Course: Data Structures in C++ | Module: Going Beyond The Basics

Lesson 2.3: Structs

https://codehs.com/course/12223/lesson/2.3

Description	In this lesson, students will be introduced to the idea of a **struct**. A struct is a basic C++ element that is used to store multiple data elements (members) in a single variable. Students will explore how to create and access structs in this lesson.
Objective	Students will be able to: • Create user-defined structs • Access and update struct member values • Understand when and where structs can be used
Activities	2.3.1 Video: Structs 2.3.2 Quiz: Quiz: Structs 2.3.3 Example: Defining and Accessing Structs 2.3.4 Example: Using Structs: Line Length 2.3.5 Exercise: Songs 2.3.6 Exercise: Movies - Part 1
Prior Knowledge	Students should be familiar with variable and variable types. If students have prior knowledge of Classes, you can use that to help make comparisons to structs, but it is not required.
Planning Notes	• There is a handout that accompanies this lesson. It can be used as an in-class activity or a homework assignment. Determine how and if this handout will be used and make the appropriate number of printouts prior to the class period. • This lesson starts a multi-lesson exercise called *Movies*. Make sure students complete the exercise successfully before moving on to the next lesson. • Data structures share similarities to classes that students may have seen in other courses. Consider taking time to discuss how structs are similar and different from other things they have seen in computer science.
Standards Addressed	
Teaching and	

Learning Strategies	**Lesson Opener:**
	
	• Have students brainstorm and write down answers to the discussion questions listed below. Students can work individually or in groups/pairs. Have them share their responses. [5 mins]
	
	Activities:
	
	• Walk through the lesson slides and complete the corresponding quiz. This quiz is a quick check for understanding [7-10 mins]
	• Explore the *Defining and Accessing Structs* example. [5-10 mins]
	  ◦ Have students create a new student struct and fill in their details.
	• Explore the *Using Structs: Line Length* example. [5-10 mins]
	  ◦ Have students pay attention to the order that the line and point structs are placed in the code. Have them try to switch this order and see what happens.
	• Complete the *Song* exercise. [15 mins]
	  ◦ Have students look back to the first example for help.
	  ◦ Students can come up with their own songs, so the exact output may vary.
	• Complete the *Movies - Part 1* exercise. [45-60 mins]
	  ◦ While the exact implementation may vaary, make sure that students consider a top-down approach and break the problem into functions.
	  ◦ Students will be building on this project, so excourage flexible code.
	
	Lesson Closer:
	
	• Have students reflect and discuss their responses to the end-of-class discussion questions. [5 mins]

Discussion Questions	**Beginning of Class:**
	
	• If we want to store multiple data elements, right now we can use a vector. What are some limitations of a vector?
	  ◦ *Vectors can only store data elements of the same type and typically store things as a list of the same type of element.*
	• From your experience with other languages, what other tools/data structures allow us to group and store collections of variable?
	  ◦ *Answers will vary depending on past experience, but Java users should think about classes becuase they hold instance variable. Python users may think about other commonly used structures such as lists and dictionaries.*
	• What is the difference between a private variable and a public variable when creating an object? How does this impact the way we access them?
	  ◦ *Private classes do now allow direct access to variable values. Instead they need to be updated via a function. Public access allows direct access to *
	
	End of Class:

	• Where are some places we might use a struct?
	  ◦ *Student answers will vary, for example structs can be used when we want to group items together, for example creating a student struct, shaape struct, etc. *
	• Given how we saw structs use, how are they similar to classes from other languages?
	  ◦ *The main similiarity is that they both store member variables and group these together.*
	• How are structs different to classes from other languages?
	  ◦ *The two main ways that the are different is that they are public by default and then generally only contain member variables, not fuctions.*

Resources/Handouts	Structs (Student) Structs (Teacher)

Vocabulary

Term	Definition

Modification: Advanced	Modification: Special Education	Modification: English Language Learners