C++ INSTITUTE CPP-22-02

C++ Institute CPP Certified Professional Programmer Certification Questions & Answers

Exam Summary – Syllabus –Questions

CPP-22-02

CPP - C++ Certified Professional Programmer

40 Questions Exam – 70 % Cut Score – Duration of 65 minutes

Table of Contents:

Know Your CPP-22-02 Certification Well:

The CPP-22-02 is best suitable for candidates who want to gain knowledge in the C++ Institute C++ Programming. Before you start your CPP-22-02 preparation you may struggle to get all the crucial CPP Certified Professional Programmer materials like CPP-22-02 syllabus, sample questions, study guide.

But don't worry the CPP-22-02 PDF is here to help you prepare in a stress-free manner.
The PDF is a combination of all your queries like-

- What is in the CPP-22-02 syllabus?
- How many questions are there in the CPP-22-02 exam?
- Which Practice test would help me to pass the CPP-22-02 exam at the first attempt?

Passing the CPP-22-02 exam makes you CPP - C++ Certified Professional Programmer. Having the CPP Certified Professional Programmer certification opens multiple opportunities for you. You can grab a new job, get a higher salary or simply get recognition within your current organization.

C++ Institute CPP-22-02 CPP Certified Professional Programmer Certification Details:

Exam Name	CPP - C++ Certified Professional Programmer
Exam Code	CPP-22-02
Exam Price	$295 (USD)
Duration	65 mins
Number of Questions	40
Passing Score	70%
Books / Training	**C++ Advanced (Advanced) (Edube, self-enroll/self-study)**
Schedule Exam	**Pearson VUE**
Sample Questions	**C++ Institute CPP Certified Professional Programmer Sample Questions**
Practice Exam	**C++ Institute CPP-22-02 Certification Practice Exam**

CPP-22-02 Syllabus:

Topic	Details
Sequence Containers and Container Adapters	- basic concepts of using: • std::vector • std::deque • std::list • std::queue • std::priority_queue • std::stack - methods of vector, deque, list, queue, priority_queue, and stack; - using vector, deque, list, queue, priority_queue, and stack with simple and complex (i.e., their own classes) types; - iterators for vector, deque, list, queue, priority_queue, and stack; - accessing data stored in vector, deque, list, queue, priority_queue, and stack.
Associative Containers	- basic concepts of using associative containers, like std::set, std::multiset, std::map, and std::multimap; - methods of set, multiset, map, and multimap; - using set, multiset, map and multimap with simple, and complex (i.e., their own classes) types; - iterators for set, multiset, map, and multimap; - accessing data stored in set, multiset, map, and multimap.
Algorithms: Non-Modifying Sequence Operations	- concept of non-modifying iterating through containers, using std::for_each function; - using std::find, std::find_if, std::find_end, std::find_first_of, std::adjacent_find, std::search, and std::search_n operations for finding elements in a wide spectrum of containers; - using std::count and std::count_if operations for counting elements in containers; - using std::mismatch and std::equal to compare two ranges of containers.
Algorithms: Modifying Sequence Operations	- using std::copy, std::copy_backward, std::fill, std::fill_n, std::generate, and std::generate_n for creating data; - using std::swap_ranges, std::swap, std::iter_swap, std::transform, std::replace, std::remove, std::remove_if, std::unique, std::unique_copy, std::reverse, std::reverse_copy, std::rotate, std::partition, and std::stable_partition for creating and modifying data.

Topic	Details
Algorithms: Sorting and Binary Search	- using std::sort and std::stable_sort for sorting containers; - using std::lower_bound, std::upper_bound, and std::binary_search for searching in ordered containers.
Algorithms: Merge, Heap, Min, Max	- using std::merge and std::inplace_merge for merging data; - using std::includes, std::set_union, std::set_intersection, std::set_difference, and std::set_symmetric for manipulating sets of simple and complex types; - using std::min_element and std::max_element for finding extreme values in containers of simple and complex types.
STL Functional Objects and Utilities	- using different utilities to transform data with std::transform (plus(), minus(), and ptr_fun bounded functions) for simple and complex types.
Advanced I/O	- advanced use cases of cout/cin and other stream objects; - using setf and unsetf flag methods to manipulate I/O stream format; - using operators like boolalpha, noshowpoint, setprecision, fixed, and setw to manipulate I/O stream format.
Templates	- defining template functions and specialized functions; - defining template classes and instantiating them; - using functions and operator functions from other classes with template classes; - using template classes inside other template classes.
Smart Pointers	- basic syntax and semantics of std::unique_ptr and std::shared pointers; - typical use cases; - possibilities of conversion between shared and unique pointers.
Selected Important Language Features	- using auto specifier for declaring variables with automatically deducted types; - range-based for loops; - using lambdas for creating shorter and more readable code; - using constexpr specifier for declaring possibility to evaluate value of the function or variable at compile time; - using tuples for grouping data together; - using strongly-typed enums for better enumerating of elements.

C++ Institute CPP-22-02 Sample Questions:

Question: 1

Which statements are true about std::binary_search()?

a) It works on sorted containers
b) It requires a comparator function
c) It returns a boolean indicating if the element is found
d) It modifies the container
e) It operates in O(log n) time complexity

Answer: a, c, e

Question: 2

What is the primary purpose of using templates in C++?

a) To provide polymorphism at runtime
b) To reduce code duplication by supporting generic programming
c) To enable inheritance in classes
d) To optimize memory allocation

Answer: b

Question: 3

You need to find the range of positions where a specific value occurs in a sorted vector. Which algorithm should you use?

a) std::lower_bound()
b) std::upper_bound()
c) std::equal_range()
d) std::binary_search()

Answer: c

Question: 4

How do you define a class template in C++?

a) template<class T> class MyClass {}
b) template<typename T> class MyClass {}
c) template<class T, typename U> class MyClass {}
d) All of this

Answer: d

Question: 5

Which operations allow counting elements in a container?

a) std::count()
b) std::count_if()
c) std::find()
d) std::find_if()
e) std::search()

Answer: a, b

Question: 6

How do you reset the formatting flags for an std::ostream object?

a) unsetf()
b) setf()
c) clear()
d) reset()

Answer: a

Question: 7

Which are valid use cases for lambda functions?

a) Passing as arguments to algorithms
b) Capturing local variables by value or reference
c) Declaring global functions
d) Defining inline, anonymous functions
e) Overloading operators in classes

Answer: a, b, d

Question: 8

You are tasked with finding the maximum element in a container and its position. Which algorithm should you use?

a) std::max()
b) std::max_element()
c) std::find()
d) std::count()

Answer: b

Question: 9

What are valid uses of std::setw()?

a) To specify the minimum width for output
b) To control alignment in an output field
c) To pad output with default spaces or specified characters
d) To truncate output exceeding the width
e) To center-align output

Answer: a, b, c

Question: 10

How do you insert a key-value pair into an std::map?

a) insert()
b) emplace()
c) operator[]
d) All of the above

Answer: d

Study Guide to Crack C++ Institute CPP Certified Professional Programmer CPP-22-02 Exam:

- Getting details of the CPP-22-02 syllabus, is the first step of a study plan. This pdf is going to be of ultimate help. Completion of the syllabus is must to pass the CPP-22-02 exam.
- Making a schedule is vital. A structured method of preparation leads to success. A candidate must plan his schedule and follow it rigorously to attain success.
- Joining the C++ Institute provided training for CPP-22-02 exam could be of much help. If there is specific training for the exam, you can discover it from the link above.
- Read from the CPP-22-02 sample questions to gain your idea about the actual exam questions. In this PDF useful sample questions are provided to make your exam preparation easy.
- Practicing on CPP-22-02 practice tests is must. Continuous practice will make you an expert in all syllabus areas.

Reliable Online Practice Test for CPP-22-02 Certification

Make EduSum.com your best friend during your CPP - C++ Certified Professional Programmer exam preparation. We provide authentic practice tests for the CPP-22-02 exam. Experts design these online practice tests, so we can offer you an exclusive experience of taking the actual CPP-22-02 exam. We guarantee you 100% success in your first exam attempt if you continue practicing regularly. Don't bother if you don't get 100% marks in initial practice exam attempts. Just utilize the result section to know your strengths and weaknesses and prepare according to that until you get 100% with our practice tests. Our evaluation makes you confident, and you can score high in the CPP-22-02 exam.

Start Online practice of CPP-22-02 Exam by visiting URL
https://www.edusum.com/c-institute/cpp-22-02-cpp-c-certified-professional-programmer