1. The term STL stands for?
   a) Simple Template Library
   b) Static Template Library
   c) Single Type Based Library
   d) Standard Template Library

Answer: d
2. Which of the following statements regarding the design of the Standard Template Library (STL) in C++ is (are) true?
Each STL algorithm is usable with one specific container.
The STL does not use templates and instead relies on polymorphism.

(a) None  
(b) II only  
(c) I and II  
(d) I only

Answer : a
3. For an STL iterator it, execution of the statement `it--;` does which of the following?
(a) Decreases by 1 the size of the container pointed to by it.
(b) Post-decrements the item to which the iterator points.
(c) Pre-decrements the item to which the iterator points.
(d) Steps the iterator backwards to the previous item.

Answer : d
4. Which of the following data structures is not a container implemented in the C++ Standard Template Library?

(a) List  
(b) Hash table  
(c) Vector  
(d) Stack

Answer : b
5. Consider the following C++ program fragment.

```cpp
vector<int> A(10);
A.push_back(5000);
```
At the end of an execution of this fragment, the size of vector A is

(a) 5000
(b) 10
(c) 1
(d) dependent on machine and compiler

Answer: c
In STL vectors, _____ refers to the maximum number of items that can be stored without resizing, and _____ refers to the number of items stored.

(a) range, domain
(b) capacity, size
(c) size, capacity
(d) domain, range

Answer : b
Consider the following program fragment that calls the method count_if in the C++ Standard Template Library.

deque<int> numbers;
...

count_if(numbers.begin(), numbers.end(), is_odd);

Which of the following declarations for the method is_odd is correct for the call to count_if?
(a) bool is_odd(int i);
(b) bool is_odd(int begin, int end);
(c) int is_odd(int begin, int end);
(d) int is_odd(bool i);

Answer: a
Consider the execution of the following.

`vector<int> A(10,20);`

Which of the following accurately describes what is created?

(a) An array of 20 arrays of ints, each of size 10  
(b) An array of ints, indexed from 10 to 20  
(c) An array of 10 arrays of ints, each of size 20  
(d) An array of 10 ints, each initialized to 20

Answer : d
Which of the following statements is (are) true regarding strings in C++?

I. Strings in C++ are supported by the standard class string.
II. A constructor for the class string can accept a C-style string as an argument.

(a) None
(b) II only
(c) I and II
(d) I only

Answer: c
Consider the following C++ program fragment, which is a partial declaration of the class `string`.

class string {
    public:
        string(const char* cstring = "");
        bool operator==(const string & rhs);
    ...
};

Given this class and two string objects called `s1` and `s2`, which of the following is not a legal statement?

(a) bool ans = (s2 == "hello");
(b) string s3("Hello, World");
(c) bool ans = ("hello" == s1);
(d) bool ans = (s1 == s2);

Answer: c
The main abstractions of the Standard Template Library include which of the following?

I. Iterators
II. Exception handlers
III. Algorithms

(a) III only
(b) I only
(c) I and II only
(d) I and III only

Answer: d
12 In the STL, common algorithms are instantiated for multiple types of container classes by using _____ to provide a uniform interface between the algorithms and containers.

(a) arrays  
(b) virtual functions  
(c) iterators  
(d) pointers

Answer: c
13 The class vector in the C++ STL contains which of the following methods?

I. push_back
II. push_front
III. pop_front

(a) I and II only
(b) I only
(c) I, II, and III
(d) II and III only

Answer : b
14 If A is an STL vector, then the effect of executing the statement
A.push_back( x ); is to

(a) append x to A if there is room, and otherwise overwrites the currently last element of A
(b) append x to A if and only if the size of A is less than capacity of A
(c) check whether the capacity of A is larger than the size of A, enlarges A if necessary, and append x to A
(d) append x to A, without checking the size and capacity of A

Answer : c
15 If A is an STL vector, then the effect of executing the statement

\[ A[i] = x; \]
is to
(a) check array bounds, enlarge the vector if necessary, and then write \( x \) to position \( i \)
(b) check array bounds, and write \( x \) to position \( i \) if and only if \( i \) is in the proper range
(c) write \( x \) to position \( i \) of the vector, without bounds checking
(d) create an iterator \( x \) pointing to position \( i \) in the array

Answer : c
16 For an STL iterator it, execution of the statement ++it; does which of the following?

(a) Post-increments the item to which the iterator points.
(b) Advances the iterator to the next item.
(c) Pre-increments the item to which the iterator points.
(d) Increase by 1 the size of the container pointed to by it.

Answer : b
17  Access to ranges of elements in an STL container is typically handled by

(a) references
(b) pointers
(c) iterators
(d) suitable access member functions

Answer : c
18 In the C++ Standard Template Library, vectors and deques differ in their interfaces for handling which of the following operations?
I. Insertion of an element at the front of the container
II. Insertion of an element at the end of the container
III. Removal of an element from the front of the container
(a) III only
(b) I and II only
(c) I and III only
(d) I only
Answer: c
19 The STL deque container contains which of the following methods?

I. push_back
II. push_front
III. pop_front
(a) II and III only
(b) III only
(c) I only
(d) I, II, and III

Answer: d
20 The size of an STL vector is defined to be the
(a) total of the sizes of the data members in the vector class
(b) number of bytes the vector occupies in memory
(c) maximum number of elements that can be stored in the vector without resizing
(d) number of elements currently stored in the vector

Answer: d
21 Consider the following declaration that makes use of a user-defined class Thing.

```cpp
vector<Thing> A(10);
```
In order that it compile, the class Thing must have which of the following?

(a) a destructor
(b) an assignment operator
(c) a copy constructor
(d) a default constructor

Answer: d
22 Execution of which of the following statements sets an STL iterator ‘it’ so that it points to the first element of a container A?

(a) A.reset( it );
(b) A.begin( it );
(c) begin( A, it );
(d) it = A.begin();

Answer : d
23 The STL is heavily based on

(a) polymorphism
(b) inheritance
(c) templates
(d) object oriented programming

Answer : c
24  The capacity of an STL vector is defined to be the
(a) difference between the current number of elements and the maximum number of elements
(b) maximum number of elements that can be stored in the vector without resizing
(c) number of elements currently stored in the vector
(d) maximum number of elements the vector could possibly have on the given machine

Answer : b
25 Name of the deque class comes from:
(a) the acronym "Double Ended QUEue"
(b) the acronym "Dynamically Extended QUEue"
(c) the acronym "Discrete Elements, Quantised Under Extension"
(d) the name of woman who invented it: Marie Alperìn de Qué

Answer: a
26 Iterators which can move freely any number of steps in one operation are called ---------?
   a) input iterators
   b) forward iterators
   c) random access iterators
   d) output iterators

Answer: c
27 There are --------types of sequence containers in the STL.?
   a) two
   b) four
   c) Eight
   d) three

Answer : d
28 An STL container can be used to
a. hold objects of class employee.
b. store elements in a way that makes them quickly accessible.
c. organize the way objects are stored in memory.
d. All of above

Answer : d
29 The STL sequence containers are v________, l________, and d________.

a. Vector, list, dequeu
b. Vector, List, dequeue
c. Vector List, deque

d. None

Answer : c
Two important STL associative containers are s_______ and ma_______.

a. Sort, Map
b. Set, Map
c. Standard Set, Map
d. None

Answer: b
31 An STL algorithm is
a. a standalone function that operates on containers.
b. a link between member functions and containers.
c. a friend function of appropriate container classes.
d. a member function of appropriate container classes.

Answer : a
34 One purpose of an iterator in the STL is to connect algorithms and containers.

a. True
b. False

Answer: a
35 The find() algorithm
a. finds matching sequences of elements in two containers.
b. finds a container that matches a specified container.
c. takes iterators as its first two arguments.
d. takes container elements as its first two arguments.

Answer : c
Algorithms can be used only on STL containers.

a. True
b. False

Answer : b
A vector is an appropriate container if you

a. want to insert lots of new elements at arbitrary locations in the vector.
b. want to insert new elements, but always at the front of the container.
c. are given an index number and you want to quickly access the corresponding element.
d. are given an element’s key value and you want to quickly access the corresponding element.

Answer: c
38 The back() member function removes the element at the back of the container.

a. True
b. False

Answer: b
If you define a vector v with the default constructor, and define another vector w with a one-argument constructor to a size of 11, and insert 3 elements into each of these vectors with push_back(), then the size() member function will return _______ for v and ______ for w.

a. 3, 11
b. 3,14
c. 3,3
d. None

Answer : b
40 The `unique()` algorithm removes all _________ element values from a container.

a. Present
b. Unique
c. Duplicate
d. a, b both

Answer: c
41 In a deque
a. data can be quickly inserted or deleted at any arbitrary location.
b. data can be inserted or deleted at any arbitrary location, but the process is relatively slow.
c. data can be quickly inserted or deleted at either end.
d. data can be inserted or deleted at either end, but the process is relatively slow.

Answer: c
An iterator can always move forward or backward through a container.

a. True
b. False

Answer: b
43 You must use at least a ______________ iterator for a list.
a. Forward
b. Backward
c. Bi-directional
d. Both a and b.

Answer : c
44 If iter is an iterator to a container, write an expression that will have the value of the object pointed to by iter, and will then cause iter to point to the next element.

a. *iter++  
b. (*iter).++  
c. Iter++  
d. All of above

Answer : a
45 The `copy()` algorithm returns an iterator to
a. the last element copied from.
b. the last element copied to.
c. the element one past the last element copied from.
d. the element one past the last element copied to.

Answer : d
To use a reverse_iterator, you should
a. begin by initializing it to end().
b. begin by initializing it to rend().
c. increment it to move backward through the container.
d. decrement it to move backward through the container.

Answer : c
47 In an associative container
a. values are stored in sorted order.
b. keys are stored in sorted order.
c. sorting is always in alphabetical or numerical order.
d. you must use the sort() algorithm to keep the contents sorted.

Answer : b
48 If you store pointers to objects, instead of objects, in a container
a. the objects won’t need to be copied to implement storage in the container.
b. only associative containers can be used.
c. you can’t sort the objects using object attributes as keys.
d. the containers will often require less memory.

Answer : a,d
1. What is meant by template parameter?
   a) It can be used to pass a type as argument
   b) It can be used to evaluate a type.
   c) It can of no return type
   d) None of the mentioned

   **Answer : a**
2. Which keyword can be used in template?
   a) class
   b) typename
   c) both a & b
   d) function

   Answer : c
3. What is the validity of template parameters?
   a) inside that block only
   b) inside the class
   c) whole program
   d) any of the mentioned

Answer: a
What is the output of this program?
#include <iostream>
using namespace std;
template <class T, int N>
class mysequence
{
    T memblock [N];
public:
    void setmember (int x, T value);
    T getmember (int x);
};
template <class T, int N>
void mysequence<T,N> :: setmember (int x, T value)
{
    memblock[x] = value;
}
template <class T, int N>
T mysequence<T,N> :: getmember (int x)
{
    return memblock[x];
}

int main ()
{
    mysequence <int, 5> myints;
    mysequence <double, 5> myfloats;
    myints.setmember (0, 100);
    myfloats.setmember (3, 3.1416);
    cout << myints.getmember(0) << 'n';
    cout << myfloats.getmember(3) << 'n';
    return 0;
}

a) 100
b) 3.1416
c) 100
3.1416
d) none of the mentioned

Answer : c
5. For declaring function templates, all the arguments in template declaration must be generic (class T).

a. True  b. False

Answer : a
6. A function template can have more than one template argument.

a. True  b. False

Answer : a
7. Template creates different versions of a function at run time.
   a. True  b. False

Answer : a
8. For declaring class templates, some of the arguments in template declaration may be non-generic ordinary data types.

a. True  b. False

Answer : a
9. In the template `<class T>` declaration, T stands for
a) an integer data type
b) a generic data type
c) an arbitrary class
d) a class defined earlier

Answer : b
What is the output of this program?

```cpp
#include <iostream>
using namespace std;
template <class T> 
T max (T a, T b) 
{
    return ( a > b ? a : b);
}

int main ()
{
    int i = 5, j = 6, k;
    long l = 10, m = 5, n;
    k = max(i, j);
    n = max(l, m);
    cout << k << endl;
    cout << n << endl;
    return 0;
}
```

Answer: b
What is the output of this program?

```cpp
#include <iostream>
using namespace std;
template <typename T, int count>
void loopIt (T x)
{
    T val[count];
    for (int ii = 0; ii < count; ii++)
    {
        val[ii] = x++;
        cout << val[ii] << endl;
    }
}

int main ()
{
    float xx = 2.1;
    loopIt<float, 3>(xx);
}
```

Answer: d
12. When defining a function template or class template in C++, one can write this:

a) template <class T>
b) template <typename T>
c) Both (a) and (b)
d) Only (a)

Answer: c
13. A class template in C++ has the following structure
   template <class T> class TemplatedClass {
   }
   What is the meaning of T in the above declaration?
   a. It is a placeholder for a pointer value
   b. It is a placeholder for a type name
   c. It is a string variable
   d. It must be an integer constant
   
   Answer : b
14. Function templates can accept
a) any type of parameters
b) only one parameter
c) only parameters of the basic type
d) only parameters of the derived type

Answer : c
15. What is a function template?
a) creating a function without having to specify the exact type.
b) creating a function with having a exact type.
c) both a & b
d) none of the mentioned

Answer : a
16. Using template....
a) you only need to write one function, and it will work with many different types.
b) it will take a long time to execute
c) duplicate code is increased
d) none of the mentioned

Answer : a
17. What is the Output of the program
#include <iostream>
using namespace std;
template
type Max(type Var1, type Var2)
{
    return Var1 > Var2 ? Var1:Var2;
}
int main()
{
    int p;
p = Max(100, 200);
cout << p << endl;
return 0;
}

Answer : b
What is the output of this program?

```cpp
#include <iostream>
using namespace std;
template<class type>
class Test
{
public:
    Test() {}
    ~Test() {}
type  func1(type var1)
    {
        return var1;
    }
type  func2(type var2)
    {
        return var2;
    }
}
int main()
{
    Test Var1;
    Test Var2;
    cout << Var1.func1(200) << endl;
    cout << Var2.func2(3.123) << endl;
    return 0;
}
```

a) 200
b) 3.123
c) 200
d) 3.123

Answer: a
#include <iostream>
using namespace std;
template<typename T>
class clsTemplate
{
public:
    T value;
    clsTemplate(T i)
    {
        this->value = i;
    }
    void test()
    {
        cout << value << endl;
    }
};
class clsChild : public clsTemplate<char>
{
    public:
        clsChild(): clsTemplate<char>(0)
        {
        }
        clsChild(char c): clsTemplate<char>(c)
        {
        }
        void test2()
        {
            test();
        }
    }
};
int main()
{
    clsTemplate<int> a(42);
    clsChild b('A');
    a.test();
    b.test();
    return 0;
}

Answer : c
What is the output of this program?

```cpp
#include <iostream>
using namespace std;
template <class T>
class A
{
public:
A(int a): x(a) {}
protected:
int x;
};
template <class T>
class B: public A<char>
{
public:
B(): A<char>::A(100)
{
    cout << x * 2 << endl;
}
};

int main()
{
    B<char> test;
    return 0;
}
```

Answer: b
What is the output of this program?

```cpp
#include <iostream>
using namespace std;
template <class T>
class A
{
public:
A(int a): x(a) {}
protected:
int x;
};
template <class T>
class B: public A<char>
{
public:
B(): A<char>::A(100)
{
    cout << x * 2 << endl;
}
};
int main()
{
    B<char> test;
    return 0;
}
```

a) 100  
b) 200  
c) error  
d) runtime error

Answer: b