Capitulo 23

Section 23.2.3 deque Sequence Container 23.2.3 Q1: Class deque provides:

This activity contains 39 questions.

1. *Section 23.1 Introduction to the Standard Template Library (STL)*

23.1 Q1: Which of the following is not a key component of the STL?

- *Pointers.*
- *Containers.*
- *Algorithms.*
- *Iterators.*

2. *Section 23.1.1 Introduction to Containers*

23.1.1 Q1: Which of the following is not an STL container type?

- *Container adapters.*
- *Sequence containers.*
- *Second-class containers.*
- *Associative containers.*

3. *Section 23.1.2 Introduction to Iterators*

23.1.2 Q1: Iterators are similar to pointers because of the:

- *-> operator.*
- *& operator.*
- ** and ++ operators.*
- *begin and end functions.*

4. *Section 23.1.3 Introduction to Algorithms*

23.1.3 Q1: An STL algorithm cannot:

- *Take two iterators as arguments to specify a range.*
- *Return an iterator.*
- *Be used with containers that support more powerful iterators than the minimum requirements for the algorithm.*
- *Access STL members directly.*

5. *Section 23.2 Sequence Containers*

23.2 Q1: Which of the following is not a sequence container provided by the STL?

- *deque.*
- *array.*
- *list.*
- *vector.*

6. *23.2 Q2: Which of the following applications would a deque not be well suited for?*

- *Applications that require frequent insertions and deletions at the front of a container.*
- *Applications that require frequent insertions and deletions at the back of a container.*
- *Applications that require frequent insertions and deletions at the front and at the back of a container.*
- *Applications that require frequent insertions and deletions in the middle of a container.*

7. *Section 23.2.1 vector Sequence Container*

23.2.1 Q1: Which of the following is a difference between vectors and arrays?

- *Access to any element using the [] operator.*
- *The ability to change size dynamically.*
- *Efficient direct access to any element.*
- *Stored in contiguous blocks of memory.*

8. *23.2.1 Q2: The erase member function of class vector cannot:*

- *Specify a range of elements to be removed from the vector.*
- *Specify a value to be removed from the vector.*
- *Specify an element to be removed from the vector.*
- *Be called by member function clear.*

9. *Section 23.2.2 list Sequence Container*

23.2.2 Q1: The list sequence container does not:

- *Use a doubly linked list.*
- *Automatically sort inserted items.*

- *Efficiently implement insert and delete operations anywhere in the list.*
- *Support bidirectional iterators.*

10. *23.2.2 Q2: Which of the following is not a member function of all sequence containers?*

- *back.*
- *front.*
- *push_back.*
- *push_front.*

11. *Section 23.2.3 deque Sequence Container*

23.2.3 Q1: Class deque provides:

- *The ability to add storage at either end of the deque.*
- *Efficient indexed access to its elements.*
- *Efficient insertion and deletion operations at the front and back of a deque.*
- *All of the above.*

12. *Section 23.3 Associative Containers*

23.3 Q1: The main difference between set and multiset is:

- *That one deals with keys only, and the other deals with key/value pairs.*
- *How they handle duplicate keys.*
- *Their interface.*
- *Their efficiency.*

13. *23.3 Q2: Data loss could occur if the contents of a _____ were placed into any of the other three associative container types.*

- *set.*
- *multimap.*
- *multiset.*
- *map.*

14. *Section 23.3.1 multiset Associative Container*

23.3.1 Q1: The multiset associative container does not:

- *Permit random access to its keys.*
- *Need a header file to be used.*

- *Use its comparator function object to determine the order of its data.*
- *Arrange its data in ascending order by default.*

15. *Section 23.3.2 set Associative Container*

23.3.2 Q1: If a program attempts to insert a duplicate key into a set:

- *The set will contain multiple copies of that key.*
- *A compile error will occur.*
- *The duplicate key will be ignored.*
- *An exception is thrown*

16. *Section 23.3.3 multimap Associative Container*

23.3.3 Q1: The expression std::multimap< int, double, std::less< int > >::value_type(15, 2.7):

- *Creates a multimap object containing one key/value pair.*
- *Creates a pair object in which first is 15 (type int) and second is 2.7 (type double).*
- *Returns the number of times the key/value pair (15, 2.7) appears in the multimap.*
- *Creates an empty multimap object.*

17. *Section 23.3.4 map Associative Container*

23.3.4 Q1: If pairs is a map containing int keys and double associated values, the expression pairs[5] = 10:

- *Associates the value associated with key 10 to key 5 in pairs.*
- *Associates the value 5.0 to the key 10 in pairs.*
- *Associates the value associated with key 5 to key 10 in pairs.*
- *Associates the value 10.0 to the key 5 in pairs.*

18. *Section 23.4 Container Adapters*

23.4 Q1: Select the false statement. Container adapters:

- *Do not provide the actual data structure implementation for elements to be stored.*
- *Have limited iterator support.*
- *Have their data stored by underlying data structures.*
- *Can push and pop elements.*

19. *Section 23.4.1 stack Adapters*

23.4.1 Q1: To pop an element off the top of a stack for processing:

- *Use member function pop.*
- *Use member function top.*
- *Use member function top and then member function pop.*
- *Use member function pop and then member function top.*

20. *Section 23.4.2 queue Adapters*

23.4.2 Q1: Which of the following is a not a member function of queue?

- *enqueue.*
- *pop.*
- *empty.*
- *size.*

21. *Section 23.4.3 priority_queue Adapters*

23.4.3 Q1: Which of the following statements is true of a priority_queue?

- *Each of its common operations is implemented as an inline function.*
- *It must be implemented as a deque.*
- *A bucket sort is usually associated with it.*
- *It does not allow insertions in sorted order.*

22. *Section 23.5 Algorithms*

23.5 Q1: The algorithms in the STL:

- *Use virtual function calls.*
- *Are not as efficient as the algorithms presented in most textbooks.*
- *Are implemented as member functions of the container classes.*
- *Are implemented as member functions of the container classes.*

23. *Section 23.5.1 fill, fill_n, generate and generate_n*

23.5.1 Q1: The easiest way to set all the values of a vector to zero is to use function:

- *generate_n.*
- *generate.*
- *fill_n.*

- *fill.*

24. *23.5.1 Q2: Which of the following function calls is a valid way to place elements into vector< char > chars?*

- *std::fill_n(chars.begin(), chars.end(), '5');.*
- *std::generate_n(10, chars.end(), '5');.*
- *std::fill(chars.begin(), chars.end(), '5');.*
- *std::generate(chars.begin(), 10, '5');.*

25. *Section 23.5.2 equal, mismatch and lexicographical_compare*

23.5.2 Q1: Given that v1 and v2 are vectors, the function call std::equal(v1.begin(), v1.end(), v2.begin()) returns:

- *An iterator pointing to the first location where v1 and v2 are not equal.*
- *An iterator pointing to the first location where v1 and v2 are equal.*
- *A bool indicating whether the first element of v1, the last element of v1 and the first element of v2 are all equal.*
- *A bool indicating whether v1 and v2 are equal.*

26. *Section 23.5.3 remove, remove_if, remove_copy and remove_copy_if*

23.5.3 Q1: Mr. Smith has a shopping list stored in a vector. Today, Mrs. Smith decides that she will go get the items that cost less than 10 dollars. If Mr. Smith wants to give his wife a list of her own, he should use the function:

- *remove_if.*
- *remove_copy.*
- *remove.*
- *remove_copy_if.*

27. *Section 23.5.4 replace, replace_if, replace_copy and replace_copy_if*

23.5.4 Q1: The order of the arguments passed to function replace_copy_if must be:

- *OutputIterator, InputIterator, ReplacementValue, PredicateFunction.*
- *OutputIterator, InputIterator, InputIterator, ReplacementValue, PredicateFunction*
- *InputIterator, InputIterator, OutputIterator, PredicateFunction, ReplacementValue*
- *InputIterator, OutputIterator, PredicateFunction, ReplacementValue*

28. *Section 23.5.5 Mathematical Algorithms*

23.5.5 Q1: Which of the following is not a mathematical algorithm included in the STL?

- *copy.*
- *accumulate.*
- *transform.*
- *min_element.*

29. *Section 23.5.6 Basic Searching and Sorting Algorithms*

23.5.6 Q1: The easiest way to search through a list of names and output the first one that begins with a vowel would be to use function:

- *binary_search.*
- *sort.*
- *find*
- *find_if.*

30. *Section 23.5.7 swap, iter_swap and swap_ranges*

23.5.7 Q1: Functions iter_swap and swap_ranges are similar in that both:

- *Can only swap elements within the same array or container.*
- *Take two arguments.*
- *Take forward iterators as arguments.*
- *Swap a range of elements*

31. *Section 23.5.8 copy_backward, merge, unique and reverse*

23.5.8 Q1: Which of the following statements produces identical results as the statement:

```
std::copy( v1.begin(), v1.end(), v2.begin() );
if v1 and v2 are both 10-element vectors?
```

- *std::copy_backward(v1.begin(), v1.end(), v2.end());.*
- *std::copy_backward(v2.begin(), v2.end(), v1.end());.*
- *std::copy_backward(v1.begin(), v1.end(), v2.begin());.*
- *std::copy_backward(v2.begin(), v2.end(), v1.begin());.*

23.5.9 Q1: If v1 is a vector< int > containing some number of int elements sorted in

32. *ascending order, after these statements execute:*

```
std::vector< int > results1;
std::vector< int > results2;
std::unique_copy( v1.begin(), v1.end(), std::back_inserter( results1 ) );
std::reverse_copy( v1.begin(), v1.end(), std::back_inserter( results2 ) );
```

which of the following could be true?

- *results1 contains more elements than results2.*
- *The first element in results1 matches the last element in results2.*
- *results1 is empty but results2 is not.*
- *None of the above.*

33. *Section 23.5.10 Set Operations*

23.5.10 Q1: The _____ function would produce the sequence 1, 5, 6 when passed the sequences 1, 2, 3, 4, 5, 6 and 2, 3, 4, 7 as first/second and third/fourth arguments, respectively.

- *set_difference.*
- *set_union.*
- *set_symmetric_difference.*
- *set_intersection.*

34. *Section 23.5.11 lower_bound, upper_bound and equal_range*

23.5.11 Q1: Functions lower_bound, upper_bound and equal_range are different in their:

- *Return types.*
- *Second argument types.*
- *First argument types.*
- *Third argument types.*

35. *Section 23.5.12 Heapsort*

23.5.12 Q1: Attributes of a heap do not include:

- *The children of a given node are less than or equal to the parent node's value.*
- *Having a binary-tree structure.*
- *Having the largest element at the top of the heap.*
- *A preference to pop, rather than push, elements in the heap.*

36. *Section 23.5.13 min and max*

23.5.13 Q1: Which of the following function calls would not return the value that is its first argument?

- *std::min('N', 'P').*
- *std::max(17, 16).*
- *std::max('d', 'k').*
- *std::min(3, 23).*

37. *Section 23.5.14 STL Algorithms Not Covered in This Chapter*

23.5.14 Q1: The difference between functions partition and stable_partition is that:

- *partition may throw an exception while stable_partition will not.*
- *stable_partition allows an element to be duplicated and placed into both partitions.*
- *partition can only be called on a sequence that is already sorted.*
- *stable_partition maintains the original order for the elements in each of the two resulting partitions with respect to the other elements in that same partition.*

38. *Section 23.6 Class bitset*

23.6 Q1: Which of the following bitset member functions cannot be called with an empty argument list?

- *reset.*
- *test.*
- *none.*
- *size.*

39. *Section 23.7 Function Objects*

23.7 Q1: Function objects have their functions called by using:

- *operator().*
- *The arrow operator (->).*
- *The dot operator (.).*
- *The binary scope resolution operator (::).*

Clear Answers / Start Over Submit Answers for Grading

Answer choices in this exercise appear in a different order each time the page is loaded.