Question Name	Question	Option 1	Option 2	Option 3	Option 4 (correct answer)					
OOP_UNIT_1_01	Data members and member functions of a class in C++ program are by default	protected	public	None	private					
OOP_UNIT_1_02	Which operator is used to allocate an object dynamically of a class in C++?	Scope resolution operator	Conditional operator	Membership access	New operator					
OOP_UNIT_1_03	Which is used to define the member function of a class externally?	:	?	<<	::					
OOP_UNIT_1_04	In C++, an object cannot be created for	Derive Class	Both	None	An Abstract Class					
OOP_UNIT_1_05	Which is the way of creating an object of a class called Car is	Car obj;	Car *obj = new Car();	None	Both					
OOP_UNIT_1_06	In C++, Class object created statically(e.g. Car obj; and dynamically (Car *obj = new Car() ;) are stored in memory	Heap, Stack	Stack Only	Heap only	Stack, heap					
OOP_UNIT_1_07	In C++ programming, cin is a/an	Function	Macro	operator	Object					
OOP_UNIT_1_08	Pick out the correct statement	A friend function may be a member of another class	A friend function may not be a member of	None	A friend function may or may not be a member of another class					
OOP_UNIT_1_09	In friend function, Where does keyword 'friend' should be placed?	function definitio	main function	None	function declaration					
OOP_UNIT_1_10	When you create an object of a class, what is called automatically	Destructor	Copy constructor	Assignment ope	Deafult Constructor					
OOP_UNIT_1_11	Inline is a ________________	Class	Variable	Object	Keyword					
OOP_UNIT_1_12	______________ is an instance of class	Pointer	code	variable	Object					
OOP_UNIT_1_13	Public, private, protected are ______	Identifiers	Variables	Data Members	Access Specifiers					
OOP_UNIT_1_14	The ______ access specifies allows functions or data to be accessible to other parts of the	private	protected	all	public					
OOP_UNIT_1_15	The variable myNum has the value 5. How to print a variable to the screen?	. cout<< "My numb	cout<< "My numb	cout<< My numb	cout<< "My number is" << myNum << endl;.					
OOP_UNIT_1_16	________ is a stream connected to standard	cin	fin	none	cout					
OOP_UNIT_1_17	_________ is the symbol that precedes the destructor	*	&	@	~					
OOP_UNIT_1_18	The parameters specified in the function call are known as ____________ parameters	Formal	Value	Original	Actual					
OOP_UNIT_1_19	________ is the process of using the same name for two or more functions	Default Function	Constant Functio	None	Function Overloading					
OOP_UNIT_1_20	What does your class can hold?	Data	Functions	None	Both					
OOP_UNIT_1_21	Which of the following statements is correct about the constructors and destructors?	Destructors can ta	Constructors and	None	Constructors can take arguments but destructors cannot					

OOP_UNIT_1_22	Can constructors be overloaded?	FALSE			TRUE					
OOP_UNIT_1_23	Every class has at least one constructor function, even when none is declared.	FALSE			TRUE					
OOP_UNIT_1_24	The default access level assigned to members of a class is ______	Publlic	Protected	None	Private					
OOP_UNIT_1_25	Which type of variable of class has only one unique value for all the objects of that same	Friend	This	None	Static					
OOP_UNIT_1_26	What is a constructor?	A class automatica	A class automati	A function auto	A function automatically called whenever a new object of this class is created.					
OOP_UNIT_1_27	Under what conditions a destructor destroys an object?	Scope of existence	Object dynamical	None	Both					
OOP_UNIT_1_28	When class B is inherited from class A, what is the order in which the constructers of those classes are called	Class B first Class	Class B's only as	Class A's only as	Class A first Class B next					
OOP_UNIT_1_29	Variables declared in the body of a particular member function are known as data members and can be used in all member functions of	TRUE			FALSE					
OOP_UNIT_1_30	In a class definition, data or functions designated private are accessible	to any function in	only to public me	only if you kno	to member functions of that					
OOP_UNIT_1_31	Which of the following determines how your program will be used by other program?	Private	Protected	None of These	Public					
OOP_UNIT_1_32	Which of the following is true?	All objects of a class share all data members of class	Both	None of These	Objects of a class do not share non-static members. Every object has its own					
OOP_UNIT_1_33	What happens when a class with parameterized constructors and having no default constructor is used in a program and we create an object that needs a zero-argument	Run-Time Error	Preprocessing Er	None of These	Compile-time error					
OOP_UNIT_1_34	Which of the following data type does not return anything?	long	short	int	Void					
OOP_UNIT_1_35	The main intention of using inheritance is …………	to help in converting one data type to other	to hide the details of base class	to help in modu	to extend the capabilities of base class					
OOP_UNIT_1_36	Which feature of C++ contain the concept of super class and subclass?	Polymorphism	Encapsulation	Data Binding	Inheritance					
OOP_UNIT_1_37	In object oriented programming the focus is	Function	Pointer	Structure	Data					

OOP_UNIT_1_38	Which of the following feature of object oriented program is false?	Data and Function	Data can be hidd	Object can comm	The focus is on procedures					
OOP_UNIT_1_39	Which of the following functions are performed by a constructor?	Construct new cla	Construct a new	Construct new d	Initialize objects					
OOP_UNIT_1_40	Which of the following is the correct class of the object cout?	fstream	istream	ofstream	ostream					
OOP_UNIT_1_41	C++ does NOT supports the following	Multilevel inherit	Hierarchical inhe	Hybrid inherita	None of these					
OOP_UNIT_1_42	Default return type of C++ main( ) is	void	double	char	int					
OOP_UNIT_1_43	How many objects can be created by a class?	1	2	3	As Many as required					
OOP_UNIT_1_44	Which operator is used to define member function of a class outside the class?	:	:?	~	::					
OOP_UNIT_1_45	The function of a class is called as	Method	None of these	Procedure	Member Function					
OOP_UNIT_1_46	.... constructor is used for copying the object of same class type.	default	Parameterized	None of These	Copy					
OOP_UNIT_1_47	Objects are destroyed in the reverse order of its creation.	FALSE			TRUE					
OOP_UNIT_1_48	Which is NOT type of constructor?	default	Parameterized	Copy	None of These					
OOP_UNIT_1_49	Which is NOT the feature of constructor?	It do not have retu	It cannot be inhe	All of These	It should be declared in Private.					
OOP_UNIT_1_50	.... is a member function with the same name as the class.	Destructor	Friend Function	None of These	Constructor					
OOP_UNIT_1_51	Which of the following header file includes for cin and cout?	fstream	string.h	None of These	iostream					
OOP_UNIT_1_52	cout is a/an __________	Function	Macro	operator	Object					
OOP_UNIT_1_53	In C++ .... operator is used for Dynamic memory allocation.	||	delete	<<	new					
OOP_UNIT_1_54	Overloaded functions	all have the same	None of these		are a group of functions with the same name.					
OOP_UNIT_1_55	The mechanism that binds code and data together and keeps them secure from outside	Polymorphism	Inheritance	Abstraction	Encapsulation					
OOP_UNIT_1_56	Function overloading, operator overloading and virtual functions are the means for	Encapsulation	Inheritance	Abstraction	Polymorphism					
OOP_UNIT_1_57	Static variables can be	cannot be created	a constant	a class	initialized only					
OOP_UNIT_1_58	If the type specifier of parameters of a function is followed by an ampersand (&), that	call by value	pass by array	None of These	call by refrence					
OOP_UNIT_1_59	Which of the following statement is correct with respect to the use of friend keyword inside a class?	We can use friend keyword as a class name	An object may be declared as a friend	A private data member can be declared as a friend	A class may be declared as a friend.					

OOP_UNIT_1_60	Function overloading is also similar to which of the following?	Operator Overload	Destructor Overl	None of These	Constructor Overloading					
OOP_UNIT_1_61	Variables that are declared, but not initialized, contain ________.	zero	blank spaces	nothing	Garbage value					
OOP_UNIT_1_62	Which of the following statement is correct whenever an object goes out of scope?	The parameterized	The default cons	None of These	The default destructor of the					
OOP_UNIT_1_63	Which of the following gets called when an object is being created?	Virtual Function	Destructor	main() function	Constructor					
OOP_UNIT_1_64	How many times a constructor is called in the lifetime of an object	Twice	Thrice	As many times	Only once					
OOP_UNIT_1_65	Which of the following statements are correct?	Destructor is alwa	Constructor is al	Constructor and	Constructor is called either implicitly or explicitly, whereas destructor is					
OOP_UNIT_1_66	Which of the following never requires any	Member Funstion	Friend Function	inline function	Deafult					
OOP_UNIT_1_67	A destructor takes __________ arguments.	one	two	three	No					
OOP_UNIT_1_68	Which of the following implicitly creates a default constructor when the programmer does not explicitly define atleast one	Preprocessor	Linker	Loader	Compiler					
OOP_UNIT_1_69	Which of the following statement is correct about destructors?	A destructor has v	A destructor has	A destructors re	A destructor has no return type.					
OOP_UNIT_1_70	Which of the following statement is correct?	A constructor has a	A constructor alw	None of These	A constructor has the same name as the class in which it is present.					
OOP_UNIT_1_71	If the programmer does not explicitly provide a destructor, then which of the following creates an empty destructor.	Preprocessor	Linker	Loader	Compiler					
OOP_UNIT_1_72	How many default constructors per class are	Two	Three	Unlimited	Only one					
OOP_UNIT_1_73	A function with the same name as the class, but preceded with a tilde character (~) is called __________ of that class.	Constructor	Function	Object	Destructor					

OOP_UNIT_1_74	What is the output of this program?
	#include <iostream h>
	using namespace std;
	int main()
	{
	        void a = 10, b = 10;
	        int c;
	        c = a + b;
	        cout << c;
	        return 0;
	}
		Run Time Error	20	None of These	Compile time error					

OOP_UNIT_1_75	#include<iostream>
	using namespace std;
	int main()
	{
	  int x = 10;
	  int& ref = x;
	  ref = 20;
	  cout << "x = " << x << endl ;
	  x = 30;
	  cout << "ref = " << ref << endl;
	  return 0;
	}
		x = 20
		ref = 20
			x = 10
			ref = 30
				x = 30
				ref = 30
					x = 20
					ref = 30					

OOP_UNIT_1_76	class Test {
	    int x;
	};
	int main()
	{
	  Test t;
	  cout <<t.x;
	  return 0;
	}
		Garbage Value	0	1	Compile time error					

OOP_UNIT_1_77	Output of following program?
	#include<iostream>
	using namespace std;
	class Point {
	Point() { cout <<"Constructor called"; }
	};
	
	int main()
	{
	Point t1;
	return 0;
	}
		Constructor called	Run time error	None of These	Compiler Error					

OOP_UNIT_1_78	Output of following program?
	#include<iostream>
	using namespace std;
	class Point {
	public:
	Point() { cout <<"Constructor called"; }
	};
	
	int main()
	{
	Point t1, *t2;
	return 0;
	}
		Constructor called Constructor called	Compiler Error	None of These	Constructor called					

OOP_UNIT_2_01	Which among the following can show polymor	Overloading ||	Overloading +=	Overloading &&	Overloading <<					
OOP_UNIT_2_02	Which among the following is not true for poly	It is feature of OO	Ease in readabili	Helps in redefin	Increases overhead of function definition always					
OOP_UNIT_2_03	Which among the following best describes poly	It is the ability for	It is the ability fo	It is the ability f	It is the ability for a message/data to be processed in more t					
OOP_UNIT_2_04	What do you call the languages that support cla	Class based langua	Procedure Orien	If classes are su	Object-based language					
OOP_UNIT_2_05	If same message is passed to objects of several c	Inheritance	Overloading	Overriding	Polymorphism					
OOP_UNIT_2_06	Which type of function among the following sh	Inline function	Undefined functi	Class member fu	Virtual function					
OOP_UNIT_2_07	Which among the following can not be used for	Static member fun	Predefined opera	Constructor ove	Static member functions					
OOP_UNIT_2_08	A virtual function that has no definition within	Pure static functio	Pure Const funct	Friend function	Pure virtual function					
OOP_UNIT_2_09	If abstract class is inherited by derived class, the	Derived class shou	Derived class als	Objects of deriv	All of these					
OOP_UNIT_2_10	If a class contains pure virtual function, then it	Virtual class	Sealed class	Pure Local class	Abstract Class					
OOP_UNIT_2_11	When a virtual function is redefined by the der	Overloading	Rewriting	All of these	Overriding					
OOP_UNIT_2_12	Which of the followings are true about Virtual f	They must be non	They cannot be f	Constructor Fur	All of these					
OOP_UNIT_2_13	Find the wrong statement/s about Abstract Class	We can't create its	It contains at lea	We can create re	We can't create pointers to an abstract class.					
OOP_UNIT_2_14	Syntax for Pure Virtual Function is	virtual void show(	void virtual shov	void virtual sho	virtual void show()=0					

OOP_UNIT_2_15	is a member function that is declared within a b	static function	friend function	const member fu	virtual function
OOP_UNIT_2_16	Compile time polymorphism in C++ language ar	Operator overload	Function overloa	None	Both
OOP_UNIT_2_17	C++ abstract class can contain	Pure virtual funct	Non-virtual func	Only pure virtua	Both pure virtual and non-virtual function
OOP_UNIT_2_18	Following keyword is used before a function in	override	void	None	virtual
OOP_UNIT_2_19	Usually a pure virtual function	Has complete func	Will never be cal	Will be called or	Is defined only in derived class
OOP_UNIT_2_20	Not using virtual destructor feature in a C++ obj	An Issue in creatir	An issue in callir	Nothing	Memory leak
OOP_UNIT_2_21	Polymorphism is supported by the c++ by using	function overload	operator overloa	virtual function	all of the above
OOP_UNIT_2_22	Compile time polymorphism is supported by	function overload	Operator overloa	None	function overloading & Operator Overloading
OOP_UNIT_2_23	Run time polymorphism is supported by	function overload	operator overloa	None	virtual functions
OOP_UNIT_2_24	Selecting the appropriate overloaded function b	late binding	Both	None	early binding
OOP_UNIT_2_25	object to function binding is done at compile tir	early binding	compile time bin	None	Both
OOP_UNIT_2_26	Run time polymorphism is done by using	function overload	operator overloa	None of these	virtual function
OOP_UNIT_2_27	Which of the following operator cannot be over	scope resolution o	Size of operator (	Conditional ope	All
OOP_UNIT_2_28	Which of the operator cannot be overloaded	>=	&	<=	::
OOP_UNIT_2_29	While performing operator overloading which	Function	Op	None of these	Operator
OOP_UNIT_2_30	We are overloading a unary operator without fr	1	2	None of these	0
OOP_UNIT_2_31	Suppose we are overloading a binary operator w	1	3	None of these	2
OOP_UNIT_2_32	we are overloading a binary operator without fr	2	0	None of these	1
OOP_UNIT_2_33	What is true about the operator overloading	with friend functi	with friend func	None of these	Both
OOP_UNIT_2_34	allows you to give special meaning to some ope	function overload	virtual function	None of these	operator overloading
OOP_UNIT_2_35	Reusability is supported by following feature	polymorphism	message passing	Object	inheritance
OOP_UNIT_2_36	Deriving a new class from a base class is known	polymorphism	Abstraction	Encapsulation	inheritance
OOP_UNIT_2_37	Base class is also known as___.	super class	parent class	None of these	Both
OOP_UNIT_2_38	Child class is also known as	super class	Known class	None of these	Sub Class
OOP_UNIT_2_39	Pick the correct option	We can make the i	Both	None of these	We can not make the instance of the abstract class
OOP_UNIT_2_40	What is default access specifier for class membe	public	protected	None of these	private
OOP_UNIT_2_41	What types of inheritance are supported by c++?	single	multiple	multilevel	All
OOP_UNIT_2_42	Choose the correct option.	A base class may h	Derived class ma	None of these	Both
OOP_UNIT_2_43	The ability of function or operator to act in diffe	inheritance	Encapsulation	None of these	polymorphism
OOP_UNIT_2_44	Which is the correct class defination for class C	Class C:A,B	Class C:: public A	Class C::A,B	Class C:public A,public B
OOP_UNIT_2_45	How to deaclare class tier which is derived fron	Class wheel:publi	Class rubber:pub	None of these	Class Tier:public wheel, public rubber
OOP_UNIT_2_46	Suppose class derived is derived from a class Ba	Base:display()	Display()	Can make such a	Base ::display()
OOP_UNIT_2_47	The base class will offer_____	more specific obje	empty templates	none of these	more generalized version of its derived class
OOP_UNIT_2_48	When base class pointer points to derived class	it can access only c	it can both base c	None of these	it can access only base class members
OOP_UNIT_2_49	Class Test:public A, public B is an example of m	FALSE	Can't Say		TRUE
OOP_UNIT_2_50	When the object of derived class expire, first th	derived class cons	base class destru	None of these	derived class destructor , base class destructor
OOP_UNIT_2_51	How many objects can be created from an abstra	One	Many	None of these	Zero
OOP_UNIT_2_52	Which of the following statement is correct?	A constructor of a	Constructor cann	None of these	Both
OOP_UNIT_2_53	Destructor calls are made in which order of the	Forward order	Depends on how	None of these	Reverse order
OOP_UNIT_2_54	Which of the following is correct about the stat	Both True	Ony I true	Only II true	Both Flase
OOP_UNIT_2_55	Which of the following is a mechanism of stati	Function Overload	Operator Overloa	Templates	All

OOP_UNIT_2_56	Pick out the correct statement about override.	Overriding has dif	Both	None of these	Overriding refers to a derived class function that has the sa
OOP_UNIT_2_57	Pick out the correct option	We can make an in	Both	None of these	We cannot make an instance of an abstract base class
OOP_UNIT_2_58	What is meant by pure virtual function	Function which do	Function which (	None of these	Function which does not have definition of its own
OOP_UNIT_3_01	Which of the following header file does not exi	<iostream>	<fstream>	<string>	<sstream>
OOP_UNIT_3_02	A file stream that receives or reads data from file into program is referred to as____.	cout	cin	ouput file strear	input file stram
OOP_UNIT_3_03	Which one of the following statement connects the file stream object named fin with	fin.open(test.txt);	fin="test.txt"	None of these	fin.open(“test.txt”);
OOP_UNIT_3_04	Which of these are binary file operations ?	get() & put()	Both of these	None of these	read() & write()
OOP_UNIT_3_05	Which of the funtions return the current position of get pointers in bytes ?	tellp()	seekg()	seekp()	tellg()
OOP_UNIT_3_06	Which of the funtions return the current position of put pointers in bytes ?	tellg()	seekg()	seekp()	tellp()
OOP_UNIT_3_07	From the following which functions does the bu	open()	copy()	None of these	close()
OOP_UNIT_3_08	Which of the following will act as intermediate between I/O operations and	memory	RAM	None of these	stream buffer
OOP_UNIT_3_09	Choose the correct option I. stream acts as an interface between file and a program II. the read() and write() handles the data in text format	only II is true	both I and II is tr	neither I nor II i	only I is true
OOP_UNIT_3_10	Choose the correct option I. data is transferred between console and program II. data is transferred between the program	only I is true	only II is true	neither I nor II i	both I and II is true
OOP_UNIT_3_11	Which of the following is the correct format of reading the binary input from file ?	infile.read(sizeof(v	infile.read(char*,	read(char*,sized	infile.read((char*)&v,sizeof(v))
OOP_UNIT_3_12	Which operator is used to insert data into the fil	>>	<	None of these	<<
OOP_UNIT_3_13	For reading with cin object we need to include_	fstream	stdio.h	None of these	iostream
OOP_UNIT_3_14	Which of the following object is used to get the	cout	coi	None of these	cin
OOP_UNIT_3_15	Which operator is used for input stream?	<<	>	<	>>
OOP_UNIT_3_16	The eof() is used to	check end of sente	check end of pro	None of these	check end of file
OOP_UNIT_3_17	If we create a file by ifstream then we can ___	write data to file	read from as wel	None of these	read data from file
OOP_UNIT_3_18	A file to be opened for reading requires the ____	ofstream	iostream	None of these	ifstream
OOP_UNIT_3_19	A file to be opened for writing requires the ____	ifstream	iostream	None of these	ofstream
OOP_UNIT_3_20	The input and output streams cin and cout are _______ therefore have _________.	functions,objects	structure,functio	None of these	objects,member functions
OOP_UNIT_3_21	Which one of the following reads a single chara	getw()	cin()	put()	get()
OOP_UNIT_3_22	.__________ is used to write a single character to	get()	cout()	cin	put()
OOP_UNIT_3_23	If we have object from ofstream class, then defa	input	append	None of these	output
OOP_UNIT_3_24	If we have object from ifstream class, then defa	output	append	None of these	input

OOP_UNIT_3_25	Streams that will be performing both input and output operations must be declared as	ifstream	ofstream	None of these	fstream					
OOP_UNIT_3_26	__________ is return type of is_open() function	int	char	float	bool					
OOP_UNIT_3_27	Which among following is used to open a file ir	ios::in	ios::out	ios::app	ios::binary					
OOP_UNIT_3_28	What is use of eof() ?	Returns true if a fi	Returns true if a	Returns true if a	Returns true if a file open for reading has reached the end.					
OOP_UNIT_3_29	Which functions allow to change the location of the get and put pointers ?	tellg() & tellp()	sg() & sp()	None of these	seekg() &seekp()					
OOP_UNIT_3_30	Which is correct syntax for, position n bytes bac	fileObject.seekg(ios:	fileObject.seekg(n	fileObject.seekg(	fileObject.seekg(n, ios::end);					

OOP_UNIT_3_31	#include<iostream>
	#include <fstream>
	using namespace std;
	int main ()
	{
	ofstream outfile ("test.txt");
	for (int n = 0; n < 100; n++)
	{
	outfile << n;
	outfile.flush();
	}
	cout << "Done";
	outfile.close();
	return 0;
	}
		Syntax Error	Runtime Error	None of these	Done					

OOP_UNIT_4_01	What is meaning of template parameter?	Used to evaluate a	It can of no retur	None of these	It is used to pass a type as argument					
OOP_UNIT_4_02	______Keyword/s is/are used in template	class	typename	None of these	Both					
OOP_UNIT_4_03	What is scope of template parameter?	Inside a block only	Inside the class c	Throughout pro	All					
OOP_UNIT_4_04	Template are of types	Function template	Class template	None of these	Both					
OOP_UNIT_4_05	Class template can be created using________synta	template<class T>	template<class T1	None of these	Both					
OOP_UNIT_4_06	Syntax for creating a function template is	template<typenam	template<class T:	None of these	Both					
OOP_UNIT_4_07	Pick up the correct statement	i only	i and ii only	ii and iii only	i, ii and iii					
OOP_UNIT_4_08	An exception is typically caused by	Syntax error	the creator of a c	the programmer	a runtime error					
OOP_UNIT_4_09	Statements that might cause an exception must l	TRUE	Can't Say		FALSE					
OOP_UNIT_4_10	Exceptions are thrown	from the catch blo	from a throw sta	from the point c	from a throw statement to a catch block.					
OOP_UNIT_4_11	Pick up the correct statement from following 1.Exception handling is not supported c++ 2.Template support generic programming in c+ + 3.overloading of function template is possible in c++ [illegible]	2 and 3 only	3 and 4 only	1, 2 and 3 only	2, 3 and 4 only					

OOP_UNIT_4_12	We can restrict a function to throw only a set of specified exceptions by adding a throw specification clause to the function definition.	FALSE			TRUE					
OOP_UNIT_4_13	We may also use non-type parameters such basic or derived data types as arguments	FALSE			TRUE					
OOP_UNIT_4_14	It is also possible to make a single catch statement to catch all types of exceptions	FALSE			TRUE					
OOP_UNIT_4_15	We can place two or more catch blocks together to catch and handle multiple types of exceptions thrown by a try blocks	FALSE			TRUE					
OOP_UNIT_4_16	When an exception is not caught	Program is go in wait condition	Program works fine way	None of These	Program is aborted					
OOP_UNIT_4_17	While specifying the exceptions, the type-list specifies the________ that may be thrown.	How many exceptions	Both of these	None of These	Type of exception					
OOP_UNIT_4_18	If the thrown exception will not be caught by any catch statement then it will be passed to next outer try/catch sequence for processing	FALSE			TRUE					
OOP_UNIT_4_19	Irrespective of exception occurrence, catch handler will be always executed	Yes			No					
OOP_UNIT_4_20	Pick up the correct statement i) Catch statement be placed immediately after try block ii) It can have multiple parameters iii) There must be multiple catch handler for a try block iv) Generic catch statement we can placed anywhere in program	i and ii	i and iv	i , ii and iii	i and iii					
OOP_UNIT_4_21	What is STL	Standard Tree Libr	Standard Term L	None	Standard Template Library					
OOP_UNIT_4_22	Can we write a throw statement inside catch	No			Yes					
OOP_UNIT_4_23	We can define our own exceptions in c++	No			Yes					
OOP_UNIT_4_24	Can we write try block within try block	No			Yes					
OOP_UNIT_4_25	______is a generic catch handler(catchall)	catch(---)	catch(-,-)	catch(void)	catch(...)					
OOP_UNIT_4_26	In catch statement we have multiple parameters	Yes			No					
OOP_UNIT_4_27	Function template is applicable for_________	For that class only	Both of these	None of these	For function only					
OOP_UNIT_4_28	Template is a way creating generalize functions and classes which are applicable for	FALSE			TRUE					
OOP_UNIT_4_29	Exception can be handle if_______	Throwing argument is not match with catch	Exception is not thrown	None of these	Throwing argument is match with catch block					
OOP_UNIT_4_30	Which statement we have to use rethrowing	throw (exception)	Both of these	None of These	throw					
OOP_UNIT_4_31	How the exception is throw	throw	throw (exception	throw exception	All of these					

OOP_UNIT_4_32	Run time error is known as ______	Logical Error	Run time Error	None of these	Exception					
OOP_UNIT_4_33	When template is defined with parameter that would be replaced by specified _______at the time of actual use of class or function.	Keyword	Operator	None of these	Data type					
OOP_UNIT_4_34	__________is used to perform the generic programming.	Class	Function	Inheritance	Template					
OOP_UNIT_4_35	Which is used to get the input during	cout	template	all of these	cin					
OOP_UNIT_4_36	Which statement is used to catch all types of exceptions?	catch()	catch	catch(Test T)	None of these					
OOP_UNIT_4_37	An Exception is thrown using _____________keyword in cpp	throws	threw	thrown	throw					
OOP_UNIT_4_38	Which of the following is used to check the error in the block?	throw	catch	None of these	try					
OOP_UNIT_4_39	In C++ program handling, a try block must be followed by _____catch blocks	exactly one	exactly two	None of these	one or many					
OOP_UNIT_4_40	Which block should be placed after try block ?	any statement			catch block					
OOP_UNIT_4_41	Which of the following causes an exception	Syntax error	Missing parenthesis in main()	Calling a function which is not present	run time error					
OOP_UNIT_4_42	Which keyword can be used with template?	operator	Both of these	None of these	typename					
OOP_UNIT_4_43	Pick up the correct statement	To throw exception we have to use catch	We can not have multiple throwing	None of these	Error occurring code is placed in try block					
OOP_UNIT_4_44	Which of the following is NOT sequence	vector	list	dequeue	map					
OOP_UNIT_4_45	Which of the following container is NOT	list	dequeue	None of these	vector					
OOP_UNIT_4_46	Which of the following is NOT correct function for vector container	push_back()	pop_back()	begin()	push_front()					
OOP_UNIT_4_47	What is/are Components of STL	Container	Algorithms	Iterators	All of these					
OOP_UNIT_4_48	Types of Containers	Sequence Container	Associative Container	Derived Container	All of these					
OOP_UNIT_4_49	Which of the following container allows random access	vector	list	dequeue	Both vector and dequeue					
OOP_UNIT_4_50	How to create container object for integer type	list obj	vector <int>obj	list<char>L	list<int>obj					
OOP_UNIT_4_51	Containers are used	To calculate size	To perform operations	To manupulate	To hold Data					
OOP_UNIT_4_52	Iterators are	Used to Store Data	Used to manupulate Data	None of these	Pointers used to traverse data in Container					
OOP_UNIT_4_53	Vector <int> Arr(5,10) means_____.	an array of 10 integers each of size 5	an array of ints, indexed from 5 to 10	None of these	an array of 5 integers each initialized to 10					

OOP_UNIT_4_54	Access to f elements in an STL container is typically handled by_____.	algorithms	refrences	all of these	iterators					
OOP_UNIT_4_55	The size of STL vector is defined to be____.	Total size of data members in the vector class	Number of bytes the vector occupies in memory	None of These	number of elements currently stored in the vector					
OOP_UNIT_4_56	STL is based on following programming paradigm____.	inheritance	polymorphism	None of These	template					
OOP_UNIT_4_57	Which of following statement sets the STL iterator ITR to point to the first element of	V1.begin(ITR);	V1.reset(ITR)	V1.first(ITR)	ITR=V1.begin();					
OOP_UNIT_4_58	Which of following statement sets the STL iterator ITR to point to the last element of	V1.end(ITR)	V1.last(ITR)	None of These	ITR=v1.end();					
OOP_UNIT_4_59	Which of the following data structure is NOT container implemented in STL?	list	stack	vector	Hash Table					

OOP_UNIT_4_60	Consider following code fragrent
	vector <int> arr(10);
	Arr.push_back(100);
	at the end of execution of above statemnet,the size of vector Arr will be
		10	100	None of these	11					

OOP_UNIT_4_61	Following are the main elements of STL. I. Iterators II.exception handlers III.Algorithms	only I and II	only II and III	None of these	only I and III					
OOP_UNIT_4_62	In STL, the common interface between algorithm and containers is provided by means of___.	algorithms	virtual functions	friend functions	iterators					
OOP_UNIT_4_63	If Arr is an STL vector,then the effect of following statement Arr.push_back(x) is to_____.	append x to array at first	add x to array in between	None of these	append x to array at last					
OOP_UNIT_4_64	For STL iterator itr,the statement ++itr does the following:	increase by 1 the size of container pointed to by it	post increament the item to which the	pre-increament the item to which the	advances the iterator to the next item					

OOP_UNIT_4_65	What will be output of program
	#include<iostream>
	using namespace std;
	template<class T>
	void display(T x)
	{
	cout<< “using template x=”<<x<<“\n”;
	}
	void display(int x)
	{
	cout<<“Normal display x=”<<x <<“\n”;
	}
	int main()
	{
	display(2.3);
	display(3);
	diplay(1.1);
	}
		Normal display x=2.3
		Using template x=3
		Normal display x=1.1
			using template x=3
			Normal display x=2.3
			using template x=1.1
				None of these	using template x=2.3
					Normal display x=3
					using template x=1.1					

OOP_UNIT_4_66	What will be output of the a following program
	#include<iostream>
	using namespace std;
	template <class T>
	void display(T x)
	{
	cout<<“Template display:”<<x<< “\n”;
	}
	void display(int x)
	{
	cout<<“Explicit display:”<<x <<“\n”;
	}
	int main()
	{
	display(100);
	display(12.34);
	display(‘c’);
	}
		Template display:100
		Template display:12.34
		Template display: c
			Explicit display:100
			Template display:12.34
			Explicit display: c
				Template display:100
				Template display:12.34
				Template display: c
					Explicit display:100
					Template display:12.34
					Template display: c					

OOP_UNIT_4_67	What will be output of program
	#include <iostream>
	using namespace std;
	int main()
	{
	cout <<"Start\n";
	try {
	cout <<"Inside try block\n";
	throw 100;
	cout << "This will not execute";
	}
	catch (int i) {
	cout <<"Caught an exception -- value is: ";
	cout <<i <<"\n";
	}
	cout <<"End";
	return 0;
	}
		Start
		End
			Start
			Inside try block
			End
				None of the above mentioned	Start
					Inside try block
					Caught an exception -- value is: 100
					End					

OOP_UNIT_4_68	What will be output of following programming
	#include <iostream>
	using namespace std;
	template <class T>
	T GetMax (T a, T b) {
	T result;
	result = (a>b)? a : b;
	return (result);
	}
	int main () {
	int i=5, j=6, k;
	long l=10, m=5, n;
	k=GetMax<int>(i,j);
	n=GetMax<long>(l,m);
	cout <<k << endl;
	cout <<n <<endl;
	return 0;
	}
		5
		5
			10
			10
				Compilation err	6
					10					

OOP_UNIT_4_69	What will be output of following program
	#include <iostream>
	using namespace std;
	template <class T>
	class mypair {
	T a, b;
	public:
	mypair (T first, T second)
	{a=first; b=second;}
	T getmax ();
	};
	template <class T>
	T mypair<T>::getmax ()
	{
	T retval;
	retval = a>b? a : b;
	return retval;
	}
	int main () {
	mypair <int>myobject (100, 75);
	cout << myobject.getmax();
	return 0;
	}
		75	75 100	Compilation err	100					

OOP_UNIT_4_70	What will be output of following program
	#include <iostream>
	#include <exception>
	using namespace std;
	class myexception: public exception
	{
	virtual const char* what() const throw()
	{
	return "My exception happened";
	}
	} myex;
	int main () {
	try
	{
	throw myex;
	}
	catch (exception&e)
	{
	cout << e.what() <<endl;
	}
	return 0;
	}
		Exception happened	Run Time error	Compilation err	My exception happened.					

OOP_UNIT_4_71	#include <iostream>
	using namespace std;
	template <class Type1, class Type2> class myclass
	{
	Type1 i;
	Type2 j;
	public:
	myclass(Type1 a, Type2 b) { i = a; j = b; }
	void show() { cout <<i << ' ' <<j <<'\n'; }
	};
	int main()
	{
	myclass<int, double>ob1(10, 0.23);
	myclass<char, char *>ob2('X', "Templates add power.");
	ob1.show(); // show int, double
	ob2.show(); // show char, char *
	return 0;
	}
		0.23 10
		X Template add power
			10 10
			X template add power
				Compilation err	10 0.23
					X Templates add power.					

OOP_UNIT_4_72	using namespace std;
	void Xhandler(int test)
	{
	try{
	if(test==0) throw test; // throw int
	if(test==1) throw 'a'; // throw char
	if(test==2) throw 123.23; // throw double
	}
	catch(int i) { // catch an int exception
	cout <<"Caught an integer\n";
	}
	catch(...) { // catch all other exceptions
	cout <<"Caught One!\n";
	}
	}
	int main()
	{
	cout <<"Start\n";
	Xhandler(0);
	Xhandler(1);
	Xhandler(2);
	cout <<"End";
	return 0;
	}
	using namespace std;
	void Xhandler(int test)
	{
		Start
		Caught One!
		Caught One!
		Caught One!
		End
			Start
			Caught One!
			Caught an
			integer
			Caught One!
			End
				Compilation err	Start
					Caught an integer
					Caught One!
					Caught One!
					End					

OOP_UNIT_4_73	What will be output of the a following program #include<iostream> using namespace std; template <class T> void display(T x) { cout<<“Template display:”<<x<< “\n”; } void display(int x) { cout<<“Explicit display:”<<x <<“\n”; } int main() { display(100); display(12.34); display(‘c’); }	Template display:100 Template display:12.34 Template display: c	Explicit display:100 Template display:12.34 Explicit display: c	Template display:100 Template display:12.34 Template display: c	Explicit display:100 Template display:12.34 Template display: c					

OOP_UNIT_4_74	What will be output of following program
	#include <iostream>
	using namespace std;
	void Xhandler(int test)
	{
	try{
	if(test) throw test;
	else throw "Value is zero";
	}
	catch(int i) {
	cout << "Caught Exception #: " <<i <<'\n';
	}
	catch(const char *str) {
	cout <<"Caught a string: ";
	cout << str <<'\n';
	}
	}
	int main()
	{
	cout <<"Start\n";
	Xhandler(1);
	Xhandler(2);
	Xhandler(0);
	Xhandler(3);
	cout << "End";
	return 0;
	}
		Start
		Caught Exception #: 1
		Caught Exception #: 2
		Caught Exception #: 0
		Caught Exception #: 3
		End
			Start
			Caught Exception #: 1
			Caught Exception #: 2
			Caught a string: 0
			Caught Exception #: 3
			End
				None of These	Start
					Caught Exception #: 1
					Caught Exception #: 2
					Caught a string: Value is zero
					Caught Exception #: 3
					End					

OOP_UNIT_4_75	What will be the output of following code ?
	#include<iostream>
	using namespace std;
	int main()
	{
	try
	{
	throw 100;
	}
	catch(int a)
	{
	cout<<”Number : “<<a<<endl;
	return 0;
	}
	cout<<”No exception!!! “<<endl;
	return 0;
	}
		Number	No exception!!!	Syntax error	Number:100					

OOP_UNIT_4_76	What will be the output of following code ?
	#include<iostream>
	#include<exception>
	using namespace std;
	int main()
	{
	try
	{
	int *A=new int[1000];
	cout<<”Memory is allocated”;
	}
	catch(exception &e)
	{
	cout<<”Error in memory allocation”<<endl;
	}
	return 0;
	}
		Error in memory a	Syntax Error	Run Time Error	Memory is allocated					