Slide 1

The Glass Class

Glass Members

UML class diagrams should be language agnostic

For clarity, the pour function is implemented as pass by reference

Glass
<u>-pours : int</u> -volume : int
-amount : int
+Glass(a_volume : int, a_amount : int) +getVolume() : int
+getAmount() : int
+display() : void
<u>+getPours() : int</u>
+pour(source : Glass &) : void

UML class diagrams are meant to be independent of any specific programming language. That means that features that are specific to a given language are typically omitted from a UML diagram. For example, looking at the pour operation at the bottom of the diagram, we wouldn't typically show it as a pass by reference on a UML diagram, and instead would let C++ programmers choose pass by reference or pass by pointer; programmers using other languages would choose a passing technique appropriate for their language. However, to simplify and clarify this programming demonstration, we'll explicitly note that the pour function is implemented as pass by reference.

Slide 2

File Edit View Project Build Debug Test Analyze Tools Extensions Window	Help Sear	ch (Ctrl+Q)	P	chap.0	9	OB –	
🕴 © → © 📅 → 🍅 💾 📽 "기 → (" → Debug → x86 → ト Local Windows Debu	gger 🔹 🏓 🤷	÷				🖻 Live S	hare 🕅
Server Explorer			2		Sc Ce	Iution Explorer	▼
Toolbox					Build Solution Rebuild Solution Clean Solution Analyze and Code Cleanup Batch Build Configuration Manager Manage NuGet Packages fo Restore NuGet Packages New Solution Explorer View	CI r Solution	trl+Shift+B
					Calculate Code Metrics		
				Ŷ	Retarget solution		
	New Proje	:t			Add		٠
	Existing Pr	oject		ta	Add Solution to Source Con	itrol	
	New Item.	ed Site	Ctrl+Shift+A	- A	Paste Rename	G	trl+V
	Existing Ite	m on Folder	Shift+Alt+A	6	Open Folder in File Explorer Save As Solution Filter		
	Installation	Configuration File			Hide Unloaded Projects		
				1	Properties	A	lt+Enter
					So	olution Expl	Explorer
This item does not support previewing					↑ Ac	ld to Source Control	• 🐥 🔐

Text Captions

We begin by making a new project.

Slide 3



Text Captions

Make sure that the project is empty.

Slide 4

Image: Weight of the second	iew Proiect Ruild Debun Test Analyze Tools Extensions Window Heln Search(Ctrl+O)	chap.09	1	€ A	- 🗆	×
Server Explorer Toolbox	Configure your new project Empty Project C++ Windows Console Project name				rer • 4 × 1 • 7 • " h Explorer (9 • chap.09' (1 of 1 p nt ferences ternal Dependence	Notifications
	Glass1 Location D:\tmp\cs1410\Chap.09				ternal Dependenc exader Files source Files urce Files student.cpp	
\square This item does not	support previewing		Back Create	d to Source	Team Explorer	

Text Captions

Name the project "Glass1" (we'll write a slightly different version in the next section).

Slide 5

🕅 File Edit View Project Build Debug Test Analyze Tools Extensions Window	w Help Search (Ctrl+Q)	P chap.0) 📴 – 🗆 X
🖹 🛛 - 💿 🔯 - 🏠 🔛 🔐 🔊 - 🦿 - Debug - 🛛 x86 - 🕞 Local Windows Deb	ugger + 🛛 👼 🚽		🖻 Live Share 🕺
Servet Explorer Too			Solution Explorer ♥ # × Search Solution Explorer (𝒫 ♥ Search Solution 'chap.09' (2 of 2 p ♥ Glass1
lbox			Build Rebuild Clean View Analyze and Code Cleanup Project Only Retarget Projects Scope to This New Solution Explorer View Profile Guided Optimization Build Dependencies
	1 New Item	Ctrl+Shift+A	Add
	ta Existing Item	Shift+Alt+A	æ [™] Class Wizard Ctrl+Shift+X ∰ Manage NuGet Packages
	Reference tp Connected Service		 Set as StartUp Project Debug
	🎕 Class		Source Control
	* Resource		Cut Ctrl+X
			Paste Ctn+v Remove Del Rename
☐ Creating project 'Glass1' project creation successful.			Unload Project

Text Captions

Our next step will add a both a header and a source code file to the project.



🕅 File Edit View	Project Build De	bug Test Analyz	e Tools	Extensions	Window	Help	Search (Ctrl+Q)		P ch	ap.09		()B – 🗆 X
🕴 O + O 🏠 + 🖕 💾	Add New Item - Glas	551									?	×	🔓 Live Share 🛛 🕅
Ser	▲ Installed		Sort by	Default		• #	E		Search (Ct	rl+E)		ρ.	on Explorer 🔻 🖣 🗙 🦉
Server Explorer Toolbox	 Installed Visual C++ Code Formatting ATL Data Resource Web Utility Property She Test HLSL Graphics Online 	eets		-1)	p) .h)			Visual C++ Visual C++ Visual C++	Search (Ct Type: V Adds an	rl+E) isual C++ empty C++ class		- 0	on Explorer
	Location	er (mp (er Holen	ap 103 (0103						Diotident	Add	Cano	cel	
											Carro		on Expl Team Explorer
											î	Add to	o Source Control 🔺 🐥 💡

Text Captions

Choose "C++ Class" from the menu and name it "Glass." Doing this creates a header file with the Glass class specification started and a source code file – all in one step!

Slide 7

File Edit View Project Build Debug Te	st Analyze Tools Extensions	Window Help Search (Ctr	I+Q) P chap.	9 📵 – L X
Server Explorer Toolbox	Add Class Class name [5lass Base class Other options: Inline Managed	.h file Glass.h	.cpp file Glass.cpp	Solution Explorer Search Solution Explorer (Search Solution 'chap.09' (2 of 2 p Search Solution 'chap.09' (2 of 2 p
				🔶 🔶 🔺 📥 👘

Text Captions

The next pop up window allows us to change the name of the class and/or the names of the two files. We'll take the default values, so just press the "OK" button.

Slide 8

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) C + O C + O C + O C + O C + O D + C + Debug + x86 > Local Windows Debugger + D D + C + D C + O C + O D + C + D C + O C + O D + C + D D + D D + C + D D + D <th>♪ chap.09 *1 *1 ↓ ↓ ↓ ↓</th> <th>Image: Constraint of the second operation operat</th>	♪ chap.09 *1 *1 ↓ ↓ ↓ ↓	Image: Constraint of the second operation operat
5 6	Ln:2 Ch:1 TABS CRLF	 Header Files Glass.h Resource Files Source Files Source Files Student Header Files External Dependence Header Files Source Files Source Files Source Files Source Files Source Files Source Files Team Explorer
🗇 Ready	1	Add to Source Control 🔺 🐥 🚽

Text Captions

You can now see both file names in the "Solution Explorer" pane. Click on "Glass.h" in the Solution Explorer to open the file. We'll begin by filling out the class specification started by Visual Studio.

Slide 9

Dig File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) P chap.C	9	08 – 🗆 X
🔋 © + O 粉 + 🏠 🔛 🔐 🎐 - ペ + Debug 🔸 x86 🔹 🔸 Local Windows Debugger + 声 図 🛫 陆 佰 国 注 異 領 潮 潮 🖕		🖄 Live Share 🔗
Construction Construction<	■ ■ ■ ●	Live Share Solution Explorer * Solution Chap.09'(2 of 2 p) * Solution 'chap.09'(2 of 2 p) * Rebuild * Clean * Yow * Analyze and Code Cleanup * Project Only * Retarget Projects * Soup to This * New Solution Explorer View * Profile Guided Optimization * Add * Add * Ada * Class Wizard Ctrl+Shift+X Manage NuGet Packages * Set as StartUp Project * Debug *
	X	Source Control
	њ Л	Paste Ctrl+V
	×	Remove Del
	X	Rename
□ 150 % • © No issues tound In: 1 0		Unload Project

Text Captions

I often forget to make the new project the startup project, so let's do that now by right-clicking the project and selecting "Set as Startup Project."

Slide 10

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	08 – 🗆 X
🛛 🗢 + 💿 🏠 + 🏩 💾 🎾 - 🤆 + Debug - 🛛 x86 ト Local Windows Debugger + 声 🚳 🛫 ी 🌾 🧵 🦉 🗮 🧌 🦄 🦄 🛫	匠 Live Share 🕺
Glassh* • X SiGash I #pragma once I #pragma once I private: I private: I public: I public: I }; I Image: Signal	Solution Explorer Search Solution Explorer Search Solution 'chap.09' (2 of 2 p Gassh Factor Files Gassh Resource Files Gassch Resource Files Source Files Header Files Source Files Source Files Files Source Files Files Source Files Files Source Files Files Files Files Files Files Files Files Files Files Files Files Files Files Files Files F
C Ready	🛧 Add to Source Control 🔺 🐥

Text Captions

We'll need both a private and a public section.

Slide 11

🕅 File Edit View	Project Build Debug Test Analyze Tools Extensio	ns Window Help Search (Ctrl+Q) Vindows Debugger + 🔎 🙆 🛫 🔚 🖷 🗉 🧐	▶ chap.09	00 − □ × 112 Live Share
Server Explorer 1 2 1 2 1 2 1 2 1 2 1 2 1 2 6 7 8 9 10 1 0 0 0 0 0 0 0 0 0 0 0 0 0	<pre>#pragma once class Glass { private: static int pours; public: };</pre>		Lr: 5 Ch: 19 Coi: 23 TABS CRIF	Solution Explorer Solution Explorer Search Solution Explorer Search Solution Explorer Search Solution Chap.09 (2 of 2 p Search Solution Chap.09 (2 of 2 p Search Solution Chap.09 (2 of 2 p Search Solution Explorer Solution
🗇 Ready				↑ Add to Source Control ▲ 🐥

Text Captions

One of the challenges of the game is to minimize the number of times that we must pour water from one glass to another to solve the puzzle. There are a total three glasses, but only two glasses are involved during each pouring operation. This fact makes it impossible to track the number of times that water is poured from one glass to another with a member variable in each Glass object. So, our approach is to create a single variable that is shared by all three Glass objects. We do this by making the variable "static," which means that it is a class variable rather than an instance variable (that is, it's a variable that is owned by the class as a whole rather than a variable that is owned by a single object). The UML class diagram denotes class or static variables by underling them.

Slide 12

Kile Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) P chap.09 Co O O O	🕕 — 🗆 🗙 🖄 — 🗇 🔿
Guesh" • X Scient 1 #pragma once 2 class Glass 3 4 private: 5 static int pours; 6 7 int volume; 8 9 public: 10% 10%	Solution Explorer Solution Explorer Solution Explorer Search Solution Explorer Solution 'chap.09' (2 of 2 p Solution 'chap.09' (2 o
☐ Ready	↑ Add to Source Control ▲ 🐥 📲

Text Captions

From the results of the previous section, where we solved the problem and designed the program, we know that three values characterize each glass: the glass's total volume, the current amount of water in the glass, and the current amount of empty space in the glass. It's convenient to represent these values with member variables in the Glass class.

Given any two of these values, it's possible to calculate the third. So, our Glass class only needs two member variables to solve the pouring problem and I choose the total volume as the first.

Slide 13

Image: Weight of the second	Project Build Debug Test Analyze Tools Extensions Window ↓ ♥ • ♥ • ↓ Debug • x86 • ▶ Local Windows Debuge	Help Search (Ctrl+Q) ger • 🎜 🔕 🛫 🔚 🌾 🗉 😤 📕	우 chap.09	🕑 — [A Live Sha	ניא re א?
Glass.h* ↔ × Glass.h* ↔ × Glass1 1 2 3 4 5 6 7 8 9 10 11 12 13 10 10 10 10 10 10 10 10 10 10	<pre>#pragma once class Glass { private: static int pours; int volume; int amount; public: };</pre>		Ln: 8 Ch: 13 Col: 16 T	Solution Explorer Solution Explorer Search Solution Explorer Solution Crapper Solution Expl TABS CRLF	A x Notifications I of 2 p endenc s endenc s ppp
🗇 Ready				↑ Add to Source Control 🔺	Ļ.

Text Captions

For the second value or variable, I choose the current amount of water in the glass. While the choices are arbitrary, once they are made, the functions that follow rely on these variables. If we change the Glass member variables, then must also update the Glass functions.

Slide 14



Text Captions

Our next task is to develop the Glass member functions. I usually start with the constructor or constructors. We'll use the constructor to initialize the Glass's volume and the amount of water initially in the Glass as we instantiate each glass object.

Slide 15

Kile Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) P chap.09 Image: I	🕕 — 🗆 🗡 🖄 🖓 🕑
Gussh* • x Sidust 1 #pragma once 2 class Glass 3 { private: static int pours; 6 int volume; 10 public: 11 Glass(int a_volume, int a_amount) : volume(a_volume), amount(a_amount) {} 12 }; 13 14	olution Explorer
□ Ready Ac	dd to Source Control 🔺 🐥 📰

Text Captions

This constructor is a very simple function and is a perfect place to use an initializer list. Recall that an initializer list can only be used with a constructor, begins with a colon, and has one element for each member variable. Each element consists of the member variable's name and its initial value, which in this example, is one of the constructor's arguments. The initial value is enclosed with parentheses.

Once the member variables are initialized, there are no remaining tasks for the constructor to do. So, we end the list with a pair of empty braces, which is the function's body.

Slide 16



Text Captions

It's often convenient to have getter functions for some of the member variables and this is the case for the overall puzzle or game that we are creating. Getters are typically very simple functions as illustrated by getVolume.

Slide 17



Text Captions

A second getter function, getAmount, will allow users to see the amount of water currently stored in the Glass object.

Slide 18



Text Captions

The program really doesn't need a display function: we can get the same information from the two getter functions, but it is a convenient function to have.

Notice that Intellisense is flagging "cout" as an error. The standard mantra is, "When you use a feature, #include the corresponding header file."

Slide 19

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) P chap.09 Image: Im	📴 — 🗆 🗙 P <u>i</u> l Live Share R
Glassh* * X Biglisst • (Global Scope) 1 #pragma once 2 #include <iostream> 3 using namespace std; 4 5 5 class Glass 6 { 7 private: 8 static int pours; 9 int volume; 11 int amount; 12 </iostream>	Solution Explorer
	🛧 Add to Source Control 🔺 🐥 🔮

Text Captions

We correct the problem by adding the #include <iostream> directive and the using namespace statement at the top of the file, above the class specification. The red underlining will go away once we click the mouse at the end of line 17 or beyond.

Slide 20

🕅 File Edit View	v Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) 🔎 chap.09	0B − □ × I& Live Share &
File Edit View Image: Second state sta	<pre>v Project Build Debug Test Analyze Tools Extensions Window Help Search(Ctrl+Q) P chap.09 w Project Build Debug * x86 • ▶ Local Windows Debugger • # @</pre>	Image: Solution Explorer Image: Solutio
16 17 18	<pre>int getAmount() { return amount; } void display() { cout << amount << " / " << volume << endl; } static int getPours() { return pours; }</pre>	
19 20 21	issues found	roi E Solution Expl., Team Explorer
Ready		↑ Add to Source Control ▲ ♣

Text Captions

The function getPours is just another getter function. However, the pours variable is a class variable and not a member variable – or said another way, pours is a static variable. That means that the getter function must also be static. We'll see later how that impacts the way that we call the function.

Slide 21

Text Captions

Finally, we add the pour function, which is the only complex member function in the program. Since it is a larger function, we'll only prototype it in the class and define it in a separate source code or .cpp file.

Slide 22



Text Captions

The newest version of the Visual Studio editor has a neat feature that speeds navigation just a little. After finishing the prototype, the editor displays a little screwdriver on the left side – you may need to wait just a moment for it to appear. If we click on the down-arrow next to the screwdriver, we'll see two options: one option is to copy the function's signature or prototype to the clipboard, which we can use to begin defining the function elsewhere. The other option is to outline the function in Glass.cpp. Let's select that option.

Slide 23

K Fi	ile Edit View	Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	D B – D ×
G •	· 💿 🛛 📸 🖌 🍅 🖡	💾 🚰 🎔 - 🤆 - 🛛 Debug 🔹 x86 🔹 - 🕨 Local Windows Debugger - 🍺 🖾 🛫 🔚 🌆 👷 🔚 🐄 🦄 🦄 🖕	🖻 Live Share 🛛 🕅
Server Explorer Toolbox	● 記・企 ass.h* + × Glass1 8 9 10 11 12 13 14 15 16 17 18 19	<pre>builty of the builty of t</pre>	Live Share Vertical Sector Vertical Sector
150	20	>; issues found + b (n: 19 Ch: 27 Col: 30 TARS CDIF	Solution Expl Team Explorer
☐ Read	dy		Add to Source Control 🔺 🐥 💡

Text Captions

It looks like this little yellow window is opening in Glass.h, but notice the line numbers and the name Glass.cpp on the tab. Visual Studio has written this outline of the pour function in Glass.cpp. While we can edit the function in the yellow window . . .

Slide 24

🕅 File Edit View	Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	0 ₿ − □ ×
🔆 😋 • 😂 🏠 • 🖕	💾 🔐 🦻 • 🤆 • 🗍 Debug 🔹 x86 🔹 🔹 Local Windows Debugger • 🏓 🙆 🛫 🔚 🌾 📋 🦉 📕 🧌 🧌 👘 🧤	🖻 Live Share 🛛 🕅
Seven Glass.h* +>× IO 11 12 13 14 15 16 17 18 19	<pre></pre>	Solution Explorer Search Solution Explorer Search Solution Explorer Search Solution Chap.09' (2 of 2 p Search Solution 'chap.09' (2 of 2
20	};	
150 % - ⊘ No	issues found	CRLF Solution Expl Team Explorer

Text Captions

I prefer - and that's all it is, a personal preference – to edit the function in a "normal" editor window. So, click the "x" button on the Glass.cpp tab to close the yellow window.

Slide 25

File Edit View Project Build	Debug Test Analyze Tools Extensions	Window Help Search (Ctrl+Q)	Chap.09	0 3 − □ ×
	P Debug • x86 • Debug • Local Wind	lows Debugger 🔹 🍺 🛛 💭 🛫 🖄 🖓 📋 🖫 🖉	■ 31 31 41 .	🔓 Live Share 🛛 🛱
Glass.cp* + × Glassh* Glass1 Glass1 Finclude Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Colorer Color	- → Glass "Glass.h" ss::pour(Glass& source)	● pour(Glass & s	purce)	Solution Explorer Solution Explorer Search Solution Chap.09' (2 of 2 p) Search Solution 'chap.09' (2 of 2 p) Solution Expl CRLF Solution Expl CRLF
🗇 Ready				🛧 Add to Source Control 🔺 🐥 📑

Text Captions

Find Glass.cpp in the solution explorer and double-click the name. This opens the file in a standard editor window, where we can see the function outline created just few moments ago. Notice the class name, "Glass," followed by the scope resolution operator on line 3. This is how the compiler knows that this function is a part of the Glass class.

This function will pour or transfer water between two glass objects: the source and the destination. The role played by each glass object is determined by its position the function call:

```
destination.pour(source);
```

Slide 26

Kile Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) P chap.09 Image: Im	0B — 🗆 🗙
Guss: • • • Gass 1 #include "Glass.h" 3 • • • Gass • • void Glass::pour(Glass& source) 6 7 8	Solution Explorer Solution Explorer Solution Explorer Search Solution Explorer Search Solution Explorer Search Solution 'chap.09' (2 of 2 p Search Solution 'chap.09' (2 of 2 p Search Solution 'chap.09' (2 of 2 p Search Solution Explorer Search Solution Explorer Search Solution Explorer Search Solution Explorer Solution Explore Solution Expl. Team Explorer Solution Expl. Team Explorer Solution Expl. Solution
☐ Ready	↑ Add to Source Control ▲ ♣

Text Captions

Lest we forget, there is one task that we should do before completing the pour function. We need to initialize the static or class variable pours. On line 3, this looks very much like a dreaded global variable, but the class name and scope resolution operator tie this variable solidly to the Glass class. That is, it reduces the variable's scope to the Glass class.

This example demonstrates the syntax for initializing a static variable in C++. Java has a somewhat cleaner syntax for doing this.

Slide 27

Image: Search (Ctrl+Q) P Chap.09 Image: Search (Ctrl+Q) Image: Search (Ctr	00 − □ × & Live Share
Glass @ pour(Glass & source) 1 #include "Glass.h" 3 int Glass::pours = 0; 4 > void Glass::pour(Glass & source) 6 { 7 pours++; 8 } 9	 Solution Explorer Solution Explorer Search Solution Chap.09 (2 of 2 p) Solution 'chap.09 (2 of 2 p) S
□ Ready	🛧 Add to Source Control 🔺 🐥 💡

Text Captions

Our goal is to minimize the number of times that we must pour water from one glass to another. So, we must count each pour operation, which we do by incrementing the pours counter whenever the pour function runs.

Slide 28

🕅 File Edit View	Project Build Debug Test Analyze Tools Extensions Win	dow Help Search (Ctrl+Q)	P chap.09		08 – 🗆	×
G • O 🏠 • 🖕	💾 🗳 🍫 🖓 - 🖓 - Debug - x86 - 🕨 Local Windows I	Debugger 🗸 🏓 🕼 🖕 🔚 🖷 🗍 🖫 🖉 📕	영 해 제 -		🖻 Live Share	ጽ
Sever (Glass.cpp* +> × ⊂ (Glass.1 1 2 5 6 7 8 9 10 11 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0	<pre>issues found #include "Glass.h" int Glass::pours = 0; void Glass::pour(Glass& source) { pours++; int space = volume - amount; } </pre>	■ pour(Glass & sour	се) 	 Sol Sel Sel	ution Explorer ♥ ♥ ♥ ♥ ♠ ↓ ♥ ♥ ↓ ♥ ♥ ♠ ↓ ♥ ♥ ↓ ♥ ♥ ↓ ♥ ♥ ↓ ♥ ♥ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓ ♥ ↓	Notifications
🗇 Ready				↑ Ad	d to Source Control 🔺 🛛 🦂	V

Text Captions

The rest of the pours function can be divided into two parts:

- First, calculate the maximum amount of water that can be poured or transferred from the source glass to the destination glass, and
- Second, update the two Glass objects to complete the pouring operation.

Both problems were solved in the previous section, and the solution implemented here begins by calculating how much space is available in the destination glass.

Slide 29



Text Captions

The next step is calculating how much water to pour or transfer from the source to the destination glass. We can't pour more water than there is in the source glass, nor can we pour more water than there is space available in the destination glass. So, the amount that we can pour is the smallest or minimum of the space in the destination and the amount in the source.

C++ has a library function named min that we can use to select the smallest or minimum of these two values.

Slide 30

Kile Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) P chap.09 G • O 1 2 • C • Debug • x86 • Local Windows Debugger • 月 回 法 信 注 注 用 知 개	oB ー ロ × Live Share を
Glusscepp' • X Glush* (GlubalScope) 1 #inelude "Glass.h" 2 #inelude "Glass.h" 3 using namespace std; 4 int Glass::pours = 0; 6 - 7 void Glass::pour(Glass& source) 8 { 9 pours++; 10 int space = volume - amount; 11 int transfer 13 } 14	Solution Explorer Solution Explorer Search Solution Explorer Search Solution Explorer Search Solution 'chap.09' (2 of 2 p Search Search Solution 'chap.09' (2 of 2 p Search Sear
☐ Ready	↑ Add to Source Control ▲ 🐥 📰

Text Captions

To use the min function, we need to #include the <algorithm> header file and add the using namespace statement.

Slide 31

Kile Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) P chap.09 Image: I	0B − □ × 1& Live Share
Totascept" + X Outsought * X Outsource Outso	Solution Explorer Solution Explorer Solution Chap.09' (2 of 2 p Solution 'chap.09' (2 of 2 p
🗇 Ready	🏫 Add to Source Control 🔺 🐥 🔡

Text Captions

With the min function, it's easy to calculate the amount of water to transfer from the source to the destination.

Slide 32

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	0 B – 🗆 X
🖉 🗢 💿 📅 📲 💾 🎾 🕈 🖓 👻 🕐 🕐 Debug 🔹 x86 🔹 🕨 Local Windows Debugger 🗸 🎜 🔯 🚽 🎼 🍯 🦉 📕 🌾 🐂 👘	🖻 Live Share 🛛 🖗
State • Glass % Glass • Power(Glass & source) 1 ##include "Glass.h" 2 #include "Glass.h" 3 using namespace std; 4 int Glass::pours = 0; 6 • void Glass::pour(Glass& source) 8 { 9 pours++; 10 int space = volume - amount; 11 int transfer = min(space, source.amount); 13 amount += transfer; 16 } 17 * Onessectord	Solution Explorer • 4 × Search Solution Explorer • 4 × Search Solution 'chap.09' (2 of 2 p Search Solution 'chap.09' (2 of 2 p • Solution Chap.09' (2 of 2 p • Solution Cha
🗌 Ready	🛧 Add to Source Control 🔺 🐥 🔮

Text Captions

The transfer amount is added to the destination glass and subtracted from the source glass. The pour function is now complete.

Slide 33

🕅 File Edit View Project Build Debug Test Analyze Tools Extensions Window He	Ip Search (Ctrl+Q)	9 <mark>08</mark> – 🗆 X .
😸 😋 🔹 💿 🔤 📽 🗳 🤔 🗳 🗸 💎 💎 💎 Debug 🔹 x86 🔹 🕨 Local Windows Debugger 🔹	♬ @_;》旨作 ℡℡ ♥判判;	년 Live Share 🖉
Image: Second); New Item Ctrl+Shift+A New Filter Reference Connected Service	Solution Explorer # X Solution Explorer # X Solution Explorer # X Solution Explorer # X Solution Chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) Solution Chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) Solution Chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) Solution Chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) Solution Chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) Solution Chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) View Solution 'chap.09' (2 of 2 p) Project Only N Project Only N Retarget Projects Scope to This Scope to This New Solution Explorer View Profile Guided Optimization N Build Dependencies N Manage NuGet Packages Ctrl+Shift+X Manage NuGet Packages Set as StartUp Project Debug N
	Resource	Jource control // X Cut Ctrl+X Deste Ctrl+V
150 % - O Noiscus found	in 15 Ch 20 C	Paste Ctrl+V Remove Del Rename
Ready	μπ. 15 CM: 28 C	Unload Project

Text Captions

The next step is adding a main function that implements the game by using the finished Glass class. Right-click the project, select "Add" and "New Item...."



- O 🕅 - 🔄 🖬 Add New Item - G	ass]	Sauch (Cirl. D	? X 🖄 Live Share &
Slass.cpp* +⊨ × Glass. = Instaned	Sort by: Default	Search (Ctrl+E)	on Explorer 👻 🕂 🗙
A Glass1 1 2 1 2 3 4 5 5 6 7 7 6 7 7 8 9 10 11 12 13 14 15 16 17 4 Visual C++ Code Formattin ATL Data Resource Web Utility Property S Graphics P Online	heets	Visual C++ Visual C++ Visual C++ Visual C++	source code Solution Explorer () ~ " Solution 'chap.09' (2 of 2 p Glass1 • References External Dependence Header Files > M Glass.h Resource Files > *+ Glass.cpp Surce Files > *+ Glass.cpp Surce Files = External Dependence = References = External Dependence = Resource Files = Source Files
Name:	game		
10.55	D) to a 1410/ Chan 00/ Class1	Preven	

Text Captions

Create a C++ source code file named "game" and press the "Add" button.

Slide 35

🕅 File Edit View Project Build Debug Test Analyze Tools Exte	sions Window Help Search (Ctrl+Q)	P chap.09	08 – 🗆 X
🕴 🗢 🔹 🛍 📽 😕 - 🖓 - 🦿 - Debug 🔹 x86 🔹 🕨 Lo	al Windows Debugger 🗸 🎜 🛛 🖕 🔚 🖷 🗉 🖄	📕 গা শা 🗸	🖻 Live Share 🛛 🕅
game.cpp + X Glass.trp* Glass.h* Siglass1 • (Global Scope) 1 • Glass.h* 10000 • No issues found		Ln: 1 Ch: 1 TABS CRUE	 Solution Explorer Solution Explorer Search Solution Chap.09' (2 of 2 p) Solution 'Chap.09' (2 of 2 p) Solution 'Chap.09' (2 of 2 p) Solution 'Chap.09' (2 of 2 p) Soluce Files Header Files Glass.1 Header Files Glass.1 Source Files Team Explorer
🗇 Ready			🛧 Add to Source Control 🔺 🐥 🔡

Text Captions

We'll place the rest of the game code in main in the game.cpp file.

Slide 36

K File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	08 – 🗆 X
🧯 😋 🕶 🗧 🔐 🗳 🥊 🥙 🎔 + 🔇 + 🛛 Debug 🔹 x86 🔹 🔸 🕨 Local Windows Debugger + 🎜 🙆 🛫 陆 🎼 🗏 🦉 📕 🎕 🖄 🦄 🛫	।岔 Live Share 🛛
genec.cpt + X Gasc.ht* StGiss1 • (Global Scope) • @ main() 1 •#include <iostream> 2 •#include <iostream> 3 using namespace std; 4 • 5 •int main() { • 7 * 8 return 0; 9 }</iostream></iostream>	 Solution Explorer Solution Explorer Search Solution 'chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2
🗇 Ready	🛧 Add to Source Control 🔺 🐥 🔮

Text Captions

This is basically our standard starting code with the addition of the #include "Glass.h" directive.

Slide 37



Text Captions

The game requires three instances of the Glass class – that is, three Glass objects. Two of the glasses, the 3- and 5-ounce, are initially empty; the 8-ounce glass is initially full. As discussed in the previous, problem solving section, we can simplify some of the operations if we use an array of Glass objects rather than three separate objects. Notice that the glasses are created on the stack and that the Glass constructor is called for each object.

Slide 38



Text Captions

The game continues until at least one of the three glasses contains four ounces of water. One glass has a maximum volume of three ounces, so it isn't considered in the while-loop test.

Slide 39

Kile Edit View Project Build Debug Test Analyze Tools Extensions Windows Help Search (Ctrl+Q) P chap.09 Image: Comparison of the state of the	
<pre>gemecpp' e X Gesc.pp' Gesc.h' %Glass1 - (Global Scope) - @ main() 1 =#include <iostream> 2 #include "Glass.h" 3 using namespace std; 4 5 = int main() 6 { 7 Glass glasses[3]{ Glass(3,0), Glass(5,0), Glass(8,8) }; 8 9 = while (glasses[1].getAmount() != 4 && glasses[2].getAmount() != 4) 10 { 11 = for (int i = 0; i < 3; i++) 12 { 13 cout << "Glass " << i + 1 << ": "; 14 glasses[i].display(); 15 } 16 } 17 return 0; 19 } 10 }</iostream></pre>	Solution Explorer Search Solution Explorer Search Solution Explorer Search Solution 'chap.09' (2 of 2 p Solution 'chap.09'
150% • O No issues found 4 b Ln: 15 Ch: 4 Col: 10 TABS CRLF S	Solution Expl Team Explorer
🗖 Ready 🔷 🔿	Add to Source Control 🔺 🐥 🔐

Text Captions

To help the player decide how to take the next step, at the beginning of each move, we display the current state of the game – that is, the amount of water in each glass.

Slide 40

K File Edit View Projec	t Build Debug Test Analyze Tools Extensions V ♥ - ♡ - ↓ Debug - x86 - ▶ Local Window	Window Help Search (Ctrl+Q) ws Debugger → ♬ @ _ % 旨 @ 雪 ?雪 ₽	우 chap.09	03 − □ × Live Share R
Seven cop* * Class.cop* game.cop* * X Glass.cop* 1 - #ir 2 % Glass1 - #ir 3 usi 3 usi 4 - - 6 { - - - 7 8 9 - - 10 11 - - - 12 13 - - - 14 15 - - - 18 19 - - - 20 21 - - -	<pre>Glass.h*</pre>	<pre> • @ main() Glass(5,0), Glass(8,8) }; 4 && glasses[2].getAmount 1 << ": "; ., 2, or 3; or enter 4 to q </pre>	:() != 4) uit>: ";	 Solution Explorer Solution Explorer Solution Explorer Search Solution 'chap.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) <li< th=""></li<>
150 % - O No issues for	ind 4	•	Ln: 20 Ch: 22 Col: 28 TABS CI	RLF Solution Expl Team Explorer

Text Captions

The next step allows the player to choose to which glass the water is poured. It's convenient to label the glasses with counting numbers – that is 1, 2, or 3 – rather than starting with 0, as that is likely more familiar to most people.

Slide 41

Text Captions

As suggested by the prompt, we also allow the player to end the game by entering the value 4 and implement the early exit with an if-statement.

Slide 42

Text Captions

Next, we allow the player to choose from which glass the water is poured. Again, we label the glasses as 1, 2, or 3, and allow the player to end the game early by entering 4.

Slide 43

Ø	File Edit View	Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) → C → Debug → x86 → ► Local Windows Debugger → 🚮 🔯 🛬 👘 🐨 🐨 20 🖬 🐄 20 -	00 - 🗆 X
S I			- Solution European - 1 X - 2
iver	Glass1	v (Global Scope) v @ main()	
Explo	15	}	+ Search Solution Explorer (0 +
irer	16		Solution 'chap.09' (2 of 2 p
Tool	17	int destination;	▲ 🖪 Glass1
box	18		Image:
	19	cout << "Pour TO glass: <1, 2, or 3; or enter 4 to quit>: ";	 Kenner Sependene Files
	20	<pre>cin >> destination;</pre>	Glass.h
	21		Source Files
	22	if (destination == 4)	♦ 🕶 game.cpp
	23	exit(0);	♦ ++ Glass.cpp
	24		▶ ∎ References
	25	int source;	External Dependence
	26		Resource Files
	27	cout << "Pour FROM glass: <1, 2, or 3; or enter 4 to quit>: ";	🔟 🔺 🚛 Source Files
	28	<pre>cin >> source;</pre>	♦ *+ student.cpp
	29		
	30	if (source == 4)	
	31	exit(0);	
	32		
	33	if (source > 0 && source <= 3 && destination > 0 && destination <= 3)	
	34	<pre>glasses[destination - 1].pour(glasses[source - 1]);</pre>	
	35	else	
	36	<pre>cout << "0 < destination <= 3 AND 0 < source <= 3" << endl;</pre>	• • • • • •
	150 % 🔹 🗿 No is	ssues found + Ln: 36 Ch: 63 Col: 72 TABS CI	RLF Solution Expl Team Explorer
🗇 R	Ready		🛧 Add to Source Control 🔺 🐥 📑

Text Captions

Anytime that user input is used as an index into an array, it must be validated – this is an important security issue: indexing an array out of bounds causes a buffer overrun or a buffer overflow. What happens depends on several factors that are completely beyond our control. In the worst-case scenario, nothing bad happens while we are testing our code, but later, the program can crash, or it can leave a vulnerability that a bad actor can exploit to coopt or infect the system.

A simple if-statement can check the user input and ensure that it represents a valid array index. Assume that the index is valid and focus your attention on line 34. To correctly index both arrays, the program subtracts 1 from each of the user input values. This step is done because we labeled the glasses with counting numbers: 1, 2, and 3. But C++ arrays are zero-indexed, so, for an array of three objects, valid index values are 0, 1, and 2.

Still looking at line 34, once two of the three objects are identified in the array, the pour function is called, which pours or transfers water from the glass object inside the parentheses to the glass object appearing to the left of the dot operator. This completes the code inside the while-loop.

Slide 44



Text Captions

Solving the puzzle ends the while-loop and the program prints the total number of pour operations taken to solve the puzzle. Recall that the getPours function is static – line 40 illustrates the preferred way to call a static function – the class name, the scope resolution operator, and then the function name.

Slide 45

K	File Edit View	Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) 👂 chap.09 📴 - 🗆 🗙
8	G • 🛛 🕆 🟠 • 🏠 🛛	📲 🚰 🎔 - 💎 - Debug - x86 - 🕨 Local Windows Debugger - 🎜 🔯 🛫 🔚 🏗 🖫 📲 🕅 👘 👘 - 🖓 - 🖄 - 🕅 -
Server E	game.cpp* ⊕ 🗙 G 🖫 Glass1	lass.cpp* Glass.h* ▼ Solution Explorer ▼ - + × Global Scope) ▼ Ø main() ▼ ○ ○ ☆ @ - * *
xploi	22	if (destination == 4)
er	23	exit(0);
Toolb	24	▲ 🖏 Glass1
XOO	25	int source;
	26	🔺 🚛 Header Files
	27	cout << "Pour FROM glass: <1, 2, or 3; or enter 4 to quit>: ";
	28	cin >> source;
	29	b ++ game.cpp
	30	if (source == 4)
	31	exit(0); ▷ •■ References
	32	External Dependenc Header Files
	33	if (source > 0 && source <= 3 && destination > 0 && destination <= 3)
	34	<pre>glasses[destination - 1].pour(glasses[source - 1]);</pre>
	35	else
	36	<pre>cout << "0 < destination <= 3 AND 0 < source <= 3" << endl;</pre>
	37	}
	38	
	39	<pre>cout << "\n\nYou solved the puzzle in " <<</pre>
	40	<pre>Glass::getPours() << " pours" << endl;</pre>
	41	
	42	for (int i = 0; i < 3; i++)
	43	glasses[i].display();
	150 % 🔹 📀 No	issues found + Ln: 43 Ch: 24 Col: 30 TABS CRLF Solution Expl Team Explorer
	Ready	↑ Add to Source Control ▲ ♣

Text Captions

Finally, the program prints the final state of the game – that is, the amount of water in all three glasses.

Slide 46

🕅 File	Edit Vie	w Project Ruild Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	08 – 🗆 X
G • O	1 2 - 🎓	💾 🚰 🤊 - 🔇 - Debug - 🔹 x86 🕨 Local Windows Debugger - 🎜 🙆 🛫 🔚 🏦 🖫 🖫 🕅 🦏 🦄 🦄	🖻 Live Share 🛛 🕅
game.c B Glass Server Explorer Toolbox	initial initial initial <	Glass Save All (Ctrl+Shift+S) (Global Scope) (Global Scope)	Solution Explorer Search Solution Explorer Gass1 Gass1 Header Files Gass.h Resource Files Carlot Solution Files Carlot Solut Solution Files Carlot Solution Files Carlot Solu
	28 29 30 31 32 33 34 35 26	<pre>if (source == 4) exit(0); if (source > 0 && source <= 3 && destination > 0 && destination <= 3) glasses[destination - 1].pour(glasses[source - 1]); else cout << "0 < destination <= 3 AND 0 < source <= 3" << end];</pre>	 ▲ Source Files > +* game.cpp > ** Glass.cpp ▲ Glass.cpp ▲ Glass.cpp ▲ References ▲ External Dependence ₩ Header Files ₩ Resource Files ▲ Source Files ▲ Source Files ▲ + student.cpp
	36 37 38 39 40 41 42	<pre>cout << "0 < destination <= 3 AND 0 < source <= 3" << end1; } cout << "\n\nYou solved the puzzle in " << Glass::getPours() << " pours" << end1; for (int i = 0; i < 3; i++) classes[i] disclass(); </pre>	
150 %	43	glasses[1].dlsplay(); loissues found	Solution Expl Team Explorer
Ready			↑ Add to Source Control 🔺 🐥

Text Captions

Save all the files by clicking the button with the picture of two floppy disks.

Slide 47



Text Captions

Build the project.

Slide 48

K File Edit View ⊙ • ⊙ 🏠 • 🏠 I	Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) → Chap.09 → Local Windows Debugger → 月 回 → 1 → 1 → 1 → 1 → 1 → 1 → 1 → 1 → 1 →	📴 − □ × ເ≩ Live Share 🕅
game.cpp p Cit ™ Glass1 22 23 ™ Glass1 24 24 22 26 27 28 29 30 31 32 33 33 34 35	<pre>sscep Glassh</pre>	 Solution Explorer Search Solution Explorer () Search Solution Cxpp.09' (2 of 2 p) Solution 'chap.09' (2 of 2 p) Solution 'chap
150% No Output Show output from: 1>come.cpp 1>Glass.cpp 1>Glass.trockpro: 1>Glass.trockpro: 1>Glass.trockpro:	ssues found → Ln: 43 Ch: 24 Col: 30 TABS Build → Started: Project: Glass1, Configuration: Debug Win32 ie -> D:\Cmp+cs1410\Chap.09\Debug\Glass1.exe 1: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ===================================	CRLF

Text Captions

The program builds without any errors.

Slide 49



Text Captions

Start the program running.

Slide 50

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	08 – 🗆 X
🕈 😋 • 💿 摺 • 🏠 🗳 🗳 🥐 • 🖓 • 🖓 • 🖓 • Debug 🔹 x86 🔹 🕨 Local Windows Debugger • 声 🙆 🖓 🛬 🏦 国 🤹 🐂 🕅 🖕	년 Live Share 🕅
Image: Solution of the second seco	Live Share Live
☐ Ready	dd to Source Control 🔺 🐥 📰

Text Captions

The program prints the game's initial state: the two smaller glasses are empty and the largest, 8-ounc glass, is full.

I choose glass 2 to receive water from glass 3. Glass 2, which has a volume of 5 ounces, is filled with water, leaving 3 ounces of water in glass 3. Glass 1 wasn't used and remains empty.

Slide 51

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) P chap.09 Image:	− □ ×
In the Lak New NeyLet Your Decking Lak Angle Your Octavity Barriello Image: Control of the Lak NeyLet Your Decking Class 1: 0 / 3 Image: Control of the Lak NeyLet Your Decking Class 1: 0 / 3 Image: Control of the Lak NeyLet Your Decking Class 1: 0 / 3 Image: Control of the Lak NeyLet Your Decking Clas	Plorer Vition Explorer Plose Plorer Vition Explorer Plose Plorer Plorer Plorer Plore Plor
☐ Ready Add to Sour	rce Control 🔺 🐥 📰

Text Captions

My next move pours water from glass 2 to glass 1, which fills glass 1 with 3 ounces and leaves 2 ounces in glass 2. Glass 3 wasn't used this time, so it still holds 3 ounces.

Slide 52

File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	0 ₿ − □ ×
🖉 ♥ ♥ 📅 ♥ 🏝 🔛 🖓 ♥ ♥ ♥ Debug 🔹 x86 🔹 🕨 Local Windows Debugger ♥ 🏓 🞯 🛫 🔚 👘 🖆 🦉 📕 🐄 🦄 🦄 🤤	🛱 Live Share 🎗
game.op • X Glass.op Glass. 22 Glass 1: 0 / 3 Glass 2: 0 / 5 Glass 3: 8 / 8 23 Glass 2: 0 / 5 Glass 3: 8 / 8 Our TO glass: <1, 2, or 3; or enter 4 to quity: 2 24 Glass 3: 3 / 8 Our TO glass: <1, 2, or 3; or enter 4 to quity: 3 Glass 3: 3 / 8 26 Glass 3: 3 / 8 Our TO glass: <1, 2, or 3; or enter 4 to quity: 1 Our TO glass: <1, 2, or 3; or enter 4 to quity: 2 29 Glass 1: 3 / 3 Glass 3: 3 / 8 Our TO glass: <1, 2, or 3; or enter 4 to quity: 1 30 Glass 3: 3 / 8 Our TO glass: <1, 2, or 3; or enter 4 to quity: 1 31 Pour FMW glass: <1, 2, or 3; or enter 4 to quity: 1 32 Glass 1: 0 / 3 Glass 1: 0 / 3 33 Glass 1: 0 / 3 Glass 1: 0 / 3 33 Glass 1: 0 / 3 Glass 1: 0 / 3 33 Glass 2: 2 / 5 Glass 1: 0 / 3 33 Glass 2: 2 / 5 Glass 2: 0 / 5 34 Pour TO glass: <1, 2, or 3; or enter 4 to quity: 1 34 Pour TO glass: <1, 2, or 3; or enter 4 to quity: 3 34 Pour TO glass: <1, 2, or 3; or enter 4 to quity: 3 35 Io tou 36	 Solution Explorer A × Solution Explorer A × A ×
🗇 Ready	🛧 Add to Source Control 🔺 🐥 👘

Text Captions

Next, pour water from glass 1 to glass 3.

Slide 53

K File Edit Vie Image: Image of the state of the s	ew Project Build Debug B 🖬 📽 🛛 🕈 - 🤇 - 🗍 Debug	Test Analyze Tools Exten g • x86 • ▶ Loc	sions Window Help al Windows Debugger + }	Search (Ctri+Q)	우 chap.09	o - 🗆 🗙 🖻 📴 🕑 🕑
game.cpp → X game.cpp → X Sel Glass1 22 23 24 25 26 27 28 29 30 31 32 33 34 35 150 % • • • • • • • Output Show output from: 4	Glass.cpp Glass.h Glass.tp Glass.h Glass 1: 0 / 3 Glass 2: 0 / 5 Glass 3: 8 / 8 Pour TO glass: <1, Glass 1: 0 / 3 Glass 2: 5 / 5 Glass 3: 3 / 8 Pour TO glass: <1, Glass 2: 2 / 5 Glass 1: 3 / 3 Glass 2: 2 / 5 Glass 3: 3 / 8 Pour TO glass: <1, Glass 2: 2 / 5 Glass 3: 6 / 8 Pour TO glass: <1, Pour FROM glass: <1, Glass 1: 0 / 3 Glass 2: 2 / 5 Glass 3: 6 / 8 Pour TO glass: <1, Pour FROM glass: <1, Mor FROM glass: <1, Pour TO glass: <1, Pour TO glass: <1, Mor TO glass 1: 2 / 3 Glass 2: 0 / 5 Glass 3: 6 / 8 Pour TO glass: <1, Mor TO	Debug\Glass1.exe	<pre>quit>: 2 quit>: 3 quit>: 1 quit>: 1 quit>: 2 quit>: 1 quit>: 1 quit>: 1 quit>: 1 quit>: 1 quit>: 2 quit>: 2</pre>			Solution Explorer Solution Expl Team Explorer Add to Source Control

Text Captions

The next move repeats the second move: pour water from glass 2 to glass 1.

Slide 54

🕅 File Edit View Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) 🔎 chap.09 👰 — 🗆 🗙
🕴 😋 + 💿 指 + 🏠 🔐 🔐 🎐 + 代 + Debug - x86 > Local Windows Debugger + 第 函 🛒 旨 領 注 注 異 🧐 潮 🧤 🚎 🖓 化 Live Share 🔗
Image: Solution of the solution
🗇 Ready 🕎 Add to Source Control 🔺 🐥

Text Captions

Pouring water from glass 3 to glass 2 also repeats an earlier move – the first one – but this time it sets the game up for the final, winning move.

Slide 55

🕅 File Edit View	w Project Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q)	● – – ×
G • 🖸 🏠 • 🏠	💾 🔐 💙 • 🔇 • 🛛 Debug • x86 • • 🕨 Local Windows Debugger • 🎜 🙆 🛫 🔚 🌆 📜 📜 🐧 🦄 🦄 🛫	🖻 Live Share 🖗
game.cpp + × 0 22 23 23 23 24 25 26 27 28 29 30 31 32 33 34 35 150 % • • • Vutput Show output from: • • 4 7 Ready •	Gass, p Gass 1 Gass 1: 3 / 3 Glass 1: 3 / 3 Glass 2: 2 / 5 Glass 3: 3 / 8 Pour TO glass: (1, 2, or 3; or enter 4 to quit): 3 Pour FROM glass: (1, 2, or 3; or enter 4 to quit): 1 Glass 1: 0 / 3 Glass 1: 0 / 3 Glass 1: 0 / 3 Glass 1: 0 / 3 Glass 1: 0 / 3 Glass 1: 0 / 3 Glass 1: 0 / 3 Glass 1: 0 / 3 Glass 1: 2 / 3 Glass 1: 0 / 3 Glass 1: 2 / 3 Glass 1: 2 / 3 Glass 1: 2 / 3 Glass: (1, 2, or 3; or enter 4 to quit): 2 Pour Tog Glass: (1, 2, or 3; or enter 4 to quit): 2 Glass 1: 2 / 3 Glass: (1, 2, or 3; or enter 4 to quit): 2 Pour FROM glass: (1, 2, or 3; or enter 4 to quit): 2 Pour FROM glass: (1, 2, or 3; or enter 4 to quit): 1 Pour FROM glass: (1, 2, or 3; or enter 4 to quit): 2 Pour FROM glass: (1, 2, or 3; or enter 4 to quit): 2 Our FROM glass: (1, 2, or 3; or enter 4 to quit): 2 Our FROM glass: (1, 2, or 3; or enter 4 to quit): 2 Our FROM glass: (1, 2, or 3; or enter 4 to quit): 2 Our FROM glass: (1, 2, or 3; or enter 4 to quit): 2	Solution Explorer A Solution Explorer A Ution Chap.09 (2 of 2 p) iass1 References External Dependence Header Files Glass.h Resource Files Source Files ** game.cpp ** Glass.cpp tudent References External Dependence Header Files Source Files ** subscript tudent References External Dependence Header Files Source Files ** student.cpp V Solution Expl Feam Explorer Add to Source Control •

Text Captions

Pouring from glass 2 to glass 1 leaves 4 ounces of water in glass 2, which solves the problem and ends the while loop. Below and outside the loop, the program prints the total number of pour operations taken to solve the puzzle and the final state of the game. I've not been able to solve the puzzle with fewer than 6 pouring operations.

Slide 56

▷ () File Edit View Pr ⊙ - ○ 裕 - 쓸 💾 🚽	oject Build Debug Test Analyze Tools Extensions Window Help Search (Ctrl+Q) $ ho$ chap.09	🕞 — 🗆 🗙 🕼 Live Share 🕅
game.cpp a Glass.cp BGlass1 22 1 23 23 1 24 25 2 25 26 2 26 27 2 28 29 30 31 32 33 34 35 35	<pre>p Glass.h</pre>	 Solution Explorer Solution Explorer Search Solution Explorer () Search Solution Explorer () Solution 'chap.09' (2 of 2 p) Source Files
150% No issue	stound	CRLF

Text Captions