Week #2 Self-assessments: C++ review

Due Sep 3 at 11:59pm **Points** 15 **Questions** 15

Available until Sep 4 at 11:59pm **Time Limit** None **Allowed Attempts** 3

Instructions

This quiz is to help you review and self-assess your learning in this course. In addition, the objective is to motivate you to review concepts in a timely and diligent manner. These quizzes do count towards your final grade (see Syllabus). **Consequently, please do take these quizzes seriously** .

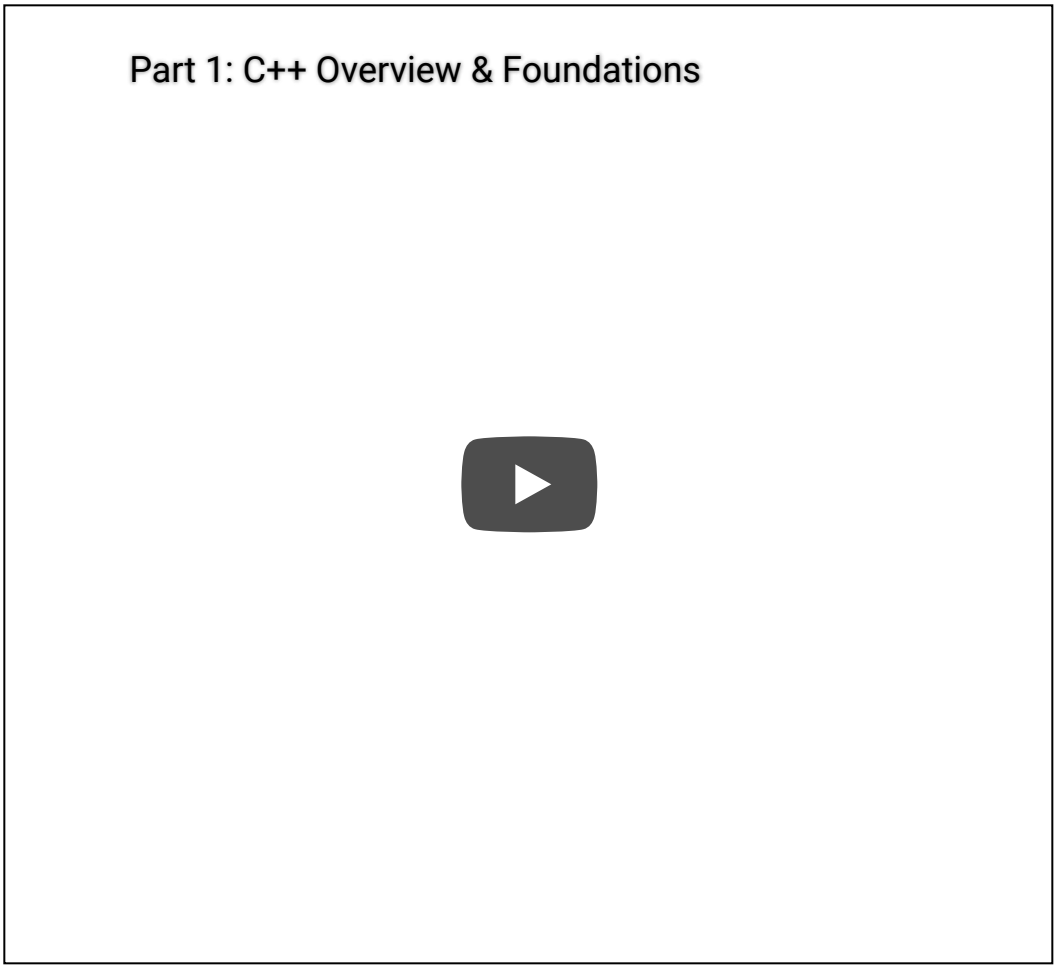
This quiz was locked Sep 4 at 11:59pm.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	20 minutes	15 out of 15

Score for this attempt: **15** out of 15
Submitted Sep 2 at 6:18pm
This attempt took 20 minutes.

The next set of questions are based on the content covered in the following video:



Question 1

1 / 1 pts

The value stored in variable s shown below is (Note: sub-string in C++ has different parameters than Java)

```
std::string str = "123456789";
std::string s = str.substr(100);
```

Correct!

- (●) This is an invalid method call
- ( ) 123456789
- ( ) 89
- ( ) An empty string

Question 2

1 / 1 pts

Assume you have to iterate over each character in the string `str`. The preferred loop to be used would be:

- foreach (char c : str) {}
- for (char c: str) { } — Correct!
- for (int i = 0; (i < str.size()); i++) {}
- for (size_t i = 0; (i < str.size()); i++) {}

Question 3

1 / 1 pts

The value stored in variable s shown below is (Note: sub-string in C++ has different parameters than Java)

```
std::string str = "123456789";
std::string s = str.substr(2, 3);
```

- This is an invalid method call
- 123
- 3
- 345 — Correct!

Question 4

1 / 1 pts

Assume the following code is used to read user inputs.

```
int i;
std::string str;
std::cin >> i >> i >> str;
```

Assume the user enters the following input. The data that will be stored in the variable `str` will be:

```
10 20 testing
```

- 10
- testing

Correct!

- This is incorrect C++
- 20

Question 5

1 / 1 pts

The following for-loop that iterates over a string works:

```
std::string str = "some data";
for (int i = 0; (i < str.size()); i++) {
    str[i] = '*';
}
```

However, the compiler generates a warning:

```
warning: comparison between signed and unsigned integer expressions
```

The correct solution to fix this issue would be:

- Change int to size_t

Correct!

- It is incorrect to modify the string

- Remove the for-loop
- Change the string to const

Question 6

1 / 1 pts

The value stored in variable s shown below is (Note: sub-string in C++ has different parameters than Java)

```
std::string str = "123456789";
std::string s = str.substr(7);
```

- This is an invalid method call
- 7
- 8
- 89

Correct!

The next set of questions are based on the content covered in the following video:

Part 2: Functions/Methods and parameter-passing

Question 7

1 / 1 pts

The correct/preferred signature for a function named `doIt`, that accepts a string that the function is not going to modify, would be:

Correct!

- (●) void doIt(const std::string& str)
- ( ) void doIt(std::string str)
- ( ) void doIt(std::string& str)
- ( ) void doIt(const std::string str)

Question 8

1 / 1 pts

Assume the following code

```
void doIt(std::string s);
```

The method is using:

orrect Answer

- Pass by pointer
- Pass by reference
- Pass by name
- Pass by value

Correct!

Question 9

1 / 1 pts

Assume the following code

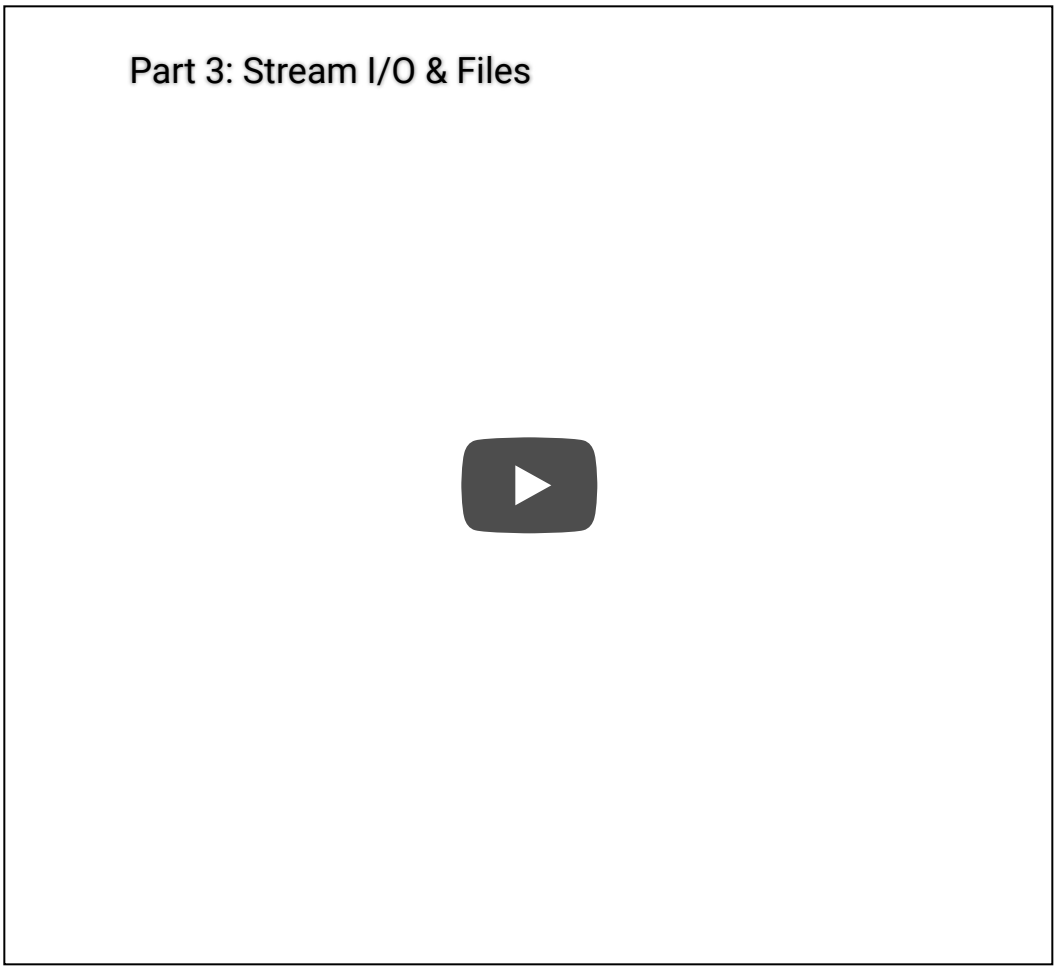
```
void doIt(int& i);
```

The method is using:

Correct!

- Pass by reference
- Pass by name
- Pass by pointer
- Pass by value

The next set of questions are based on the content covered in the following video:



Question 10

1 / 1 pts

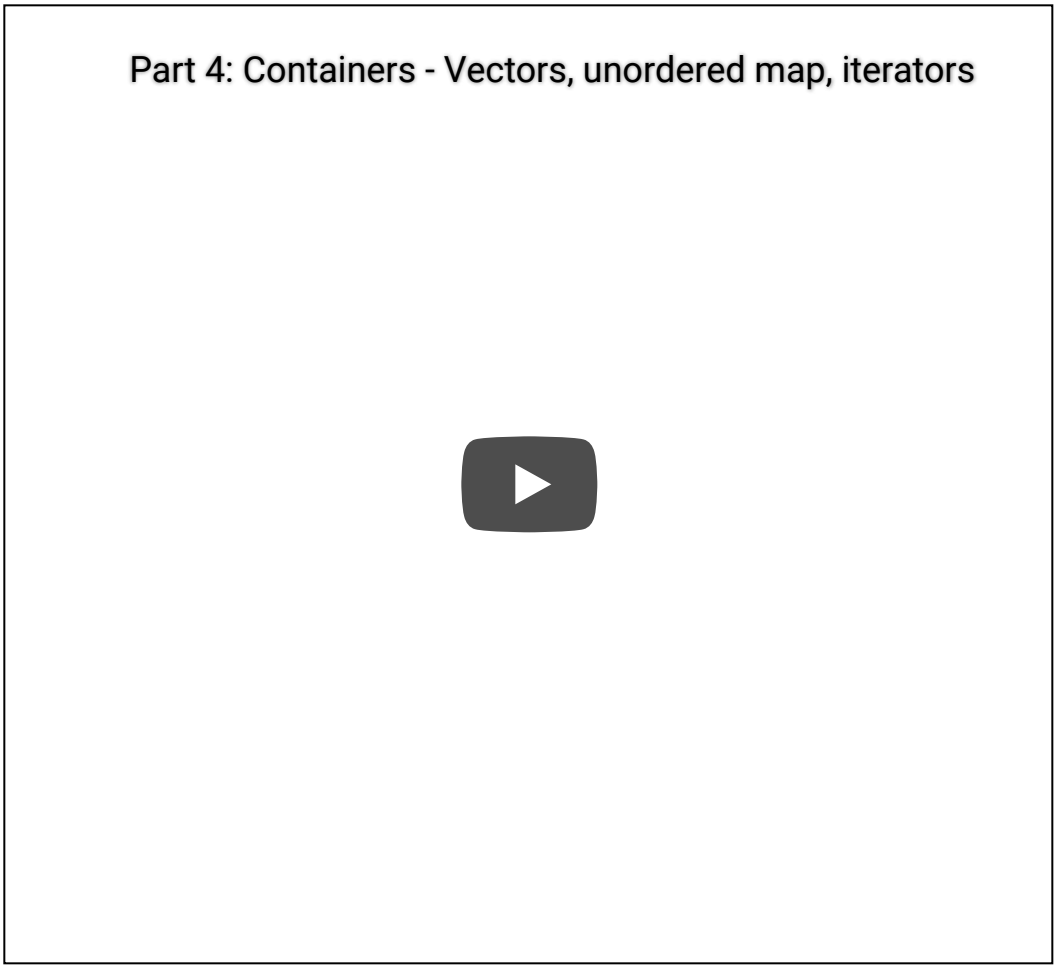
In order to write/overwrite data to a file, the correct stream to use is

- std::ostringstream
- std::cout
- std::ifstream

Correct!

- std::ofstream

The next set of questions are based on the content covered in the following video:



Question 11

1 / 1 pts

The preferred method to add an element to the end of a vector is:

Correct!

- push_back

- append
- insert
- add

Question 12

1 / 1 pts

The working with constant unordered_maps, we cannot use [] (array syntax) to access entries in it, because the property of [] is to add an entry if the value does not exist. However, for constants maps, we cannot add entries and hence we cannot use the array syntax. Instead, the correct method to use is:

Correct!

- (•) at()
- get()
- find()
- show()

Question 13

1 / 1 pts

Given the following definition for an `unordered_map`, what is the preferred approach for inserting/adding a value into the `unordered_map`?

```
using StrIntMap = std::unordered_map<std::string, int>;
StrIntMap wordNum;
```

- wordNum.put(10, "three");

- wordNum.put("three", 10);
- wordNum[10] = "three";

Correct!

- (selected) wordNum["three"] = 10;

Question 14

1 / 1 pts

Given the following code fragment, the 2nd line of code that access an entry:

```
std::unordered_map<std::string, int> check;
check["test"] = 22;
```

- Generates a compiler error

Correct!

- (selected) Adds an entry with key/index "test" with value 22
- Will generate an runtime error because "test" does not exist
- Adds an entry with key/index 22 with value "test"

Question 15

1 / 1 pts

The correct approach/method to check if a given value exists in an unordered_map is to use

- Use the insert method

Correct!

- (selected) Use find() method
- Use the has() method

- Use [] and check for zero/null

Quiz Score: **15** out of 15