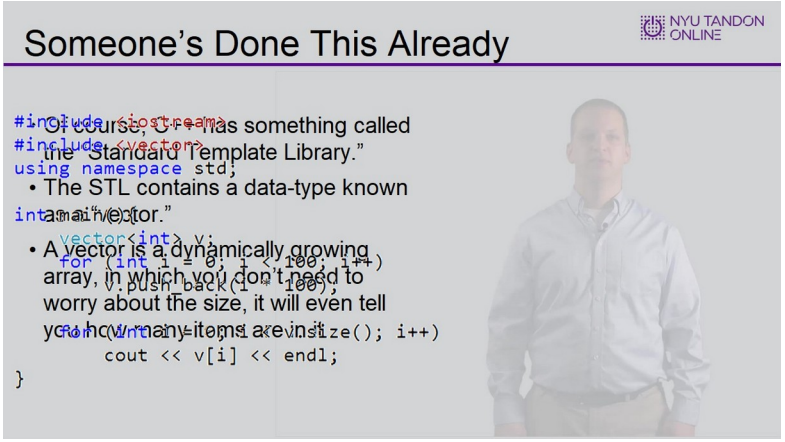
1.17 Pointer arithmetic



1.18 A real-example of a growing array

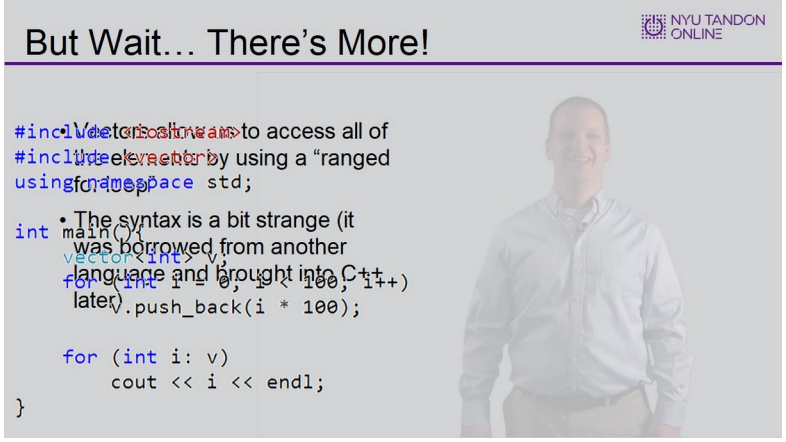


1.19 Someone's done this already

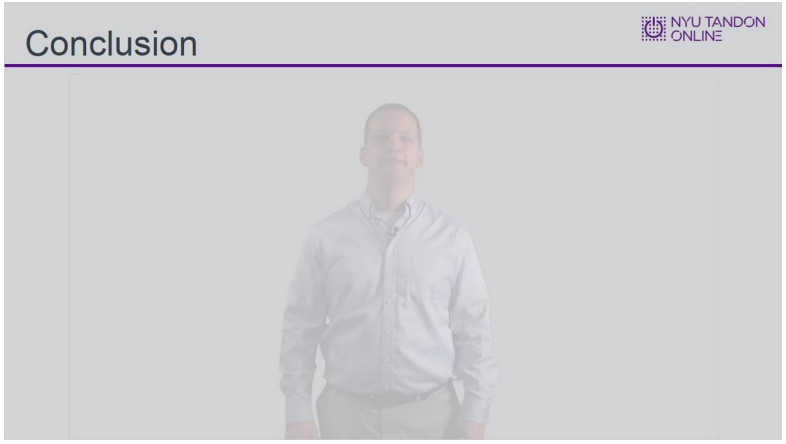


Notes:

1.20 But wait… there's more!



1.21 Conclusion



1.22 End of Module