#1 Explained Report Bookmark

The OOPs concept in C++, exposing only necessary information to users or clients is known as

- **A**
  Abstraction
- **B**
  Encapsulation
- **C**
  Data hiding
- **D**
  Hiding complexity

Correct Answer :A

Explanation

Abstraction is a good feature of OOPS, and it shows only the necessary details to the client of an object. Means, it shows only required details for an object, not the inner constructors, of an object. Example – When you want to switch On television, it not necessary to show all the functions of TV. Whatever is required to switch on TV will be showed by using abstract class.

Abstraction and Data hiding are the crucial concepts of the object-oriented programming. Abstraction is a method of expressing the important properties without involving the background details. On the other hand, Data hiding insulates the data from the straight access by the program.

#2 Explained Report Bookmark

Which of the following is an abstract data type?

- A
  Class
- B
  Int
- C
  String
- D
  Double

Correct Answer :A

Explanation

Abstract Data type (ADT) is a type (or class) for objects whose behaviour is defined by a set of value and a set of operations. ... It is called “abstract” because it gives an implementation-independent view. The process of providing only the essentials and hiding the details is known as abstraction

#3 Explained Report Bookmark

Hiding the complexity is known as

- A
  Abstraction
- B
  Encapsulation
- C
  Data hiding
- D
  Composition

Correct Answer :B

Explanation

Encapsulation is a process of combining data members and functions in a single unit called class. This is to prevent the access to the data directly, the access to them is provided through the functions of the class. It is one of the popular feature of Object Oriented Programming(OOPs) that helps in data hiding

#4 Explained Report Bookmark

For Cat and Animal class, correct way of inheritance is

- A
  class Cat: public Animal
- B
  class Animal: public Cat
- C
  Both are correct way
- D
  None is correct way

Correct Answer :A

Explanation

Animal class is implicitly extending Object class and Cat is extending Animal class but due to java inheritance transitive nature, Cat class also extends Object class

#5 Explained Report Bookmark

In a class, encapsulating an object of another class is called

- A
  Inheritance
- B
  Encapsulation

- C
  Composition
- D
  Virtualization

Correct Answer :C

Explanation

In simple word, if a class contains an object of another class as a data member, then it is known as composition. For example,

Class Y, have a class X's object as data member. Means, Y is composed of X.

```
class X {
  public:
     void f1() {

      }
};
class Y{
     X obj;//class object as a data member
public:
     void f2() {

      }

};
```

#6 Explained Report Bookmark

Features not available in C++ object oriented programming is

null

- A
  Virtual destructor
- B
  Virtual constructor
- C
  Virtual function
- D
  All

Correct Answer :B

Explanation

There is no concept of virtual constructor available in C++. However, virtual destructor is available in C++ language to maintain the destructor call from derived to base class. In polymorphic classes, if we don’t use virtual destructor in base class then the derived class destructor will not be called that may cause resource leaks.

#7 Explained Report Bookmark

IS A relationship in C++ is

- A
  Inheritance
- B
  Encapsulation
- C
  Composition
- D
  None

Correct Answer :A

Explanation

Only the Is-a relation is related to inheritance. The OO concept of the Is-a relationship is called specialization. Most OO languages use classes to implement the concept of types and inheritance as only implementation of specialization. In an Is-a relationship A Is-a B if A is a specialization of B. This means that every instance/object of type A can do at least the things that B can do, i.e. it shows the same behavior for the same message (messages are usually implemented as methods). In the case of C++ the previous example for inheritance looks like this:

```
class B {

};

class A : public B {

};
```

#8 Explained Report Bookmark

If you want to write multiple functions in a class with same name, then what C++ feature will you use?

null

- **A**
  Function overriding
- **B**
  Encapsulation
- **C**
  Function overloading
- **D**
  None

Correct Answer :C

Explanation

Compile time polymorphism feature is used i.e. function overloading in C++. Function overloading means, in a class, multiple functions with same name can be written with different signatures, return types etc.

#9 Explained Report Bookmark

Polymorphism types is/are

null

- **A**
  Compile time
- **B**
  Run time
- **C**
  Both A and B
- **D**
  None

Correct Answer :C

Explanation

Two types of Polymorphism are available in C++ object oriented programming i.e. compile time and run time polymorphism. Also, known as early binding and late binding respectively.

Compiler time polymorphism features in C++ language are function overloading, constructor and operator overloading etc. and run time polymorphism is function overriding in inheritance relationship.

#10 Explained Report Bookmark

If I want to have common functions in a class and want to defer implementations of some other functions to derived classes, then we need to use

null

- **A**
  An interface
- **B**
  An abstract class
- **C**
  A friend class
- **D**
  A static class

Correct Answer :B

Explanation

In C++ object oriented programming, abstract class is used for the same, in which we have common or say generalized function in abstract base class and also may have pure virtual function in this class that forces derived classes to implement it.

#11 Explained Report Bookmark

Not using virtual destructor feature in a C++ object oriented programing can cause

- **A**
  Memory leak
- **B**
  An Issue in creating object of the class
- **C**
  An issue in calling base class destructor

- D
  Nothing

Correct Answer :A

Explanation

Virtual destructor is used to maintain the hierarchy of destructor calls for polymorphic classes in inheritance. If we don't use it then it may cause resource leak or memory leak.

#12 Explained Report Bookmark

Which C++ oops feature is related to re-usability?

null

- A
  Encapsulation
- B
  Inheritance
- C
  Abstraction
- D
  None

Correct Answer :B

Explanation

Inheritance feature is used for concept of code re-usability as in inheritance a class can inherit properties and functions of existing well written class.

Abstraction : Provide only necessary information to client code.

Encapsulation: Hide complexity. e.g. by wrapping private class data members by functions. By making internal function of a class private and using interfaces etc.

#13 Explained Report Bookmark

Correct way of creating an object of a class called Car is

null

- **A**
  Car *obj = new Car();
- **B**
  Car obj;
- **C**
  Both A & B
- **D**
  None

Correct Answer :C

Explanation

Both Car obj; and Car *obj = new Car() are valid way to create an object of the class.

#14 Explained Report Bookmark

In C++, Class object created statically(e.g. Car obj; and dynamically (Car *obj = new Car() ;) are stored in memory

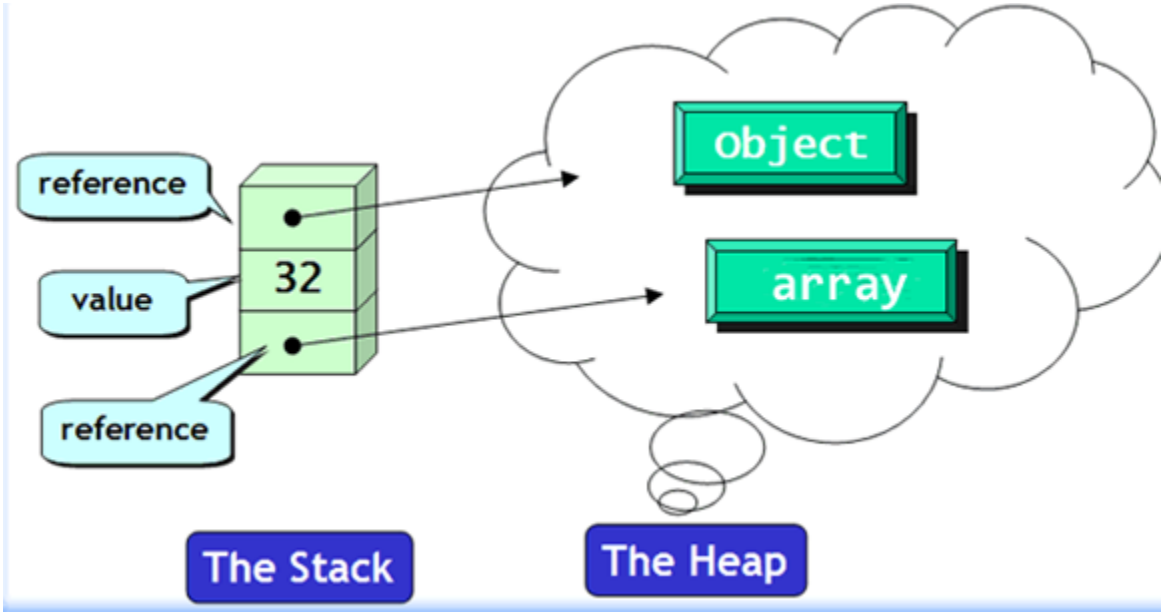
- **A**
  Stack, heap
- **B**
  Heap, heap

- C
  Heap, stack
- D
  Stack, stack

Correct Answer :A

Explanation

The stack is used for static memory allocation and Heap for dynamic memory allocation, both stored in the computer's RAM.



#15 Explained Report Bookmark

True statement about Class and structure in C++ is

- A
  Default access specifier is private in class and public in structure
- B
  Way of creating objects of class and structure are different

- C
  Way of inheriting class and structure are different
- D
  None

Correct Answer :A

Explanation

the default choice for classes defined using class is private , whilst for those defined using struct the default is public

#16 Explained Report Bookmark

In C++ programming, cout is a/an

- A
  Function
- B
  Operator
- C
  Object
- D
  macro

Correct Answer :C

Explanation

cout is a predefined object of a predefined class in C++ programming language.

#17 Explained Report Bookmark

Class allows only one object of it to be created though out the program life cycle

- **A**
  Singleton class
- **B**
  Abstract class
- **C**
  Friend class
- **D**
  All classes

Correct Answer :A

Explanation

A Class allows only one object of it to be created though out the program life cycle is known as singleton class.

#18 Explained Report Bookmark

When you create an object of a class A like A obj ; then which one will be called automatically

- **A**
  Constructor
- **B**
  Destructor
- **C**
  Copy constructor
- **D**
  Assignment operator

Correct Answer :A

Explanation

When an object of a class is created, constructor function of that class is called automatically

#19 Explained Report Bookmark

When you create an object of a derived class in C++

- **A**
  Derived class constructor is called first then the base class constructor
- **B**
  Base class constructor is called first then derived class constructor
- **C**
  base class constructor will not be called
- **D**
  none of the above

Correct Answer :B

Explanation

Base class constructors are always called in the derived class constructors. Whenever you create derived class object, first the base class default constructor is executed and then the derived class's constructor finishes execution.

#20 Explained Report Bookmark

The class in C++ which act only as a base class and object of it cannot be created is

- **A**
  parent class
- **B**
  super class

- C
  abstract class
- D
  none of the above

Correct Answer :C

Explanation

The purpose of an abstract class (often referred to as an ABC) is to provide an appropriate base class from which other classes can inherit. Abstract classes cannot be used to instantiate objects and serves only as an interface. Attempting to instantiate an object of an abstract class causes a compilation error.Abstract Class is a class which contains at least one Pure Virtual function in it. Abstract classes are used to provide an Interface for its sub classes. Classes inheriting an Abstract Class must provide definition to the pure virtual function, otherwise they will also become abstract class.

#21 Explained Report Bookmark

Data members and member functions of a class in C++ program are by default

- A
  protected
- B
  public
- C
  private
- D
  None

Correct Answer :C

Explanation

By default access to members of a C++ class is private. The private members are not accessible outside the class; they can be accessed only through methods of the class. The public members form an interface to the class and are accessible outside the class.

#22 Explained Report Bookmark

Which operator is used to allocate an object dynamically of a class in C++?

- **A**
  Scope resolution operator
- **B**
  Conditional operator
- **C**
  New operator
- **D**
  Membership access

Correct Answer :C

Explanation

New operator is used to allocate an object dynamically of a class in C++.

#23 Explained Report Bookmark

Which is used to define the member function of a class externally

- **A**
  :
- **B**
  ::
- **C**
  #
- **D**
  None

Correct Answer :B

Explanation

Scope resolution operator (::) is used to define the member function of a class externally.

#24 Explained Report Bookmark

In C++, an object cannot be created for

- A
  An interface
- B
  An Abstract class
- C
  A singleton class
- D
  A & B

Correct Answer :D

Explanation

The purpose of an abstract class is to provide an appropriate base class from which other classes can inherit. Abstract classes cannot be used to instantiate objects and serves only as an interface. Attempting to instantiate an object of an abstract class causes a compilation error.

#25 Explained Report Bookmark

base class and derived class relationship comes under

- A
  Inheritance

- B
  Polymorphism
- C
  encapsulation
- D
  None

Correct Answer :A

Explanation

Base class and derived class come under inheritance, one of the C++ oops principle. Also, it is known as parent child relationship.

#26 Explained Report Bookmark

C++ Inheritance relationship is

null

- A
  Association
- B
  IS-A
- C
  Has-A
- D
  None

Correct Answer :B

Explanation

IS A Relationship is related to inheritance in C++.

Lets consider a relation ship between classes i.e. Dog IS A Animal, Apple IS A Fruit, Car IS A Vehicle etc. Here, IS A relationship is valid, but, if we say Animal IS A Dog, then it is wrong. So, Animal class should be the base class and Dog class will be the derived class.

#27 Explained Report Bookmark

Types of inheritance in C++ are

null

- **A**
  Multilevel
- **B**
  Multiple
- **C**
  Hierarchical
- **D**
  All the above

Correct Answer :D

Explanation

All are types of inheritance relationship in C++ oops.

Multilevel Inheritance: When a class is derived from a class which is also derived from another class.

Multiple Inheritance: A class inherits multiple class. or say, A class has more than one base class.

Hierarchical Inheritance: When multiple classes derived from a same base class.

#28 Explained Report Bookmark

Inheritance allow in C++ Program?

null

- **A**
  Class Re-usability
- **B**
  Creating a hierarchy of classes
- **C**
  Extendibility
- **D**
  All

Correct Answer :D

Explanation

Advantage of inheritance are like re-usability- You can re-use existing class in a new class that avoid re-writing same code and efforts.

We can make an application easily extensible.

#29 Explained Report Bookmark

Functions that can be inherited from base class in C++ program

null

- **A**
  Constructor
- **B**
  Destructor
- **C**
  Static function

- D
  None

Correct Answer :D

Explanation

In C++, constructor and destruction of a class cannot be inherited to derived class. However, when we create the object of derived class then constructor of the base class is called automatically. You can read order of execution of constructors and destructors call in c++ inheritanc .

#30 Explained Report Bookmark

_____ members of base class are inaccessible to derived class

- A
  Private
- B
  Protected
- C
  Public
- D
  None

Correct Answer :A

Explanation

The class members declared as private can be accessed only by the functions inside the class. They are not allowed to be accessed directly by any object or function outside the class. Only the member functions or the friend functions are allowed to access the private data members of a class.

#31 Explained Report Bookmark

Accessing functions from multiple classes to a derived class is known as

- A
  multiple inheritance
- B
  single inheritance
- C
  Hybrid inheritance
- D
  multilevel inheritance

Correct Answer :A

Explanation

Multiple Inheritance is a feature of C++ where a class can inherit from more than one classes.

#32 Explained Report Bookmark

In inheritance, order of execution of base class and derived class destructors are

null

- A
  Base to derived
- B
  Derived to base
- C
  Random order
- D
  none

Correct Answer :B

Explanation

In inheritance, execution order of constructors are always from base to derived and destructors call order is in reverse i.e. from derived to base. In polymorphic classes, means the class that contain virtual functions, we need to make destructor virtual in base class. Other wise the derived class destructor will not be called

#33 Explained Report Bookmark

Compile time polymorphism in C++ language are

- **A**
  Operator overloading
- **B**
  Function overloading
- **C**
  Function overriding
- **D**
  Both A & B

Correct Answer :D

Explanation

Overloading is compile time polymorphism where more than one methods share the same name with different parameters or signature and different return type. Overriding is run time polymorphism having same method with same parameters or signature, but associated in a class & its subclass

#34 Explained Report Bookmark

Abstract class can contain

- A
  Pure virtual function
- B
  Non-virtual function
- C
  Only pure virtual function
- D
  Both pure virtual and non-virtual function

Correct Answer :D

Explanation

Abstract class can have normal functions and variables along with a pure virtual function. Abstract classes are mainly used for Upcasting, so that its derived classes can use its interface. Classes inheriting an Abstract Class must implement all pure virtual functions, or else they will become Abstract too

#35 Explained Report Bookmark

False statements about function overloading is

- A
  Defining multiple functions with same name in a class is called function overloading
- B
  Overloaded function must differ in their order and types of arguments
- C
  Overloaded functions should be preceded with virtual keyword
- D
  No statement is false

Correct Answer :C

Explanation

A '*virtual*' is a *keyword preceding* the normal declaration of a *function*. ... same name but different prototypes, C++ *will* consider them as the *overloaded functions*.

#36 Explained Report Bookmark

Following keyword is used before a function in a base class to be overridden in derived class in C++

- **A**
  override
- **B**
  virtual
- **C**
  void
- **D**
  none

Correct Answer :B

Explanation

A '*virtual*' is a *keyword preceding* the normal declaration of a *function*. ... same name but different prototypes, C++ *will* consider them as the *overloaded functions*.

#37 Explained Report Bookmark

Which of the following cannot be overloaded in C++?

null

- A
  Increment operator
- B
  Constructor
- C
  Destructor
- D
  New and delete operator

Correct Answer :C

Explanation

Destructor of a class cannot be overloaded in C++ programming. Increment operator, constructor and new and delete can be overloaded.

#38 Explained Report Bookmark

Which is the correct declaration of pure virtual function in C++

- A
  virtual void func = 0;
- B
  Virtual void func() = 0;
- C
  virtual void func(){0};
- D
  void func() = 0;

Correct Answer :B

Explanation

A pure virtual function (or abstract function) in C++ is a virtual function for which we don’t have implementation, we only declare it. A pure virtual function is declared by assigning 0 in declaration

#39 Explained Report Bookmark

In a class, pure virtual functions in C++ is used

- **A**
  To create an interface
- **B**
  To make a class abstract
- **C**
  To force derived class to implement the pure virtual function
- **D**
  All the above

Correct Answer :D

Explanation

Click Here for More Detail

#40 Explained Report Bookmark

Which public member of a base class cannot be inherited?

null

- **A**
  Constructor
- **B**
  Destructor
- **C**
  Both A & B

- **D**
  None

Correct Answer :C

Explanation

In C++ constructor and destructor both cannot be inherited to child class.

#41 Explained Report Bookmark

Number of virtual table created for a base and a derived class is

- **A**
  0
- **B**
  1
- **C**
  2
- **D**
  3

Correct Answer :C

Explanation

ach class will have its own virtual table. While each object has its own virtual pointer. There will be 2 vtables , one for class A and one for class B

#42 Explained Report Bookmark

Interface class in C++ is created by

null

- A
  Using interface keyword before class
- B
  Using pure virtual function
- C
  Using pure virtual function and virtual function both
- D
  Using class keyword

Correct Answer :B

Explanation

C++ does not have keyword interface like C# or java

#43 Explained Report Bookmark

Which statements are true about an abstract class

- A
  Abstract class has at least one pure virtual function.
- B
  Pointer for an abstract class can be created
- C
  Object of an abstract class cannot be created.
- D
  All are correct.

Correct Answer :D

Explanation

All the points mentioned in the question are correct about abstract class

An abstract class is a class in C++ which have at least one pure virtual function. Abstract class can have normal functions and variables along with a pure virtual function. Abstract class cannot be instantiated, but pointers and references of Abstract class type can be created.

we can create a pointer to an abstract class, which could be actually pointing to the objects of its derived classes. In this way, a container of base class pointers can be created which can also store derived class objects

abstract class is an incomplete class (incomplete in the sense it contains abstract methods without body and output) we cannot create an instance or object; the same way you say for an interface.

#44 Explained Report Bookmark

Run time binding is related to

- **A**
  Function overriding
- **B**
  Operator overloading
- **C**
  Both A & B
- **D**
  None

Correct Answer : A

Explanation

A runtime binding is something that the compiler has no idea where the method comes from. It still knows about the method ... If your code calls a method that the compiler knows about at compile time, then this is compile time binding. The method could be a global method, or a class method.

#45 Explained Report Bookmark

Which function cannot be overloaded in C++

- **A**
  Constructor
- **B**
  Class destructor
- **C**
  Both a & b
- **D**
  None

Correct Answer :B

Explanation

we cannot overload a destructor of a class in C++ programming. Only one empty destructor per class should be there. ... Destructor in C++ neither takes any parameters nor does it return anything. So, multiple destructor with different signatures are not possible in a class

#46 Explained Report Bookmark

Operators can be overloaded in C++ is/are

- **A**
  New

- **B**
  Delete
- **C**
  ++
- **D**
  All can be overloaded

Correct Answer :D

Explanation

All can be overloaded in C++ programming language.

#47 Explained Report Bookmark

In C++ code , variables can be passed to a function by

null

- **A**
  Pass by value
- **B**
  Pass by reference
- **C**
  Pass by pointer
- **D**
  All the above

Correct Answer :D

Explanation

In C++ programming, all pass by value, reference and pointer are used.
Note that pass by reference in C Programming language is not available.

However, sometimes people use the term pass by reference instead of pass by address or pass by pointers.

#48 Explained Report Bookmark

Constant function in C++ can be declared as

null

- **A**
  Void display()
- **B**
  void display() const
- **C**
  const void display()
- **D**
  void const display()

Correct Answer :B

Explanation

Constant function in a class is used to prevent modification of class member variables inside its body. When we only want to read member variable and use it in function body with no modification then we should use const function

#49 Explained Report Bookmark

True statements about inline function in given C++ code example is/are

I)Static function of a class can be called by class name using scope resolution operator i.e. : :

(II)Static function can receive both static and non-static data members of a class

(III)Static function is not the part of an object of a class

- **A**
  I and II
- **B**
  I only
- **C**
  I and III
- **D**
  All

Correct Answer :C

Explanation

Statement 1: Because static member functions are not attached to a particular object, they can be called directly by using the class name and the scope resolution operator. (TRUE)

Statement 2 : No, Static function of a class in C++ cannot access non-static variables, but, it can access static variable only. However, non-static member function can access static and non-static variable both. Static function is not associated with class object, means without object using class name only it can be called (FALSE)

Statement 3 : Because static member functions are not attached to a particular object, they can be called directly by using the class name and the scope resolution operator.

(TRUE)

so 1 and 3 statement are true only

#50 Explained Report Bookmark

Which of the following functions are provided by compiler by default if we don’t write in a C++ class?

- **A**
  Copy constructor
- **B**
  Assignment
- **C**
  Constructor
- **D**
  Only A & C

Correct Answer :C

Explanation

If we don't write any constructor in a class including c++ copy constructor then default constructor provided by compiler will be called when we create an object of the class E.g. in below class A, constructor is not written, hence default constructor will be called in the program

```
class A{

public:

     void function(){

     }

};
```

```
int main() {

    A ob; // default constructor will be called, generated by
compiler

    return 0;

}
```

#51 Explained Report Bookmark

Which function can be called without using an object of a class in C++

- **A**
  Static function
- **B**
  Inline function
- **C**
  Friend function
- **D**
  constant function

Correct Answer :A

Explanation

static member functions in a class do exact throughout the program and can be used without creating an object of the class in which static member functions are defined

#52 Explained Report Bookmark

True and false about inline function statements in C++

```
class A{

public:

     void func1(){

     }

     void func2();

};

inline void A::func2(){



}
```

- **A**
  Func2 only is inline function
- **B**
  Func1 is inline function
- **C**
  Func1 and Func2 both are inline functions
- **D**
  None of the above is inline

Correct Answer :C

Explanation

func1() and func2() both are inline functions.

There are two options to offer to the compiler to make a class function inline:

(1) Defining a function in the declaration of the class (in a header file)

```
class A{

public:

   // is implicit inline

  void func1(){

  }

}
```

(2) Using the inline keyword explicitly in the definition of the function (in a header file)

```
 // is explicit inline

inline void A::func2(){



}
```

#53 Explained Report Bookmark

Which function can be called without using an object of a class in C++

- A
  Inline function
- B
  Static function
- C
  constant function
- D
  Virtual function

Correct Answer :B

Explanation

Because static member functions are not attached to a particular object, they can be called directly by using the class name and the scope resolution operator.

#54 Explained Report Bookmark

Which of the following function declaration using default arguments is correct?

- A
  int foo(int x, int y =5, int z=10)
- B
  int foo(int x=5, int y =10, int z)
- C
  int foo(int x=5, int y, int z=10)
- D
  all are correct

Correct Answer :A

Explanation

Default arguments in a function in C++ program is initialized from right to left.

#55 Explained Report Bookmark

Which function cannot be overloaded in C++ program?

- **A**
  Virtual function
- **B**
  member function
- **C**
  Static function
- **D**
  All can be overloaded

Correct Answer :C

Explanation

Static functions cannot be overloaded in C++ programming.

#56 Explained Report Bookmark

Choose the correct answer for following piece of C++ pseudo code

```
void func(int a, int &b)

{



}
```

```
int main(){

     int a,b;

     func(a,b);

}
```

- **A**
  a is pass by value and b is pass by reference
- **B**
  a is pass by reference and b is pass by value
- **C**
  a is pass by value and b is pass by address
- **D**
  a is pass by value and b is pass by pointer

Correct Answer :A

Explanation

B parameter is not pass by address/pointer but reference. Here is the correct pseudo code for pass by address or say pointer.

```
void func(int a, int *b)

{



}
```

```
int main(){

     int a,b;

     func(a,&b);

}
```

#57 Explained Report Bookmark

Which of the following operators cannot be overloaded ?

- **A**
  >>
- **B**
  ?:
- **C**
  . *
- **D**
  Both B and C

Correct Answer :D

Explanation

There are 4 operators that cannot be overloaded in C++. They are :: (scope resolution), . (member selection), . * (member selection through pointer to function) and ?: (ternary operator).

#58 Explained Report Bookmark

The fields in a structure of a C program are by default

- A
  protected
- B
  public
- C
  private
- D
  none of the above

Correct Answer :B

Explanation

In c language the structure fields are public by default that is they can used by any structre instance variabes

#59 Explained Report Bookmark

The size of the object of the class is

- A
  Sum of sizes of its data and function members
- B
  size of its largest data member
- C
  size of its largest function member
- D
  sum of sizes of its data members

Correct Answer :D

Explanation

The size of the object of the class is the sum of sizes of its data members.

#60 Explained Report Bookmark

Virtual functions ____

- **A**
  Must be friend of the base class
- **B**
  Must be static member of the base class which must be defined
- **C**
  Must be non-static member of the base class which must be defined
- **D**
  Must be static member of the base class which need not be define

Correct Answer :C

Explanation

A virtual function is a member function which is declared within a base class and is re-defined(Overriden) by a derived class. When you refer to a derived class object using a pointer or a reference to the base class, you can call a virtual function for that object and execute the derived class's version of the function.

- Virtual functions ensure that the correct function is called for an object, regardless of the type of reference (or pointer) used for function call.
- They are mainly used to achieve Runtime polymorphism
- Functions are declared with a virtual keyword in base class.
- The resolving of function call is done at Run-time

#61 Explained Report Bookmark

What is true about virtual functions?

- A
  Can be called from constructors but not from destructors
- B
  Can be called from destructors but not from constructors
- C
  Cannot be called from either constructors or from destructors
- D
  Can be called from both constructors and destructors

Correct Answer :C

Explanation

As a general rule, you should never call virtual functions in constructors or destructors. If you do, those calls will never go to a more derived class than the currently executing constructor or destructor. In other words, during construction and destruction, virtual functions aren't virtual.

#62 Explained Report Bookmark

A derived class inherits every member of a base class except

- A
  its constructors and destructors
- B
  its operator = () members
- C
  its friends
- D
  all of the above

Correct Answer :D

Explanation

In principle, a publicly derived class inherits access to every member of a base class except: its constructors and its destructor. its assignment operator members (operator=)

#63 Explained Report Bookmark

Which statement is true in case of a destructor?

- **A**
  A destructor can be overloaded
- **B**
  destructor has to be called explicitly
- **C**
  A destructor does not return any value
- **D**
  A destructor can have parameters

Correct Answer :C

Explanation

Destructor is a member function which destructs or deletes an object.A *destructor* function is *called* automatically when the object goes out of scope. Destructors have same name as the class preceded by a tilde (~). Destructors don’t take any argument and don’t return anything.

#64 Explained Report Bookmark

In which of the following cases is a copy constructor invoked?

- **A**
  When a new object of a class is initialized with the existing object of that class
- **B**
  When a copy of an object is passed by value as an argument to a function

- **C**
  When you return an object of the class by value
- **D**
  All of the above

Correct Answer :D

Explanation

In C++, a Copy Constructor may be called in following cases:

1. When an object of the class is returned by value.

2. When an object of the class is passed (to a function) by value as an argument.

3. When an object is constructed based on another object of the same class.

4. When the compiler generates a temporary object.

#65 Explained Report Bookmark

Static member functions _____

- **A**
  can be used without an instantiation of an object
- **B**
  Can only access static data
- **C**
  A and B
- **D**
  Neither A nor B

Correct Answer :C

Explanation

static member function can be called even if no objects of the class exist and the static functions are accessed using only the class name and the scope resolution operator

#66 Explained Report Bookmark

Predict the output:

```
float x= 3.1496;

cout << setprecision(2) << x;
```

- **A**
  3.14
- **B**
  3.15
- **C**
  3
- **D**
  3.1

Correct Answer :D

Explanation

C++ manipulator setprecision function is used to control the number of digits of an output stream display of a floating- point value. This manipulator is declared in header file <iomanip>.

```
// setprecision example
```

```
#include <iostream>     // std::cout, std::fixed
#include <iomanip>      // std::setprecision

int main () {
  double f =3.14159;
  std::cout << std::setprecision(5) << f << '\n';
  std::cout << std::setprecision(9) << f << '\n';
  std::cout << std::fixed;
  std::cout << std::setprecision(5) << f << '\n';
  std::cout << std::setprecision(9) << f << '\n';
  return 0;
}
```

Output:

```
 3.1416
 3.14159
 3.14159

 3.141590000
```

#67 Explained Report Bookmark

The private data of a class accessed by a friend function

- **A**
  is in the object that invoked the friend
- **B**
  is in the object created by the friend function
- **C**
  is in the object of different class
- **D**
  is in the object sent to the friend function as argument

Correct Answer :D

Explanation

If a function is defined as a friend function then, the private and protected data of a class can be accessed using the function. The complier knows a given function is a friend function by the use of the keyword friend.

#68 Explained Report Bookmark

What is the main purpose of creating an abstract base class ?

- **A**
  reating the dynamic object
- **B**
  dynamic binding
- **C**
  deriving classes
- **D**
  none of these

Correct Answer :C

Explanation

An abstract class can have an abstract method without body and it can have methods with implementation also. ... An abstract class is mostly used to provide a base for subclasses to extend and implement the abstract methods and override or use the implemented methods in abstract class.

#69 Explained Report Bookmark

What is the correct value to return to the operating system upon the successful completion of a program?

- **A**
  -1
- **B**
  1
- **C**
  0
- **D**
  Programs do not return a value.

Correct Answer :C

Explanation

After successful completion of a program 0 is returned to the operating system.

#70 Explained Report Bookmark

_____ is also an operator that is used to get information about the amount of memory allocated for data types and Objects

- **A**
  typedef
- **B**
  ternary

- **C**
  sizeOf
- **D**
  shift

Correct Answer :C

Explanation

SizeOf is also an operator not a function, it is used to get information about the amount of memory allocated for data types & Objects. It can be used to get size of user defined data types too. sizeOf operator can be used with and without parentheses. If you apply it to a variable you can use it without parentheses

#71 Explained Report Bookmark

What will be the output of following code ?

```
#include <iostream>

using namespace std;



int main()

{

  int i=10;

  if(i=20)
```

```
  cout << i ;

  return 0;

}
```

- **A**
  10
- **B**
  20
- **C**
  0
- **D**
  error

Correct Answer :B

Explanation

In the given program, the value of i is changed from 10 to 20 Because of condition inside if statement, where we reassign the value i = 20

#72 Explained Report Bookmark

What will be the output of following code ?

```
#include <iostream>

using namespace std;



void fun()
```

```
{
  static int i = 10;
  i++;
  cout << i;
}


int main()
{
  fun();
  fun();
  fun();
  return 0;
}
```

- **A**
  101112
- **B**
  101112
- **C**
  111213
- **D**
  error

Correct Answer :C

Explanation

static variables create once and they remain throughout the program. Their life is more than a life of a function in which they are declared and defined. They are not destroyed even after function execution is finished in which they are present.

#73 Explained Report Bookmark

What will be the output of following code ?

```
#include <iostream>

using namespace std;



void calc(int x);



int main()

{

  int x = 10;

  calc(x);

  printf("%d", x);
```

```
  return 0;

}



void calc(int x)

{

  x = x + 10 ;

}
```

- **A**
  20
- **B**
  10
- **C**
  0
- **D**
  error

Correct Answer :B

Explanation

In this program, value of x is copied in formal arguments. Changes in formal arguments do not reflect back to actual arguments. Therefore x won’t change.

#74 Explained Report Bookmark

_____ is defined as user defined data type and it also contains functions in it ?

- A
  Object
- B
  Data members
- C
  Class
- D
  Polymorphism

Correct Answer :C

Explanation

Class is a user-*defined* data type, which holds its own data members and member functions, which can be accessed and used by creating an instance or object of that *class*. A *C++ class* is like a blueprint for an object.

#75 Explained Report Bookmark

Function overloading do not depend on _____ .

- A
  order of parameter
- B
  number of parameter
- C
  return values
- D
  differ in type of parameter

Correct Answer :C

Explanation

does not depend on Return Type. Because if return type is different and function name as well as parameter is also same

#76 Explained Report Bookmark

Which of the following statement is correct?

- **A**
  Overloaded functions can have at most one default argument.
- **B**
  An overloaded function cannot have default argument.
- **C**
  All arguments of an overloaded function can be default.
- **D**
  function if overloaded more than once cannot have default argument.

Correct Answer :C

Explanation

the parameters will be changed but not the functionality and the arguments in the function . Hence, all the arguments of function can be default. Which of the following statement is correct? A. Two functions having same number of argument, order and type of argument can be overloaded if both functions do not have any default argument. B. Overloaded function must have default arguments. C. Overloaded function must have default...

#77 Explained Report Bookmark

What will be the output of following code ?

```
#include<iostream>

using namespace std;
```

```
void sum(int x,int y=0)

{

  cout << x+y;

}



int main()

{

  sum(10);

  sum(10,0);

  sum(10,10);

  return 0;

}
```

- **A**
  101020
- **B**
  1020
- **C**
  error
- **D**
  01010

Correct Answer :A

Explanation

The idea behind default argument is simple. If a function is called by passing argument/s, those arguments are used by the function. But if the argument/s are not passed while invoking a function then, the default values are used. Default value/s are passed to argument/s in the function prototype.

#78 Explained Report Bookmark

What does a compiler add to an empty class declaration in C++ ?

- **A**
  destructor
- **B**
  assignment operator
- **C**
  constructor
- **D**
  all of the above

Correct Answer :D

Explanation

The compiler should provide (as and when needed)

1. a constructor
2. a destructor
3. a copy constructor
4. = operator
5. the reference operator(&) - the address

#79 Explained Report Bookmark

What will be the output of following code ?

```
#include<iostream>

using namespace std;



int main()

{

  int i=10;

  int &j=i;

  int k=20;

  j=k;

  cout << i << j<< k ;

  return 0;

}
```

- **A**
  102020
- **B**
  202020
- **C**
  101020

- **D**
  error

Correct Answer :B

Explanation

A reference variable is an alias, that is, another name for an already existing variable. Once a reference is initialized with a variable, either the variable name or the reference name may be used to refer to the variable.

```
#include<iostream>

using namespace std;



int main()

{

  int i=10; // initialize i = 10

  int &j=i;  // A Reference to the variable i

  int k=20;// initialize k = 20

  j=k; //  j = 20   //  j will asign 20 to  both i and j.

  cout << i << j<< k ;

// Now try to print the address of both variables i and j
```

```
    cout<<  &i  <<  &j  <<endl;

    // surprisingly both print the same address and make us
feel that they are

    // alias to the same memory location.



  return 0;

}
```

#80 Explained Report Bookmark

Destructor has the same name as the constructor and it is preceded by _____ .

- **A**
  !
- **B**
  ?
- **C**
  ~
- **D**
  $

Correct Answer :C

Explanation

Destructor has the same name as the constructor and it is preceded by ~.

#81 Explained Report Bookmark

_____ used to make a copy of one class object from another class object of the same class type ?

- **A**
  constructor
- **B**
  copy constructor
- **C**
  destructor
- **D**
  default constructor

Correct Answer :B

Explanation

A copy constructor is a member function which initializes an object using another object of the same class.

#82 Explained Report Bookmark

What will be the outputof following code ?

```
#include <iostream>

using namespace std;



int main()

{
```

```
  const int i = 10;

  const int j = i+10;   // line 4

  cout << i++;   // line 5
}
```

- **A**
  10
- **B**
  11
- **C**
  error at line 4
- **D**
  error at line 5

Correct Answer :D

Explanation

Error will be like this

[Error] increment of read-only variable 'i'.

#83 Explained Report Bookmark

For below code snippet, the public and protected members of Superclass becomes _____ members of Sub class.

```
class subclass : protected Superclass
```

- **A**
  public
- **B**
  private

- C
  protected
- D
  None of the above

Correct Answer :C

Explanation

The public and protected members of superclass become protected members of sub class.

#84 Explained Report Bookmark

What will be the output of following code ?

```
#include <iostream>

using namespace std;



class Animal

{

  public:

  int legs = 4;

};
```

```
class Dog : public Animal

{

  public:

  int tail = 1;

};



int main()

{

  Dog d;

  cout << d.legs;

  cout << d.tail;

}
```

- **A**
  44
- **B**
  43
- **C**
  40
- **D**
  41

Correct Answer :D

Explanation

Here object of Dog is created whose name is d. Dog is a derived class of Animal and Animal is a base class. legs variable is a public variable and it is also accessible from its derived class. d.legs gives 4 and d.tail gives 1.

#85 Explained Report Bookmark

What will be the output of following code?

```
#include <iostream>

using namespace std;



class Base

{

  public:

  Base() { cout << "Base"; }

};



class Derived : public Base

{

  public:
```

```
  Derived(int i) { cout << i; }

};



int main()

{

  Derived d2(10);

  return 0;

}
```

- **A**
  Base10
- **B**
  10
- **C**
  10Base
- **D**
  error

Correct Answer :A

Explanation

When an object is created, a constructor which is present in object’s class is called automatically and if that class is derived class then it’s constructor is called but before execution it calls parent class constructor. Parent class constructor is executed first and then derived constructor is executed.

#86 Explained Report Bookmark

What will be the output of following code?

```
#include <iostream>

using namespace std;



class A

{

  int x;

};



class B : public A

{

  public:

  void show()

  {

    x=10;

    cout << x;
```

```
  }
};



int main()
{
  B b;
  b.show();
  return 0;
}
```

- **A**
  10
- **B**
  0
- **C**
  error
- **D**
  garbage value

Correct Answer :C

Explanation

Compilation error occurs because x is not declared in class B and used in this class without declaration. x of parent class is not accessible in class B because in parent class it is private. Funcitons or methods and variables of

parent class are derived in a derived class but all of them cannot be accessible. They are available but not accessible. There is a difference between accessibility and availability.

#87 Explained Report Bookmark

What will be the output of following code?

```
#include <iostream>

using namespace std;



class Base

{

  public:

  void show()

  {

    cout << "Base";

  }

};



class Derived:public Base
```

```
{
  public:
  void show()
  {
    cout << "Derived";
   }
};


int main()
{
  Base* b;
  Derived d;
  b = &d;
  b->show();
  return 0;
}
```

- A
  Base

- B
  Derived
- C
  Base Derived
- D
  error

Correct Answer :A

Explanation

The method show() is not overridden therefore as per the pointer type respective method is called.

#88 Explained Report Bookmark

Which of the following is a correct way to declare Pure Virtual function?

- A
  virtual void show();
- B
  virtual void show() = 0;
- C
  virtual void show(){};
- D
  virtual void show() = pure;

Correct Answer :B

Explanation

A pure virtual function (or abstract function) in C++ is a virtual function for which we don’t have implementation, we only declare it. A pure virtual

function is declared by assigning 0 in declaration. See the following example.

```
// An abstract class

class Test

{

    // Data members of class

public:

    // Pure Virtual Function

    virtual void show() = 0;



   /* Other members */

};
```

#89 Explained Report Bookmark

What will be the output of following code?

```
#include <iostream>

using namespace std;
```

```
class Base

{

  public:

  ~Base() {cout << "Base Destructor"; }

};



class Derived:public Base

{

  public:

  ~Derived() { cout<< "Derived Destructor"; }

};



int main()

{

  Base* b = new Derived;

  delete b;

  return 0;
```

}

- **A**
  Error
- **B**
  Base Destructor
- **C**
  Derived Destructor Base Destructor
- **D**
  Derived Destructor

Correct Answer :B

Explanation

Base* b = new Derived; is used when one can use a single function to handle all derived class objects. delete is used when an object created by using a new keyword needs to be deleted. Whenever an object is created the constructor of base class is invoked first and then of child class's. Thus combining our whole conclusion we can say that only the Base destructor will be printed on screen.

#90 Explained Report Bookmark

The address of the virtual Function is placed in the _____

- **A**
  Heap
- **B**
  Memory
- **C**
  VTable
- **D**
  Register

Correct Answer :C

Explanation

If a function is declared as virtual in the base class, it will be virtual in all its derived classes. The address of virtual function is placed in the virtual table and the compiler uses virtual pointer to point to the virtual function

#91 Explained Report Bookmark

Which feature allows you to create a Derived class that inherits properties from more than one Base class?

- A
  Multilevel Inheritance.
- B
  Multiple Inheritance.
- C
  Hybrid Inheritance.
- D
  Hierarchical Inheritance.

Correct Answer :B

Explanation

Multiple Inheritance is a feature of C++ where a class can inherit from more than one classes.

#92 Explained Report Bookmark

What does C++ append to the end of a string literal constant?

- A
  A space.
- B
  A number sign (#).

- C
  An asterisk (*).
- D
  A null character.

Correct Answer :D

Explanation

C++: What does C++ append to the end of a string literal constant? a null character string is a char array with a null value (0x00) after the last valid character in the string

#93 Explained Report Bookmark

To hide a data member from the program, you must declare the data member in the _____ section of the class

- A
  Protected
- B
  Confidential
- C
  Hidden
- D
  Private

Correct Answer :D

Explanation

The class members declared as private can be accessed only by the functions inside the class. They are not allowed to be accessed directly by

any object or function outside the class. Only the member functions or the friend functions are allowed to access the private data members of a class.

#94 Explained Report Bookmark

What is the Output of this Program?

```
#include <iostream>

using namespace std;



class sample

{

  private:

  int a, b;

  public:

  void test()

  {

    a = 100;

    b = 200;

  }
```

```
    friend int compute(sample e1);
  };


int compute(sample e1)
{
  return int(e1.a + e1.b) - 5;
}


int main()
{
  sample e;
  e.test();
  cout<<compute(e);
  return 0;
}
```

- **A**
  295
- **B**
  100

- C
  300
- D
  200

Correct Answer :A

Explanation

In main() first the object is created from a class sample.

Class sample contain int variable a , b which are private, test() which is public . These private variables can be accessible only in the public function of the same class so in test() we can access a, b variables.

Class sample have 1 friend function as compute(). Now the compute() function can have direct access to only public and protected members and not to private members but in this case the private members are also accessible due to public test().

Therefore e1.a holds value 100 and e1.b 200. (100+200 -5 = 295)

ANS : 295

#95 Explained Report Bookmark

Null character needs a space of _____ .

- A
  0 Byte
- B
  1 Byte
- C
  3 Byte

- D
  4 Byte

Correct Answer :B

Explanation

The 'null' is a character that holds an ASCII value of zero (0x00). The '\0' notation is used to distinguish it from the character zero, '0', which has an ASCII value of 0x30 (or 48 decimal). So yes, the null character does occupy space - one byte - and holds a value of zero.

#96 Explained Report Bookmark

Which of the following will store the number 320000 as a Float number?

- A
  CounPop = (float) 3.2e5;
- B
  CounPop = (float) 3.2e6;
- C
  CounPop = (float) .32e5;
- D
  CounPop = (float) .32e7;

Correct Answer :A

Explanation

The explicit type casting is done in this question to convert the integer to floating point number

Here 320000 can be written as 3.5 * 10 ^ 5

exponents in floating point number is denoted by e or E

As exponent value is 5 we can write 3.2e5

#97 Explained Report Bookmark

Which of the following statements declare a variable that can contain a decimal number?

- **A**
  Dec payRate;
- **B**
  Decimal hourlyPay
- **C**
  Integer payRate
- **D**
  Float hourlyPay;

Correct Answer :D

Explanation

float is a basic data type but Dec Decimal or Integer is not at all data types

For converting into decimal numbers we can use either float or double. float is available in option.

#98 Explained Report Bookmark

Which of the following type casts will convert an Integer variable named 'amount' to a Double type?

- **A**
  (Double)amount

- B
  (Int to double)amount
- C
  Int to double(amount)
- D
  Int (amount) to double

Correct Answer :A

Explanation

In C, only (double)value works, but the double(value) syntax became legal in C++ to allow primitive casting to appear like a function call. (No, it's not a constructor.) No need for a cast at all, the conversion is implicit. Just assign the int to the double.

#99 Explained Report Bookmark

Which of the following is a string literal constant?

- A
  "Visual C++"
- B
  "137.45"
- C
  "2,365"
- D
  All of above.

Correct Answer :D

Explanation

All of them, as long as you use " "

#100 Explained Report Bookmark

The following program fragment _____.

```
#include <iostream>

using namespace std;



int i = 10;

int main()

{

  int i = 20;

  {

   int i = 30;

      cout << i << ::i;

   }

      return 0;

}
```

- **A**
  prints 3010
- **B**
  prints 3020

- C
  will results in a runtime error
- D
  None of the above

Correct Answer :A

Explanation

prints 3010. :: is basically meant to manipulate a global variable, in case a local variable also has the same name.

#101 Explained Report Bookmark

For the below defined function abc, Which of the following function calls is/are illegal? (Assume h,g are declared as integers)

```
void abc(int x=0, int y=0)

{

        cout << x<< y;

}
```

- A
  abc();
- B
  abc(h);
- C
  abc(h, h)
- D
  None of the above

Correct Answer :D

Explanation

none of the above. Both the arguments are optional. All calls are legal.

#102 Explained Report Bookmark

The following C++ code results in :

```
#include "iostream"

void main(void)

{

     cout << (int i=5) << (int j=6);

}
```

- **A**
  Compilation error
- **B**
  Runtime error
- **C**
  Linktime error
- **D**
  None of the above

Correct Answer :A

Explanation

It will give compile time error

int should be declare before cout statement. In C and C++ initialization should at first.

#103 Explained Report Bookmark

If many functions have the same name, which of the following information, if present, will be used by the compiler to invoke the correct function to be used?

- **A**
  The operator `::`
- **B**
  The return value of the function
- **C**
  Function signature
- **D**
  None of the above

Correct Answer :C

Explanation

The process of having two or more functions with the same name, but different parameters (function signature), is known as function overloading. It is only through these differences that a compiler can differentiate between functions.

#104 Explained Report Bookmark

The below statement outputs _____?

```
int a = 5;

cout << "FIRST" << (a<<2) << "SECOND";
```

- **A**
  FIRST52SECOND

- B
  FIRST20SECOND
- C
  SECOND25FIRST
- D
  An Error Message

Correct Answer :B

Explanation

The left-shift operator causes the bits in shift-expression to be shifted to the left by the number of positions specified by additive-expression. The bit positions that have been vacated by the shift operation are zero-filled.

#105 Explained Report Bookmark

A constructor is called whenever _____.

- A
  an object is declared
- B
  an object is used
- C
  a class is declared
- D
  a class is used

Correct Answer :A

Explanation

A constructor is automatically called when an object is created.

#106 Explained Report Bookmark

Which of the following remarks about the differences between constructors and destructors are correct?

- **A**
  Constructors can take arguments but destructors cannot.
- **B**
  Constructors and destructors can both return a value.
- **C**
  Destructors can take arguments but constructors cannot.
- **D**
  Destructors can be overloaded but constructors cannot be overloaded.

Correct Answer :A

Explanation

Constructors can take arguments but destructors cannot. Since destructors do not take arguments, the question of overloading does not arise at all.

#107 Explained Report Bookmark

The following program fragment _____.

```
#include <iostream>

using namespace std;



int main()

{
```

```
  int x = 10;

     int &p = x;

     cout<< &p<< &x;

     return 0;
}
```

- **A**
  prints 10 and the address of x
- **B**
  results in a runtime error
- **C**
  prints the address of x twice
- **D**
  prints the address of p twice

Correct Answer :C

Explanation

prints the address of x twice. int &p=x aliases p to x. This means they refer to the same memory location. So, the address of x will be same as that of p.

#108 Explained Report Bookmark

The declaration int x; int &p=x; is same as the declaration int x, *p; p=&x;. This remark is?

- **A**
  true
- **B**
  false

- **C**
  sometimes true
- **D**
  none of above

Correct Answer :B

Explanation

&p means the address of memory location p

*p means the value at memory location p (this assumes p as a pointer)

int *p = &x; means p is a pointer and you are assigning the address of memory location where the value of x is stored

#109 Explained Report Bookmark

The following program segment _____.

```
const int m=10;

int &n=m;

n=11;

cout << m << n;
```

- **A**
  results in compile time error
- **B**
  results in run time error
- **C**
  prints 1111
- **D**
  prints 1011

Correct Answer :A

Explanation

Const Keyword in C++ 'const' keyword stands for constant. In C++ it is used to make some values constant throughout the program. If we make an artifact of a C++ program as constant then its value cannot be changed during the program execution.

Compile time error will be like this:

[Error] invalid initialization of reference of type 'int&' from expression of type 'const int'.

#110 Explained Report Bookmark

The following program segment _____.

```
int a = 10;

int const &b = a;

a = 11;

cout << a << b;
```

- A
  results in compile time error
- B
  results in run time error
- C
  prints 1111
- D
  none of the above

Correct Answer :C

Explanation

In Given Program

b is holding a reference of a so when we update the value of a then it automatically applied to b

It will simply print 1111 as there will be no compiler error.

#111 Explained Report Bookmark

Consider the following program segment. A complete C++ program with these two statements will _____.

```
static char X[3] = "1234";

cout << X;
```

- A
  print 1234
- B
  print 123
- C
  print 1234 followed by some junk
- D
  will give a compilation error

Correct Answer :D

Explanation

will give a compilation error. C++ forbids initialization with strings, whose length is more than the size of the array. A C compiler permits.

#112 Explained Report Bookmark

For the below function abc, Which of the following function calls is/are illegal? (Assume x, z are declared as integers)

```
void abc(int x=0,  int z=0)

{

        cout << x<< z;

}
```

- A
  abc();
- B
  abc(h);
- C
  abc(h,h);

- **D**
  None of the above

Correct Answer :D

Explanation

A default argument is a value provided in a function declaration that is automatically assigned by the compiler if the caller of the function doesn't provide a value for the argument with a default value. ... We don't have to write 3 sum functions, only one function works by using default values for 3rd and 4th arguments

#113 Explained Report Bookmark

The compiler identifies a virtual function to be pure by _____.

- **A**
  the presence of the keyword pure
- **B**
  its location in the program
- **C**
  the function being equated to 0
- **D**
  none of the above

Correct Answer :C

Explanation

Virtual member functions are resolved dynamically.

#114 Explained Report Bookmark

For a method to be an Interface between the outside world and a class, it has to be declared _____.

- **A**
  private
- **B**
  protected
- **C**
  public
- **D**
  external

Correct Answer :C

Explanation

All the class members declared under public will be available to everyone. The data members and member functions declared public can be accessed by other classes too. The public members of a class can be accessed from anywhere in the program using the direct member access operator (.)

#115 Explained Report Bookmark

A function that does the same operation on different data types is to be implemented using _____.

- **A**
  macros
- **B**
  overloading
- **C**
  function templates
- **D**
  default arguments

Correct Answer :C

Explanation

Function templates. Function templates are special functions that can operate with generic types. This allows us to create a function template whose functionality can be adapted to more than one type or class without repeating the entire code for each type. In C++ this can be achieved using template parameters.

#116 Explained Report Bookmark

Which of the following keyword supports dynamic method resolution?

null

- **A**
  abstract
- **B**
  virtual
- **C**
  dynamic
- **D**
  typeid

Correct Answer :B

Explanation

virtual. The virtual keyword indicates that the virtual method will be resolved at runtime(i.e., the method resolution is dynamic

#117 Explained Report Bookmark

Which of the following member functions is resolved dynamically?

- A
  static member function
- B
  const member function
- C
  virtual member function
- D
  non virtual member function

Correct Answer :C

Explanation

A virtual function is a member function that you expect to be redefined in derived classes.

When you refer to a derived class object using a pointer or a reference to the base class, you can call a virtual function for that object and execute the derived class's version of the function dynamically.

#118 Explained Report Bookmark

What is an exception specification?

null

- A
  Declaration of the list of exceptions a function can throw using the throws clause.
- B
  Requirements specification of how to handle exceptions in a program.
- C
  esign specification of how to handle exception in a program.

- **D**
  Specification document on exception handling implementation.

Correct Answer :A

Explanation

Declaration of the list of exceptions a function can throw using the throws clause.

#119 Explained Report Bookmark

Which of the following cannot be declared as template?

null

- **A**
  Global functions
- **B**
  Classes
- **C**
  Member functions
- **D**
  Macros

Correct Answer :D

Explanation

Macros are implemented in a preprocessor and cannot be implemented as a template. Functions and classes can be declared as templates.

#120 Explained Report Bookmark

When is std::bad_alloc exception thrown?

null

- **A**
  When new operator cannot memory.
- **B**
  When alloc function fails
- **C**
  When type requested for new operation is considered bad, this exception is thrown
- **D**
  When delete operator cannot delete the allocated(corrupted) object.

Correct Answer :A

Explanation

When new operator cannot memory.

#121 Explained Report Bookmark

Which of the following member is not automatically provided by the compiler if the programmer does not provide it explicitly?

- **A**
  Constructor
- **B**
  Destructor
- **C**
  Equality operator ==
- **D**
  Assignment operator =

Correct Answer :C

Explanation

The compiler should provide (as and when needed)

1. a constructor
2. a destructor
3. a copy constructor
4. = operator
5. the reference operator(&) - the address

#122 Explained Report Bookmark

What is the default inheritance type when no access specifier is explicitly specified for the base class?

- **A**
  internal
- **B**
  public
- **C**
  private
- **D**
  protected

Correct Answer :C

Explanation

If you do not choose an inheritance, C++ defaults to private inheritance in the same way class members default to private access for classes. The default type of the inheritance is private in C++

#123 Explained Report Bookmark

Which of the following casting operators use RTTI(Runtime Type Identification)?

null

- **A** const_cast
- **B** static_cast
- **C** dynamic_cast
- **D** reinterpret_cast

Correct Answer :C

Explanation

The dynamic_cast operator uses the runtime information about the type of the object for performing the cast.

#124 Explained Report Bookmark

Which of the following is the most general exception handler that catches exception of 'any type'?

- **A** catch(std::exception)
- **B** catch(std::any_exception)
- **C** catch(...)
- **D** catch()

Correct Answer :C

Explanation

```
try {

   // ...

} catch (...) {

   // ...

}
```

Note that the `...` inside the `catch` is a real ellipsis, ie. three dots.

However, because C++ exceptions are not necessarily subclasses of a base `Exception` class, there isn't any way to actually see the exception variable that is thrown when using this construct

#125 Explained Report Bookmark

Which of the following operators can be implemented as a non-member operator?

- **A**
  `=`(assignment operator)
- **B**
  `( )`(function call operator
- **C**
  `[ ]`(array access operator)
- **D**
  `+`(addition operator)

Correct Answer :D

Explanation

All the other three operators – function call, array access, and assignment operators must be implemented as a member operator. The + operator can be implemented as a nonmember operator.

#126 Explained Report Bookmark

Which of the STL containers store the elements contiguously (in adjacent memory locations)?

null

- **A**
  std::vector
- **B**
  std::list
- **C**
  std::set
- **D**
  std::map

Correct Answer :A

Explanation

std::vector. The vector is a dynamic array that can grow(or shrink) as needed. It stores the elements contiguously.

#127 Explained Report Bookmark

Which of the following members occupy space in xyz object?

```
class XYZ {
```

```
  int mem1;

     static int mem2;

     static void foo() { }

     void bar() { }

} xyz;
```

- **A** int mem1;
- **B** static int mem2;
- **C** static void foo(){ }
- **D** void bar(){ }

Correct Answer :A

Explanation

int mem1;. Nonstatic data members occupy space in objects. Static memebers or member functions do not occupy any space in an object.

#128 Explained Report Bookmark

Which of the following operators is used to obtain the dynamic type of an object/class?

dynamic_cast

- **A** dynamic_cast

- B
  typeid
- C
  typeof
- D
  std::type_info

Correct Answer :B

Explanation

The typeid operator allows the type of an object to be determined at run time. The result of typeid is a const type_info& . The value is a reference to a type_info object that represents either the type-id or the type of the expression, depending on which form of typeid is used

#129 Explained Report Bookmark

std::type_info

- A
  const_cast
- B
  static_cast
- C
  dynamic_cast
- D
  reinterpret_cast

Correct Answer :D

Explanation

reinterpret_cast is a type of casting operator used in C++.

- It is used to convert one pointer of another pointer of any type, no matter either the class is related to each other or not.
- It does not check if the pointer type and data pointed by the pointer is same or not.

#130 Explained Report Bookmark

Which of the following cannot be used with the keyword virtual?

- A
  Class
- B
  memeber function
- C
  constructor
- D
  destructor

Correct Answer :C

Explanation

Virtual keyword cannot be used with constructors as constructors are defined to initialized an object of particular class hence no other class needs constructor of other class

#131 Explained Report Bookmark

Which of the following is the correct way of declaring a function as constant?

null

- A
  const int ShowData(void) { /* statements */ }
- B
  int const ShowData(void) { /* statements */ }

- **C**
  int ShowData(void) const { /* statements */ }
- **D**
  Both A and B

Correct Answer :C

Explanation

int ShowData(void) const { /* statements */ }

#132 Explained Report Bookmark

What happens if the base and derived class contains definition of a function with same prototype?

null

- **A**
  Compiler reports an error on compilation.
- **B**
  Only base class function will get called irrespective of object.
- **C**
  Only derived class function will get called irrespective of object.
- **D**
  Base class object will call base class function and derived class object will call derived class function.

Correct Answer :D

Explanation

Base class object will call base class function and derived class object will call derived class function.

#133 Explained Report Bookmark

Pick up the valid declaration for overloading ++ in postfix, where T is the class name?

null

- **A**
  T operator++();
- **B**
  T operator++(int);
- **C**
  operator++();
- **D**
  T& operator++(int);

Correct Answer :B

Explanation

T operator++(int);. The parameter int is just to signify that it is the postfix form overloaded. Shouldn't return reference as per its original behavior.

#134 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream>

using namespace std;



int main()
```

```
{

  char *s = "Fine";

     *s = 'N';

     cout << (s) << endl;

     return 0;

}
```

- **A**
  Fine
- **B**
  Nine
- **C**
  Compile error
- **D**
  Runtime error

Correct Answer :D

Explanation

Runtime error. *s='N', trying to change the character at base address to 'N' of a constant string leads to runtime error.

#135 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream>
```

```
using namespace std;



class Base

{

  public:

     void f()

  {

       cout << "Base\n";

     }

};



class Derived : public Base

{

  public:

     void f()

  {

       cout<<"Derived\n";
```

```
    }
};


int main()
{
  Derived obj;
  obj.Base::f();
  return 0;
}
```

- **A**
  Base
- **B**
  Derived
- **C**
  Compile error
- **D**
  None of the above.

Correct Answer :A

Explanation

Here obj is an object of derived class. By derived class objects i.e. obj we are accessing the base class function (f()) using scope resolution operator which will print Base

#136 Explained Report Bookmark

What does the following statement mean?

```
int (*fp)(char*)
```

- **A**
  pointer to a pointer
- **B**
  pointer to an array of chars
- **C**
  pointer to function taking a char* argument and returns an int
- **D**
  function taking a char* argument and returning a pointer to int

Correct Answer :C

Explanation

`fp` is a pointer to function, which is taking a `char *` parameter and returning `int`.

#137 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream>

using namespace std;



int main()
```

```
{

  int a[] = {10, 20, 30};

     cout << *a+1;

}
```

- **A**
  10
- **B**
  11
- **C**
  20
- **D**
  21

Correct Answer :B

Explanation

*a refers to 10 and adding a 1 to it gives 11.

#138 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream>

using namespace std;



int main()
```

```
{

  int i = 1, j = 2, k = 3, r;

     r = (i, j, k);

     cout << r << endl;

     return 0;

}
```

- **A**
  1
- **B**
  2
- **C**
  3
- **D**
  compile error

Correct Answer :C

Explanation

Comma is called as the separator operator and the associativity is from left to right. Therefore 'k' is the expressions resultant.

#139 Explained Report Bookmark

The following operator can be used to calculate the value of one number raised to another.

- **A**
  **

- B
  ^
- C
  ^^
- D
  None of the above

Correct Answer :D

Explanation

None of the above. There is no such operator in C/C++.

#140 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream>

using namespace std;

#define MIN(a,b) (((a)<(b)) ? a : b)



int main()

{

     float i, j;

     i = 100.1;
```

```
    j = 100.01;

    cout <<"The minimum is"<< MIN(i, j)<< endl;

    return 0;

}
```

- **A**
  100.1
- **B**
  100.01
- **C**
  100.1
- **D**
  compile time error

Correct Answer :B

Explanation

100.01. In this program, we are getting the minimum number using conditional operator.

#141 Explained Report Bookmark

What is the output of the following C++ program?

```
void main()

{



}
```

- A
  No output
- B
  Garbage
- C
  Compile error
- D
  Runtime error

Correct Answer :A

Explanation

No output. It is valid to have main() function empty, therefore producing no displayable output

#142 Explained Report Bookmark

What does derived class does not inherit from the base class?

- A
  constructor and destructor
- B
  friends
- C
  operator = () members
- D
  all of the mentioned

Correct Answer :D

Explanation

all of the mentioned. The derived class inherit everything from the base class except the given things.

#143 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream>

using namespace std;



int main()

{

     cout<< "Value of __LINE__ : " << __LINE__ << endl;

     cout<< "Value of __FILE__ : " << __FILE__ << endl;

     cout<< "Value of __DATE__ : " << __DATE__ << endl;

     cout<< "Value of __TIME__ : " << __TIME__ << endl;

     return 0;

}
```

- A
  5
- B
  Details about your file
- C
  compile time error

- D
  none of the mentioned

Correct Answer :B

Explanation

Details about your file. In this program, we are using the macros to print the information about the file.

#144 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream.h>

using namespace std;

#define SquareOf(x) x * x



int main()

{

     int x;

     cout<< SquareOf(x + 4);

     return 0;

}
```

- **A**
  16
- **B**
  64
- **C**
  compile time error
- **D**
  None of the mentioned above

Correct Answer :D

Explanation

none of the mentioned. In this program, as we haven’t initiailzed the variable x, we will get a output of ending digit of 4.

#145 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream.h>

using namespace std;

#define PR(id)  cout << id;



int main()

{

     int i = 10;
```

```
    PR(i);

    return 0;

}
```

- **A**
  10
- **B**
  15
- **C**
  20
- **D**
  None of the mentioned above

Correct Answer :A

Explanation

In this program, we are just printing the declared values.

#146 Explained Report Bookmark

What is the output of the following C++ program?

```
#include <iostream.h>

using namespace std;

#define MAX 10



int main()
```

```
{
    int num;
    num = ++MAX;
    cout << num;
    return 0;
}
```

- **A**
  11
- **B**
  10
- **C**
  compile time error
- **D**
  none of the mentioned

Correct Answer :C

Explanation

```
error: lvalue required as increment operand
```

```
 #define MAX 10
```

#147 Explained Report Bookmark

what is the output of the following C++ program?

```
#include <iostream>
```

```
using namespace std;



int main()

{

  int *p = new int;

  delete p;

  delete p;

  cout<<"Done";

  return 0;

}
```

- **A**
  Done
- **B**
  Compile error
- **C**
  Runtime error
- **D**
  None of the above.

Correct Answer :C

Explanation

**** Error in `./a.out': double free or corruption (fasttop): 0x0000000000a45c20 ***

#148 Explained Report Bookmark

what is the output of the following C++ program?

```
#include <iostream>

using namespace std;



void f()

{

  static int i = 3;

     cout << (i);

     if(--i) f();

}



int main()

{

  f();

  return 0;
```

```
}
```

- **A**
  3210
- **B**
  321
- **C**
  333
- **D**
  Compile error

Correct Answer :B

Explanation

In this program, recursive function calls are made. When if statement will contain 0 then this will be a termination condition for recursive function calls.

#149 Not Explained Report Bookmark

What is the output of the following C++ program?

```
#include<iostream>

using namespace std;



int main()

{

     int array[] = {10, 20, 30};
```

```
    cout << -2[array];

    return 0;

}
```

- **A**
  -15
- **B**
  -30
- **C**
  compile time error
- **D**
  Garbage value

Correct Answer :B

No Explanation Available

#150 Explained Report Bookmark

What is the output of the following C++ program?

```
#include<iostream>

using namespace std;



int main()

{

    cout<< strcmp("strcmp()","strcmp()");
```

```
  return 0;

}
```

- **A**
  0
- **B**
  1
- **C**
  -1
- **D**
  Invalid use of `strcmp()` function

Correct Answer :A

Explanation

0,strcmp return 0 if both the strings are equal

#151 Explained Report Bookmark

What is the output of the following C++ program?

```
#include<iostream>

#include <string.h>

using namespace std;

template <typename T>



void print_mydata(T output)
```

```
{
     cout << output << endl;
}


int main()
{
     double d = 5.5;
     string s("Hello World");
     print_mydata( d );
     print_mydata( s );
     return 0;
}
```

- **A**
  5.5 Hello World
- **B**
  5.5
- **C**
  Hello World
- **D**
  None of the mentioned

Correct Answer :A

Explanation

Here the template 'typename' is able to capture multiple datatype & function prints the value

#152 Explained Report Bookmark

What is the output of the following C++ program?

```
#include<iostream>

using namespace std;



int main()

{

     double a = 21.09399;

     float b = 10.20;

     int c ,d;

     c = (int) a;

     d = (int) b;

     cout << c <<'\t'<< d;

     return 0;
```

```
}
```

- **A**
  20 10
- **B**
  10 21
- **C**
  21 10
- **D**
  None of the mentioned

Correct Answer :C

Explanation

Here manual typecasting is done.

Narrowing Casting (manually) - converting a larger type to a smaller size type

double -> float -> long -> int -> char -> short -> byte

#153 Explained Report Bookmark

What is the output of the following C++ program?

```
#include<iostream>

using namespace std;



class abc
```

```
{

  public:

     int i;

     abc(int i)

    {

      i = i;

      }

};



int main()

{

     abc m(5);

     cout << m.i;

    return 0;

}
```

- A
  5
- B
  Garbage

- **C**
  Error at the statement i=i;
- **D**
  Compile error: 'i' declared twice.

Correct Answer :B

Explanation

Garbage value will be printed on the screen as m will access data member i of class abc and this variable is not initialized.

#154 Explained Report Bookmark

What is the output of the following C++ program?

```
#include<iostream>

using namespace std;



int main()

{

     union abc

  {

       int x;

     char ch;
```

```
}

  var;

  var.ch = 'A';

  cout << var.x;

  return 0;

}
```

- **A**
  A
- **B**
  Garbage value
- **C**
  65
- **D**
  97

Correct Answer :C

Explanation

as the union variables share common memory for all its elements, x gets ‘A’ whose ASCII value is 65 and is printed.

#155 Explained Report Bookmark

How many bits of memory needed for internal representation of a class?

- **A**
  1

- **B**
  2
- **C**
  4
- **D**
  No memory needed

Correct Answer :D

Explanation

no memory needed. Classes that contain only type members, nonvirtual function members, and static data members do not require memory at run time.

#156 Explained Report Bookmark

What is the output of the following C++ program?

```
#include<iostream>

using namespace std;



int main()

{

     class student

     {
```

```
        int rno = 10;

    } v;

    cout << v.rno;

   return 0;

}
```

- **A**
  10
- **B**
  Garbage
- **C**
  Runtime error
- **D**
  Compile error

Correct Answer :D

Explanation

Compile error. Class member variables cannot be initialized.

#157 Explained Report Bookmark

What is the output of the following C++ program?

```
#include<iostream>

using namespace std;
```

```
int main()

{

     int i = 13, j = 60;

     i ^= j;

     j ^= i;

     i ^= j;

     cout << (i) <<" "<< (j)

    return 0;

}
```

- **A**
  73 73
- **B**
  .60 13
- **C**
  13 60
- **D**
  60 60

Correct Answer :B

Explanation

The ^ (bitwise *XOR*) in C or *C++* takes two numbers as operands and does *XOR* on every bit of two numbers. The result of *XOR* is 1 if the two bits are different.

#158 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h> #include<string.h>

void main()

{

    cout<<strlen(“Hello, World.\n”)<<”\n”;

}
```

- **A**
  14
- **B**
  13
- **C**
  12
- **D**
  none

Correct Answer :A

Explanation

```
cout<<strlen("\n")<<"\n"; //returns 1

cout<<strlen("Hello,")<<"\n"; //returns 6

cout<<strlen("Hello, ")<<"\n";//returns 7
```

```
cout<<strlen("Hello, World\n")<<"\n";//returns 13

cout<<strlen("Hello, World.\n")<<"\n";//returns 14
```

H-1 e-2 l-3 l-4 0-5

,-6

space - 7

, w-8 ,o-9 r-10 l-11 d-12

.(dot)-13 , \n-14

total length -14 characters

#159 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h> void main()

{

   /* this is /* an example */ of nested comment */

   cout<<123<<endl;

}
```

- **A**
  123
- **B**
  Compile time error
- **C**
  None
- **D**
  Run time error

Correct Answer :B

Explanation

nested comments are not allowed using /*

#160 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h>

void main()

{

   int a = 20; int &n = a; n=a++; a=n++;

    cout<<a <<",”<<n<<endl;

}
```

- **A**
  20, 20
- **B**
  20, 21
- **C**
  21, 22
- **D**
  None

Correct Answer :D

Explanation

Correct Answer : 22,22

#161 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h>

void main()

{



   int a = 20,b=100;

   int &n = a; n=a++;

  n = &b;
```

```
  cout<<a <<”,”<<n<<endl;

}
```

- **A**
  21, 21
- **B**
  20, 21
- **C**
  21, 22
- **D**
  Error

Correct Answer :D

Explanation

reference variable can not be changed

#162 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h>

void main()

{

   bool a=10; cout<<a<<endl;
```

```
}
```

- **A**
  10
- **B**
  false
- **C**
  1
- **D**
  error

Correct Answer :C

Explanation

a contains a value greater than 0 this means true and when a is printed on the screen then it will show true i.e. 1.

#163 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h>

void main()

{

   int main; main = 100;

   cout<<main++<<endl;
```

```
}
```

- **A**
  101
- **B**
  100
- **C**
  None
- **D**
  Error: one cannot use main as identifier

Correct Answer :B

Explanation

100 will be printed first and then ++ will be applied on main because it is the post increment operator.

#164 Explained Report Bookmark

What is the output of the following code?(Turbo C++)

```
#include<iostream.h>

void main()

{

    int a=0,x;

    x = ++a * --a;
```

```
    cout<<++a<< " " << a++ << " " << x <<endl;

}
```

- **A**
  0, 0, 0
- **B**
  1, 1, 0
- **C**
  2, 2, 2
- **D**
  3, 2, 2

Correct Answer :B

Explanation

x = ++a * --a; here ++a evaluated to 1 then --a evaluated 0

x=1*0 => 0 will be output

cout<<++a<< " " << a++ << " " << x <<endl;

after that ++a will be 1 but a++ will be 1 as well because its post operation

final output will be 1 1 0

first all pre increments are solved then perform operation

#165 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h>

void main()

{

   a=32;

   cout<<a<<endl;

   int a;

}
```

- **A**
  32
- **B**
  0
- **C**
  Compile time error
- **D**
  Run time error

Correct Answer :C

Explanation

Error will be like this:

[Error] 'a' was not declared in this scope.

here we assigning before declaring the variable

#166 Explained Report Bookmark

Evaluate the m%n++ expression, assuming m=24 and n=7

- **A**
  4
- **B**
  3
- **C**
  2
- **D**
  None

Correct Answer :B

Explanation

First of all % will be evaluated because ++ is a post increment operator. After whole expression evaluation, ++ will be applied on n.

#167 Explained Report Bookmark

Evaluate the m%++n expression, assuming m=24 and n=7

- **A**
  4
- **B**
  3

- **C**
  2
- **D**
  none

Correct Answer :D

Explanation

The correct answer will be zero. This happens because ++ is a pre increment operator and it will be evaluated first before % operator. n will become 8 and when 24 % 8 is done then answer must be 0.

#168 Explained Report Bookmark

Elements in an array are identified by a unique .

- **A**
  symbol
- **B**
  order
- **C**
  subscript
- **D**
  data type

Correct Answer :C

Explanation

The individual elements of an array are referenced by appending a subscript, in square brackets, behind the name. The subscript itself can be any legitimate C expression that yields an integer value, even a general

expression. ... Usually, the array size is fixed, while strings can have a variable number of elements

#169 Explained Report Bookmark

An address is a ____, while a pointer is a ____

- **A**
  variable, location
- **B**
  variable, position
- **C**
  constant, variable
- **D**
  None

Correct Answer :A

Explanation

An address indicates a variable and a pointer indicates a location.

#170 Explained Report Bookmark

To execute a C++ program, one first need to translate the source code into object code. This process if called _____

- **A**
  translating
- **B**
  sourcing
- **C**
  compiling
- **D**
  coding

Correct Answer :C

Explanation

To execute the program, however, the programmer must translate it into machine language, the language that the computer understands. The first step of this translation process is usually performed by a utility called a compiler. The compiler translates the source code into a form called object code

#171 Explained Report Bookmark

What is wrong with the following program?

```
#include<iostream.h>

void main()

{

   int a[5] = {0};

   for(int i=0;i<2;i++)

      a[i]=i;

   for(int i=0;i<5;i++)

      cout<<a[i]<<endl;

}
```

- A
  There is a run time error
- B
  Array ‘a’ is not initialized properly
- C
  There is no problem
- D
  Redeclaration of variable ‘i’

Correct Answer :C

Explanation

The above program works correctly without any error.

#172 Explained Report Bookmark

To delete a dynamically allocated array named ‘a’, the correct statement is

- A
  delete a[0];
- B
  delete []a;
- C
  delete [0]a;
- D
  delete a;

Correct Answer :B

Explanation

To delete a dynamically allocated array named ‘a’, the correct statement is

delete []a;

#173 Explained Report Bookmark

When do preprocessor directives execute?

- **A**
  Before the compiler compiles the
- **B**
  After the compiler compiles the
- **C**
  At the same time as the compiler compiles the program
- **D**
  None

Correct Answer :A

Explanation

Preprocessor directives are lines included in a program that begin with the character #, which make them different from a typical source code text. They are invoked by the compiler to process some programs before compilation.

#174 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h>

void main()
```

```
{

   int a; bool b;

   a = 12 > 100;

   b = 12 >= 100;

    cout<<a<<" "<<b<<endl;

}
```

- **A**
  Error
- **B**
  0 false
- **C**
  0 1
- **D**
  0 0

Correct Answer :D

Explanation

> and >= operators will be evaluated before = operator because relational operators have higher precedence than assignment operators.

Operator	Example
postfix	something++, something--
unary	++something, --something, -something, ! something
multiplicative	* / %
additive	+ -
relational	< > <= >=
equality	== !=
logical AND	&&
logical OR	||
assignment	= += -= *= /= %= (and other compound assignment operators)

#175 Explained Report Bookmark

What is the output of the following code?

```
#include<iostream.h>

int a = 1; void main()

{

   int a = 100;

   {

      int a = 200;

     {

         int a = 300; cout<<a<<",";

      }
```

```
        cout<<a<<",";

    }

   cout<<a<<",";

    cout<<::a<<",";

}
```

- **A**
  Error
- **B**
  100, 200, 300, 100
- **C**
  300, 200, 100, garbage,
- **D**
  300, 200, 100, 1,

Correct Answer :D

Explanation

File Scope

Identifiers declared outside all blocks and functions have file scope. It can be used in all the blocks and functions written inside the file in which the variable declaration appears. These identifiers are known from their point of declaration until the end of file.

Block scope

Blocks are portions of C++ code contained within curly braces({....}). Identifiers declared within a block have block scope and are visible from their points of definition to the end of the innermost containing block. A duplicate identifier name in a block hides the value of an identifier with the same name defined outside the block.

A variable name declared in a block is local to that block. It can be used only in it and the other blocks contained under it.

#176 Explained Report Bookmark

Binding of data and code together is called?

null

- **A**
  Abstraction
- **B**
  Encapsulation
- **C**
  Modularity
- **D**
  Data security

Correct Answer :B

Explanation

Binding of data and functions in one single unit named class is know as encapsulation. Encapsulation provides data hiding facility. In programming languages encapsulation is implemented using classes. ... To access them outside class one must use object(instance) of that class or class name.

#177 Explained Report Bookmark

Runtime polymorphism can be achieved using

null

- **A**
  Operator overloading
- **B**
  Function overloading
- **C**
  Function overriding
- **D**
  All of the above

Correct Answer :C

Explanation

Runtime polymorphism: This type of polymorphism is achieved by Function Overriding. Function overriding on the other hand occurs when a derived class has a definition for one of the member functions of the base class. That base function is said to be overridden

#178 Explained Report Bookmark

C++ is _____ kind of language

null

- **A**
  Procedure oriented programming language
- **B**
  Object based programming language
- **C**
  Object oriented programming language
- **D**
  Pure Object oriented programming language

Correct Answer :C

Explanation

C++ is a general-purpose object-oriented programming (OOP) language, developed by Bjarne Stroustrup, and is an extension of the C language. ... C++ is considered to be an intermediate-level language, as it encapsulates both high- and low-level language features

#179 Explained Report Bookmark

Which one of the following data type is newly added in C++?

null

- **A**
  char
- **B**
  wchar_t
- **C**
  void
- **D**
  int

Correct Answer :B

Explanation

wchar_t is a wide character. It is used to represent characters which require more memory to represent them than a regular char . It is, for example, widely used in the Windows API. However, the size of a wchar_t is implementation-dependant and not guaranteed to be larger than char

#180 Explained Report Bookmark

_____ is standard namespace in c++

null

- **A**
  std namespace
- **B**
  lang namespace
- **C**
  system namespace
- **D**
  Global namespace

Correct Answer :A

Explanation

All the files in the C++ standard library declare all of its entities within the std namespace. That is why we have generally included the using namespace std; statement in all programs that used any entity defined in iostream

#181 Explained Report Bookmark

To access members of namespace, we should use _____ operator

null

- **A**
  .
- **B**
  ->
- **C**
  :

- **D**
  ::

Correct Answer :D

Explanation

Identifiers outside the namespace can access the members by using the fully qualified name for each identifier, for example std::vector<std::string> vec; , or else by a using Declaration for a single identifier (using std::string), or a using Directive for all the identifiers in the namespace (using namespace std;)

#182 Explained Report Bookmark

cout is object of _____ class

null

- **A**
  istream
- **B**
  ifstream
- **C**
  ostream
- **D**
  ofstream

Correct Answer :C

Explanation

std::cout is an object of class ostream that represents the standard output stream oriented to narrow characters (of type char). It corresponds to the C stream stdout

#183 Explained Report Bookmark

Which one of the following header file is required to use manipulators in c++

null

- **A**
  iostream
- **B**
  iomanip
- **C**
  manip
- **D**
  new

Correct Answer :B

Explanation

These functions take parameters when used as manipulators. They require the explicit inclusion of the header file <iomanip>

#184 Explained Report Bookmark

Members of structure and class are by default _____ and _____

null

- **A**
  private, public
- **B**
  private, private

- C
  public, private
- D
  public, public

Correct Answer :C

Explanation

All the member variables and member functions in a structure are 'public' by default and you cannot change the 'access specifier'. Whereas in a class you can define access specifier(private, public or protected) to each and every member and all the members are 'private' by default.

#185 Explained Report Bookmark

Which one of the following member function do not get this pointer

- A
  Virtual member function
- B
  Constant member function
- C
  Static member function
- D
  Inline function

Correct Answer :C

Explanation

This *pointer is* a *pointer* accessible only within the nonstatic *member functions* of a class, struct, or union type. It points to the object for which

the *member function is* called. Static *member functions don't have* "this" *pointer*

also This *pointer is* an implicit parameter to all *member functions*. Therefore, inside a *member function*, this *may be* used to refer to the invoking object. "this" *pointer is* a *pointer* accessible only within the nonstatic *member functions* of a class, struct, or union type. It points to the object for which the *member function is* called. Static *member functions don't have* "this" *pointer*, because friends *are not members* of a class

#186 Explained Report Bookmark

Class is _____ for object

null

- **A**
  Template
- **B**
  Blueprint
- **C**
  Model
- **D**
  all of the above

Correct Answer :B

Explanation

Classes and objects from the essential part of Object-oriented programming, where a class can be considered as a construct that encapsulates a group of variables and methods; whereas, an object acts as member or instance of that class. A class is a blueprint from which you can create the instance, i.e., objects.

#187 Explained Report Bookmark

Which one of the following keyword is allowed to use with constructor

null

- **A**
  virtual
- **B**
  inline
- **C**
  constant
- **D**
  Static

Correct Answer :B

Explanation

- Constructors may be declared as inline, explicit, friend or constexpr.
- A constructor can initialize an object that has been declared as const, volatile or const volatile. The object becomes const after the constructor completes.
- To define a constructor in an implementation file, give it a qualified name as with any other member function: `Box::Box(){...}`.

#188 Explained Report Bookmark

_____ Keyword is used to modify state of non-constant data member inside const member function?

null

- **A**
  mutable

- B
  extern
- C
  volatile
- D
  static

Correct Answer :A

Explanation

The keyword mutable is mainly used to allow a particular data member of const object to be modified. When we declare a function as const, the this pointer passed to function becomes const. Adding mutable to a variable allows a const pointer to change members.

#189 Explained Report Bookmark

If new operator fails to allocate memory then __

- A
  Returns NULL
- B
  Throws bad_fail exception
- C

  Throws bad_alloc exception
- D
  Throws bad_cast exception

Correct Answer :C

Explanation

std::bad_alloc is a type of exception that occurs when the new operator fails to allocate the requested space. This type of exception is thrown by the standard definitions of operator new (declaring a variable) and operator new[] (declaring an array) when they fail to allocate the requested storage space.

#190 Explained Report Bookmark

If we use _____ with new operator then it returns NULL

null

- **A**
  null ptr
- **B**
  throw
- **C**
  nothrow
- **D**
  new do not return NULL. It throws exception

Correct Answer :C

Explanation

std::nothrow

By default, when the new operator is used to attempt to allocate memory and the handling function is unable to do so, a bad_alloc exception is thrown. But when nothrow is used as argument for new , it returns a null pointer instead.

#191 Explained Report Bookmark

To deallocate memory in C++, we should use

- A
  new() function
- B
  malloc() fucntion
- C
  Delete operator
- D
  Garbage collector

Correct Answer :C

Explanation

Use the malloc() or new() function to allocate memory in designated blocks.

To reallocate memory, the realloc() function is used.

When finished, always include a free() function call in order to free up the memory! If you used new(), use delete() to free up the memory.

#192 Explained Report Bookmark

Using the non-member function, we cannot overload the following operators

- A
  Arrow operator (->)
- B
  Index/Subscript operator ([])
- C
  Function Call operator [( )]
- D
  Assignment operator (=)

Correct Answer :D

Explanation

`operator=` is a special member function that the compiler will provide if you don't declare it yourself. Because of this special status of `operator=` it makes sense ro require it to be a member function,

so there is no possibility of there being both a compiler-generated member `operator=` and a user-declared friend `operator=` and no possibility of choosing between the two.

Short answer:A few operators *have to be member functions*. The assignment operator is one of the,

#193 Explained Report Bookmark

In C++, Default mode of inheritance is

null

- **A**
  private
- **B**
  protected
- **C**
  public
- **D**
  none of the above

Correct Answer :A

Explanation

The default type of the inheritance is private in C++. the default access specifier is an important differentiator between classes and structs. It is public by default for structs and private by default for classes

#194 Explained Report Bookmark

Diamond problem is created by _____ inheritance?

null

- **A**
  Multiple
- **B**
  Hierarchical
- **C**
  Hybrid
- **D**
  Virtual

Correct Answer :A

Explanation

The diamond problem arises when multiple inheritance is used. This problem arises because the same name member functions get derived into a single class. ... At least 2 base classes and one class to inherit those two classes. If lesser, it becomes single level inheritance.

#195 Explained Report Bookmark

To avoid diamond problem, we should use _____ keyword?

null

- **A**
  Static

- B
  Virtual
- C
  Constant
- D
  mutable

Correct Answer :B

Explanation

Click Here For More detail

#196 Explained Report Bookmark

Which one of the following is correct syntax of pure virtual function.

null

- A
  void print( ) =0;
- B
  void print( ) =0{ };
- C
  virtual void print( ) =0{ };
- D
  virtual void print( ) =0;

Correct Answer :D

Explanation

A virtual function will become pure virtual function when you append "=0" at the end of declaration of virtual function. Pure virtual function doesn't have body or implementation. We must implement all pure virtual functions in derived class.

Pure virtual function is also known as abstract function.

A class with at least one pure virtual function or abstract function is called abstract class. We can't create an object of abstract class. Member functions of abstract class will be invoked by derived class object.

virtual return_type function_name(parameters) = 0; where {=0} is called pure specifier

#197 Explained Report Bookmark

RTTI stands for _____

null

- **A**
  Runtime Type Information
- **B**
  Runtime Type Identification
- **C**
  Both a and b
- **D**
  Runtime template information

Correct Answer :C

Explanation

In computer programming, run-time type information or run-time type identification (RTTI) is a feature of the C++ programming language that exposes information about an object's data type at runtime. Run-time type information can apply to simple data types, such as integers and characters, or to generic types.

#198 Explained Report Bookmark

Which of the following best defines a class?

null

- **A**
  Blueprint of an object
- **B**
  Parent of an object
- **C**
  Instance of an object
- **D**
  Scope of an object

Correct Answer :A

Explanation

A class is a blueprint or prototype that defines the variables and methods common to all objects of a certain kind.

#199 Explained Report Bookmark

A derived class can access all the only _____members of its base class

null

- **A**
  non-public
- **B**
  private
- **C**
  non-private
- **D**
  static

Correct Answer :C

Explanation

A derived class can access all the non-private members of its base class. Thus base-class members that should not be accessible to the member functions of derived classes should be declared private in the base class.

#200 Explained Report Bookmark

feature of OOP indicates code reusability?

null

- A
  Encapsulation
- B
  Abstraction
- C
  Inheritance
- D
  Polymorphism

Correct Answer :C

Explanation

Inheritance indicates the code reusability. Encapsulation and abstraction are meant to hide/group data into one element. Polymorphism is to indicate different tasks performed by a single entity.

#201 Explained Report Bookmark

Object-based programming language follows all the features of OOPs except _____

null

- A
  composition
- B
  Inheritance
- C
  Association
- D
  Encapsulation

Correct Answer :B

Explanation

Object-based programming language follows all the features of OOPs except Inheritance

#202 Explained Report Bookmark

The ability to create a new class from an existing class is called _____.

null

- **A**
  Encapsulation
- **B**
  Inheritance
- **C**
  Abstraction
- **D**
  Polymorphism

Correct Answer :B

Explanation

Inheritance is the ability of any class to extract and use features of other classes. It is the process by which new classes called the derived classes are created from existing classes called Base classes. ... The most basic concept of inheritance is Creating or Deriving a new class using another class as a base

#203 Explained Report Bookmark

A destructor takes _____ arguments

null

- **A**
  0
- **B**
  1
- **C**
  2
- **D**
  3

Correct Answer :A

Explanation

A destructor takes no arguments and has no return type. Its address cannot be taken. Destructors cannot be declared const , volatile , const volatile or static . A destructor can be declared virtual or pure virtual

#204 Explained Report Bookmark

Object-oriented programming is follows _____ programming aproch.

null

- **A**
  Top-down
- **B**
  Bottom-up
- **C**
  Flow-up
- **D**
  Follow-up

Correct Answer :B

Explanation

Object oriented programming languages like C++ and JAVA programming language follows bottom-up approach. A top-down approach begins with high level design and ends with low level design or development. A bottom-up approach begins with low level design or development and ends with high level design.

#205 Explained Report Bookmark

When a function is defined inside a class, this function is called _____

null

- **A**
  Inside function
- **B**
  Member function
- **C**
  Inline function
- **D**
  Interior function

Correct Answer :B

Explanation

A member function that is defined inside its class member list is called an inline member function. Member functions containing a few lines of code are usually declared inline.

#206 Explained Report Bookmark

.....are member functions of the class,which modifies the state of the object

null

- **A**
  constructor
- **B**
  Inspectors
- **C**
  Iterator
- **D**
  Mutators

Correct Answer :D

Explanation

A modifier, also called a modifying function, is a member function that changes the value of at least one data member. In other words, an operation that modifies the state of an object. Modifiers are also known as 'mutators'.

#207 Explained Report Bookmark

What is the output of this program?

```
#include <iostream>

    using namespace std;

    int func(int m = 10, int n)

    {
```

```
    int c;

    c = m + n;

    return c;

}

int main()

{

    cout << func(5);

    return 0;

}
```

- **A**
  10
- **B**
  15
- **C**
  Compile time error
- **D**
  Run time error

Correct Answer :C

Explanation

In function parameters the default arguments should always be the rightmost parameters.

#208 Explained Report Bookmark

Which is correct syntax for array declaration in cpp

- **A**
  arr int[5];
- **B**
  arr[5];
- **C**
  int arr[5];
- **D**
  var arr;

Correct Answer :C

Explanation

A typical declaration for an array in C++ is: type name [elements]; where type is a valid type (such as int, float ...), name is a valid identifier and the elements field (which is always enclosed in square brackets []), specifies the size of the array.

So int arr[5]; is correct answer

#209 Explained Report Bookmark

Which functions can be used to allocate memory for array dynamically

- **A**
  calloc()
- **B**
  malloc()
- **C**
  new()

- **D**
  Both 1 and 2

Correct Answer :D

Explanation

he <stdlib.h> library has functions responsible for dynamic memory management.

Function	Purpose
malloc	Allocates the memory of requested size and returns the pointer to the first byte of allocated space.
calloc	Allocates the space for elements of an array. Initializes the elements to zero and returns a pointer to the memory.
realloc	It is used to modify the size of previously allocated memory space.
Free	Frees or empties the previously allocated memory space.

#210 Explained Report Bookmark

Where does keyword ‘friend’ should be placed?

null

- A
  function declaration
- B
  function definition
- C
  main function
- D
  none of the mentioned

Correct Answer :A

Explanation

The keyword friend is placed only in the function declaration of the friend function and not in the function definition because it is used toaccess the member of a class

#211 Explained Report Bookmark

If a member function does not modify state of object only with in that member function , then that member function can be declared as _____

null

- A
  constant member function
- B
  private member function
- C
  static member function
- D
  friend member function

Correct Answer :A

Explanation

Make any member function that does not modify the state of the class object const, so that it can be called by const objects. Although instantiating const class objects is one way to create const objects, a more common way is by passing an object to a function by const reference

#212 Explained Report Bookmark

Run time polymorphism can be achieved by using_____

- **A**
  function overloading
- **B**
  function overriding
- **C**
  operator overloading
- **D**
  function overloading and operator overloading

Correct Answer :B

Explanation

The runtime polymorphism is achieved when the object method is invoked at the runtime instead of compile time. It is achieved by method overriding which is also known as dynamic binding/function overriding.

#213 Explained Report Bookmark

Static data members and member function are also known as _____

null

- A
  class members
- B
  instance members
- C
  both class and instance members
- D
  none of above

Correct Answer :A

Explanation

It is a variable which is declared with the static keyword, it is also known as class member, thus only single copy of the variable creates for all objects. Any changes in the static data member through one member function will reflect in all other object's member functions.

#214 Explained Report Bookmark

Which problem arises due to multiple inheritance, if hierarchical inheritance is used previously for its base classes

null

- A
  Diamond
- B
  Circle
- C
  Triangle
- D
  Loop

Correct Answer :A

Explanation

The diamond problem arises when multiple inheritance is used.

#215 Explained Report Bookmark

How many basic types of inheritance are provided as OOP feature

null

- **A**
  4
- **B**
  3
- **C**
  2
- **D**
  1

Correct Answer :A

Explanation

here are basically 4 types of inheritance provided in OOP, namely, single level, multilevel, multiple and hierarchical inheritance. We can add one more type as Hybrid inheritance but that is actually the combination any types of inheritance from the 4 basic ones.

#216 Explained Report Bookmark

In a class, encapsulating an object of another class is called

null

- A
  Composition
- B
  Inheritance
- C
  Encapsulation
- D
  None

Correct Answer :A

Explanation

Composition over inheritance (or composite reuse principle) in object-oriented programming (OOP) is the principle that classes should achieve polymorphic behavior and code reuse by their composition (by containing instances of other classes that implement the desired functionality) rather than inheritance from a base

#217 Explained Report Bookmark

If a derived class object is created, which constructor is called first_____

null

- A
  Base class constructor
- B
  Derived class constructor
- C
  Depends on how we call the object
- D
  Not possible

Correct Answer :A

Explanation

Base class constructors are always called in the derived class constructors. Whenever you create derived class object, first the base class default constructor is executed and then the derived class's constructor finishes execution.

#218 Explained Report Bookmark

Which are the two blocks that are used to check error and handle the error?

null

- **A**
  Try and catch
- **B**
  Trying and catching
- **C**
  Do and while
- **D**
  TryDo and Check

Correct Answer :A

Explanation

Two blocks that are used to check for errors and to handle the errors are try and catch block. The code which might produce some exceptions is placed inside the try block and then the catch block is written to catch the error that is produced.

#219 Explained Report Bookmark

Class which contains at least one pure virtual function such type of class is called as called _____

null

- **A**
  concreate class
- **B**
  abstract class
- **C**
  pure class
- **D**
  none of above

Correct Answer :B

Explanation

A class that contains at least one pure virtual function is considered an abstract class. Classes derived from the abstract class must implement the pure virtual function or they, too, are abstract classes

#220 Explained Report Bookmark

The following keyword is used before a function in a base class to be overridden in derived class in C++

null

- **A**
  override
- **B**
  virtual
- **C**
  void
- **D**
  none

Correct Answer :B

Explanation

C++11 adds two inheritance control keywords: override and final. override ensures that an overriding virtual function declared in a derived class has the same signature as that of the base class. final blocks further derivation of a class and further overriding of a virtual function

#221 Explained Report Bookmark

Data members and member functions of a class in C++ program are by default

null

- **A**
  protected
- **B**
  public
- **C**
  private
- **D**
  None

Correct Answer :C

Explanation

A class in C++ is a user-defined type or data structure declared with keyword class that has data and functions (also called member variables and member functions) as its members whose access is governed by the three access specifiers private, protected or public. By default access to members of a C++ class is private.

#222 Explained Report Bookmark

We should handle exception

null

- **A**
  To make code maintainable
- **B**
  To handle all the runtime errors at single place
- **C**
  To avoid resource leakage
- **D**
  All of the above

Correct Answer :D

Explanation

We should handle exception for all the above reasons mentioned in the question.

#223 Explained Report Bookmark

To handle exception, we should use _____ in C++

- **A**
  try handle
- **B**
  try catch handler
- **C**
  throw
- **D**
  None of the above

Correct Answer :B

Explanation

To handle exception, we should use catch handler in C++.

#224 Explained Report Bookmark

Which one of the following is true about new in C++?

null

- **A**
  new is object
- **B**
  new is function
- **C**
  new is operator
- **D**
  new is manipulator

Correct Answer :C

Explanation

The *new operator* is an *operator* which denotes a request for memory allocation on the Heap. If sufficient memory is available, *new operator* initializes the memory and returns the address of the newly allocated and initialized memory to the pointer variable.

#225 Explained Report Bookmark

Which one of the following is correct syntax of copy constructor

null

- A
  ClassName(ClassName other)
- B
  ClassName(const ClassName other)
- C
  ClassName(const ClassName *other)
- D
  ClassName(const ClassName &other)

Correct Answer :D

Explanation

A copy constructor is a member function which initializes an object using another object of the same class. A copy constructor has the following general function prototype: ClassName (const ClassName &old_obj);

#226 Explained Report Bookmark

In case of derived class object, choose correct statement about constructor and destructor?

null

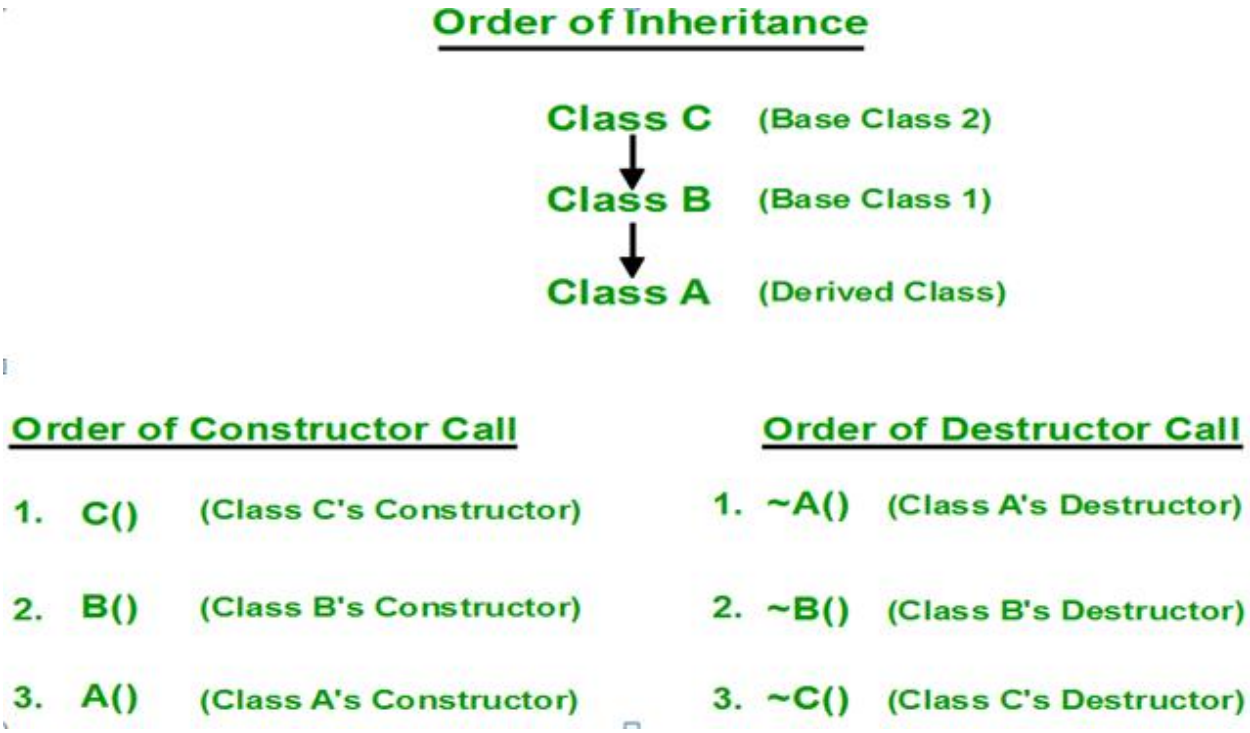
- A
  First base class and then derived class constructor will call. Destructor calling sequence is exactly opposite
- B
  First Derived class and then base class constructor will call. Destructor calling sequence is exactly opposite
- C
  First base class and then derived class constructor will call. Destructor calling sequence is exactly same

- D
  First Derived class and then base class constructor will call.
  Destructor calling sequence is exactly

Correct Answer :A

Explanation

Base class constructors are always called in the derived class constructors. Whenever you create derived class object, first the base class default constructor is executed and then the derived class's constructor finishes execution.d



#227 Explained Report Bookmark

Which one of the following member function do not inherit into derived class?

null

- **A**
  private member function
- **B**
  static member function
- **C**
  constant member function
- **D**
  assignment operator function

Correct Answer :B

Explanation

A static method, by definition, is at the class level, not the object level. However, you cannot instantiate that (meta)class, only an object of the class. As such, at the class level, there is no such thing as inheritance, as the class itself is static.

The private members of a class can be inherited but cannot be accessed directly by its derived classes. They can be accessed using public or protected methods of the base class.

The inheritance mode specifies how the protected and public data members are accessible by the derived classes. In C++, like other functions, assignment operator function is inherited in derived class.

For example, in the following program, base class assignment operator function can be accessed using the derived class object

#228 Explained Report Bookmark

Choose the correct statement about abstract class?

null

- A
  . We cannot extend abstract class
- B
  . We cannot create pointer or reference of abstract class
- C
  We cannot instantiate abstract class
- D
  All are correct

Correct Answer :C

Explanation

an abstract class in C++ is a class that has at least one pure virtual function (i.e., a function that has no definition). The classes inheriting the abstract class must provide a definition for the pure virtual function; otherwise, the subclass would become an abstract class itself. You cannot create an object of an abstract class type; however, you can use pointers and references to abstract class types. A class that contains at least one pure virtual function is considered an abstract class

#229 Explained Report Bookmark

Choose correct statement about function overriding?

null

- A
  For function overriding function in derived class must be virtual
- B
  For function overriding function in base class must be virtual
- C
  We can override static member function in derived class
- D
  For function overriding function must be exist is same scope

Correct Answer :B

Explanation

Function overriding is a feature that allows us to have a same function in child class which is already present in the parent class. A child class inherits the data members and member functions of parent class, but when you want to override functionality in the child class then you can use function overriding. Static methods cannot be overridden because method overriding only occurs in the context of dynamic (i.e. runtime) lookup of methods. Static methods (by their name) are looked up statically (i.e. at compile-time).

#230 Explained Report Bookmark

Which one of the following is correct syntax to declare function template or class template?

null

- **A**
  . template class<>;
- **B**
  template <class T>;
- **C**
  . <template class T >
- **D**
  Template <class> T

Correct Answer :B

Explanation

A template is a simple and yet very powerful tool in C++. The simple idea is to pass data type as a parameter so that we don’t need to write the same code for different data types.

#231 Explained Report Bookmark

Which one of the following is true about template?

null

- **A**
  Using Template, we can reduce code size.
- **B**
  Using Template, we can improve performance of the application
- **C**
  Using Template, we can reduce programming efforts
- **D**
  . All of the above

Correct Answer :D

Explanation

For me the most important advantages are:

- reducing the repetition of code (generic containers, algorithms)
- reducing the repetition of code advanced (MPL and Fusion)
- static polymorphism (=performance) and other compile time calculations
- policy based design (flexibility, reusability, easier changes, etc)
- increasing safety at no cost (i.e. dimension analysis via Boost Units, static assertions, concept checks)
- functional programming (Phoenix), lazy evaluation, expression templates (we can create Domain-specific embedded languages in C++, we have great Proto library, we have Blitz++)

- other less spectacular tools and tricks used in everyday life:
    - STL and the algorithms (what's the difference between `for` and `for_each`)
    - bind, lambda (or Phoenix) (write clearer code, simplify things)
    - Boost Function (makes writing callbacks easier)
    - tuples (how to genericly hash a tuple? Use Fusion for example...)
    - TBB (`parallel_for` and other STL like algorithms and containers)
- Can you imagine C++ without templates? Yes I can, in the early times you couldn't use them because of compiler limitations.
- Would you write in C++ without templates? No, as I would lose many of the advantages mentioned above.

Downsides:

- Compilation time (for example throw in Sprit, Phoenix, MPL and some Fusion and you can go for a coffee)
- People who can use and understand templates are not that common (and these people are useful)
- People who think that they can use and understand templates are quite common (and these people are dangerous, as they can make a hell out of your code. However most of them after some education/mentoring will join the group mentioned in the previous point)
- template `export` support (lack of)
- error messages could be less cryptic (after some learning you can find what you need, but still...)

#232 Explained Report Bookmark

If you want to do type conversion between primitive types then which one of the following operator we should use?

null

- **A**
  static_cast
- **B**
  dynamic_cast
- **C**
  const_cast
- **D**
  reinterpret_cast

Correct Answer :A

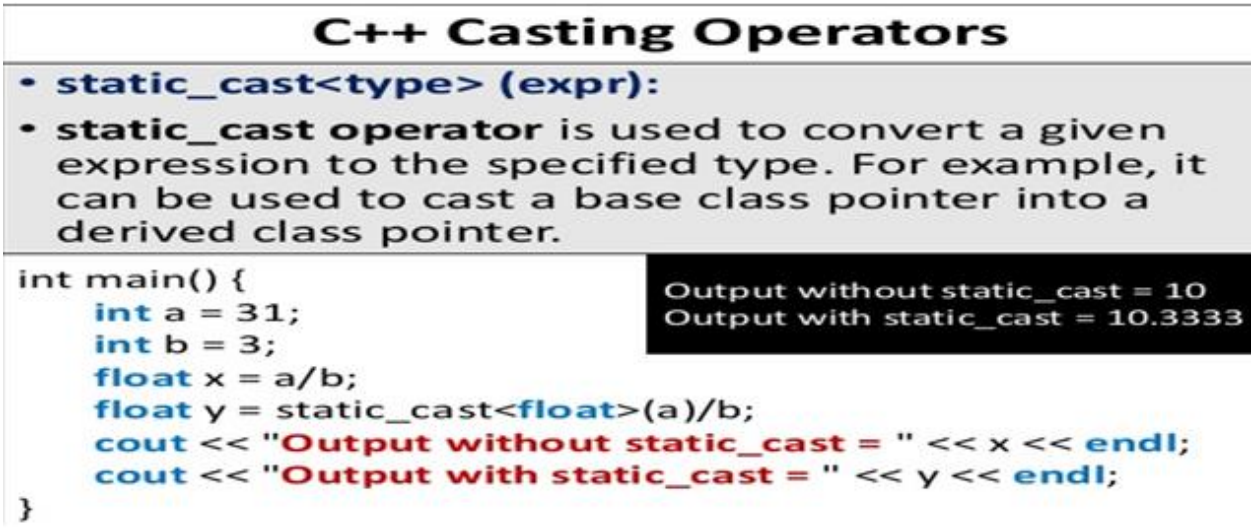
Explanation

another data type.

C++ supports four types of casting:

1. Static Cast
   2. Dynamic Cast
   3. Const Cast
   4. Reinterpret Cast

Static Cast: This is the simplest type of cast which can be used. It is a compile time cast.It does things like implicit conversions between types (such as int to float, or pointer to void*), and it can also call explicit conversion functions (or implicit ones).



#233 Explained Report Bookmark

Which of the two features match each other?

null

- **A**
  Inheritance and Encapsulation
- **B**
  Encapsulation and Polymorphism
- **C**
  Encapsulation and Abstraction
- **D**
  Abstraction and Polymorphism

Correct Answer :C

Explanation

Abstraction and Encapsulation look similar to each other by its definition. The standard definition of abstraction is showing only necessary features to the user and hide its implementation. On the other hand, Encapsulation is hiding the complexity from the outer world.

#234 Explained Report Bookmark

Which of the following operates cannot be overloaded?

i)Size of operator (sizeof)

ii)Scope resolution Operator

iii)Conditional operator (?:)

iv)Assignment Operator (=)

- **A**
  . i, ii, iii
- **B**
  ii, iii, iv
- **C**
  . i, iii, iv
- **D**
  i, ii, iii, iv

Correct Answer :A

Explanation

List of operators that can be overloaded are:

```
+    -    *    /      %        ^

&    |    ~    !,        =

     =       ++        --

     ==      !=       &&        ||
```

```
+=      -=      /=      %=          ^=          &=

|=      *=      =       []          ()

->      ->*     new     new []      delete      delete []
```

#235 Explained Report Bookmark

Which of the following is the least safe type casting in C++?

- A
  static_cast
- B
  const_cast
- C
  reinterpret_cast
- D
  dynamic_cast

Correct Answer :C

Explanation

Reinterpret *casts* are only available in *C++* and are the *least safe* form of *cast*, allowing the reinterpretation of the underlying bits of a value into another *type*.

#236 Explained Report Bookmark

To expose a data member to the program, you must declare the data member in the _____ section of the class

null

- A
  common

- **B**
  exposed
- **C**
  public
- **D**
  user

Correct Answer :C

Explanation

To expose a data member to the program, you must declare the data member in the public section of the class.

#237 Explained Report Bookmark

Use of pointers or reference to an abstract class gives rise to which among the following feature?

null

- **A**
  Static Polymorphism
- **B**
  Runtime polymorphism
- **C**
  Compile time Polymorphism
- **D**
  . Polymorphism within method

Correct Answer :B

Explanation

The runtime polymorphism is supported by reference and pointer to an abstract class. This relies upon base class pointer and reference to select the proper virtual function.

#238 Explained Report Bookmark

Which of the following decides the behaviour of the object?

null

- **A**
  data member
- **B**
  . member function
- **C**
  data member and member function both
- **D**
  none of the above

Correct Answer :B

Explanation

The behavior of a C++ object is defined by its member functions. All objects of a particular class share the member functions for that class. Every member function for a class must be declared with a function prototypein the class declaration.The term "behavior" refers to how objects interact with each other, and it is defined by the operations an object can perform

#239 Explained Report Bookmark

What is the use of Namespace?

null

- A
  To encapsulate the data
- B
  Encapsulate the data & structure a program into logical units
- C
  None of the mentioned
- D
  to combine data and functions together

Correct Answer :B

Explanation

The main aim of the namespace is to understand the logical units of the program and to make the program so robust.

#240 Explained Report Bookmark

Which of the following statement is incorrect about structure in c++?

null

- A
  we can encapsulate data member as well as member functions inside structure in c++
- B
  use of struct keyword,while declaring structure variable is optional
- C
  Default access specifier for members inside structure in c++ is private
- D
  none of the above

Correct Answer :C

Explanation

A class has all members private by default. A structure is a class where members are public by default.

#241 Explained Report Bookmark

In which of the following cases inline functions may not work

null

- **A**
  If the function has static variables and loops.
- **B**
  If the function has global and register variables.
- **C**
  If the function is recursive
- **D**
  both 1 and 3

Correct Answer :D

Explanation

A function is not inline if it has static variables, loops or the function is having any recursive calls.

#242 Explained Report Bookmark

which members get space inside object?

null

- **A**
  Non static data member
- **B**
  member functions

- C
  static as well as non-static data members
- D
  Only 1

Correct Answer :A

Explanation

static members are more like global objects accessed through A::j. Hence, they acts as global variable and do not need to be allocated different memory data and hence, they do not contribute to the size of objects.

#243 Explained Report Bookmark

which of the following statement is correct in context of class?

null

- A
  class is template for an object
- B
  Class is logical entity and it does not get space inside memory.
- C
  class is group of objects of same type i.e having same state and behaviour.
- D
  1 & 2

Correct Answer :D

Explanation

A class in C++ is the building block, that leads to Object-Oriented programming. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A C++ class is like a blueprint for an object.

#244 Explained Report Bookmark

How will you free the memory allocated by the following program?

```
#include<iostream>

int main()

{int *p=NULL;

    p=new int[10];

    return 0;

}
```

- **A**
  memfree(int p);
- **B**
  dealloc(p);
- **C**
  free(p);
- **D**
  delete[] p;

Correct Answer :D

Explanation

To free the dynamically allocated array pointed by pointer-variable, use following form of *delete*:

// Release block of memory

// pointed by pointer-variable

delete[] pointer-variable;

Example:

```
// It will free the entire array
// pointed by p.
delete[] p;
```

#245 Explained Report Bookmark

What is output of following code?

```
#include<iostream>
 using namespace std;
  int main()
   {
    int num1=10;
```

```
    int &num2=10;

    ++num2;

    cout<<"Num2::"<<num2;

     return 0;

}
}
```

- **A**
  10
- **B**
  11
- **C**
  compile time error
- **D**
  none of the above

Correct Answer :C

Explanation

Compile time error will be like this:

[Error] invalid initialization of non-const reference of type 'int&' from an rvalue of type 'int'.

#246 Explained Report Bookmark

Which of the following statements is correct?

1. A reference is not a constant pointer.
2. A referenced is automatically de-referenced.

- **A**
  Only 1 is correct.
- **B**
  Only 2 is correct
- **C**
  . Both 1 and 2 are correct
- **D**
  Both 1 and 2 are incorrect.

Correct Answer :B

Explanation

References are like constant pointers that are automatically dereferenced. It is a new name given to an existing storage. So when you are accessing the reference, you are actually accessing that storage. There is no need to use the * to dereference a reference variable.

#247 Explained Report Bookmark

Copy constructor must receive its arguments by _____

- **A**
  only pass-by-value
- **B**
  . only pass by address
- **C**
  nly pass-by-reference
- **D**
  either pass-by-value or pass-by-reference

Correct Answer :C

Explanation

Copy constructor must receive its arguments by only pass-by-reference. In this case, the reference to the argument in the calling function is passed to the equivalent formal parameter of the called function. The value of the argument in the called function can be modified by the passed in reference value.

#248 Explained Report Bookmark

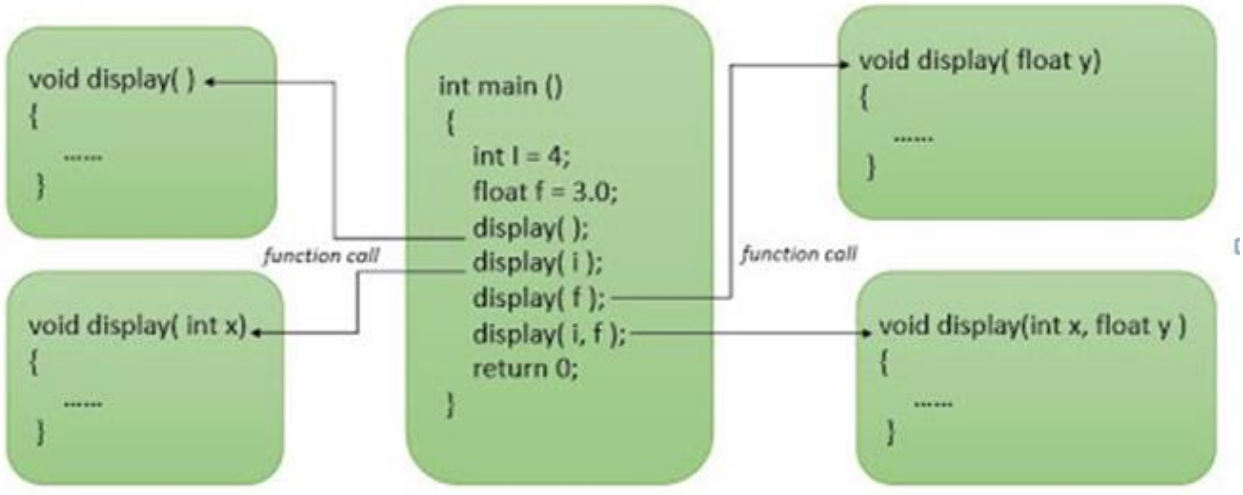
Which of the following permits function overloading in c++?

- **A**
  Argument type
- **B**
  Number of arguments
- **C**
  Sequence of arguments
- **D**
  . All of these

Correct Answer :D

Explanation

Function overloading is a C++ programming feature that allows us to have more than one function having same name but different parameter list, when I say parameter list, it means the data type and sequence of the parameters, for example the parameters list of a function myfuncn(int a, float b) is (int, float) which is different from the function myfuncn(float a, int b) parameter list (float, int). Function overloading is a compile-time polymorphism.



#249 Explained Report Bookmark

A static member function can be called using the _____ instead of its objects.

null

- **A** variable
- **B** function
- **C** class
- **D** object

Correct Answer :C

Explanation

Because static member functions are not attached to a particular object, they can be called directly by using the class name and the scope resolution operator.

#250 Explained Report Bookmark

In case of operator overloading, operator function must be _____

1. Static member functions

2. Non- static member functions

3. Friend Functions

- **A**
  Only 2
- **B**
  Only 1, 3
- **C**
  Only 2, 3
- **D**
  All 1, 2, 3

Correct Answer :C

Explanation

In case of operator overloading, operator function must be non-static member functions and friend functions.

#251 Explained Report Bookmark

State whether the following statements are True or False about the characteristics of static data members.

1. i) Only one copy of static data member is created for the entire class and is shared by all the objects of that class, no matter how many objects are created.
2. ii) we have to give global definition for static data member of class.

- A
  . i-True, ii-True
- B
  i-False, ii-True
- C
  i-True, ii-False
- D
  . i-False, ii-False

Correct Answer :A

Explanation

Defining and initializing *static member* variables Because *static member* variables *are* not part of the individual *class* objects (they *are* treated similarly to *global* variables, and *get* initialized when the program starts), *you must* explicitly *define* the *static member* outside of the *class*, in the *global* scope.

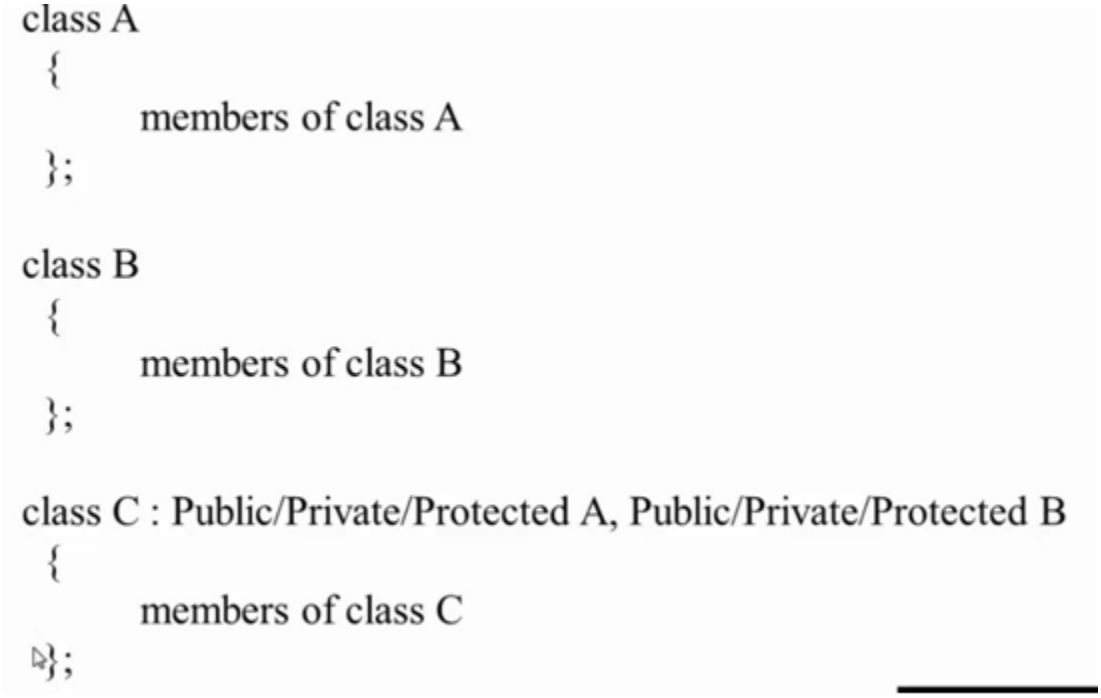
#252 Explained Report Bookmark

Which is the correct syntax of inheritance?

- A
  classderived_classname : base_classname{ /*define class body*/ };
- B
  classbase_classname : derived_classname{ /*define class body*/ };
- C
  _classname : mode of inheritance base_classname{ /*define class body*/ };
- D
  classbase_classname :mode of inheritance derived_classname{ /*define class body*/ };

Correct Answer :C

Explanation



#253 Explained Report Bookmark

types of composition are----------.

null

- **A**
  Containment
- **B**
  Association
- **C**
  Aggregation
- **D**
  all of above

Correct Answer :C

Explanation

There are two basic subtypes of object composition: composition and aggregation. The term “composition” is often used to refer to both composition and aggregation, not just to the composition subtype.

#254 Explained Report Bookmark

If I want to have common functions in a class and want to defer implementations of some other functions to derived classes, then we need to use

null

- **A**
  interface
- **B**
  abstract class
- **C**
  friend class
- **D**
  static class

Correct Answer :B

Explanation

If I want to have common functions in a class and want to defer implementations of some other functions to derived classes, then we need to use an abstract class.

#255 Explained Report Bookmark

What is up casting?

null

- **A**
  Storing address of derived class object into base class pointer
- **B**
  Storing address of derived class object into derived class pointer
- **C**
  Storing address of base class object into derived class pointer
- **D**
  Storing address of base class object into base class pointer

Correct Answer :A

Explanation

Upcasting is converting a derived-class reference or pointer to a base-class. In other words, upcasting allows us to treat a derived type as though it were its base type. It is always allowed for public inheritance, without an explicit type cast. This is a result of the is-a relationship between the base and derived classes

#256 Explained Report Bookmark

What is downcasting?

null

- **A**
  storing address of base class object into base class pointer
- **B**
  storing address of base class object into derived class pointer
- **C**
  . storing address of derived class object into derived class pointer
- **D**
  storing address of derived class object into base class pointer

Correct Answer :B

Explanation

The downcasting process, converting a base-class pointer (reference) to a derived-class pointer (reference) is called downcasting. Downcasting is not allowed without an explicit type cast. The reason for this restriction is that the is-a relationship is not, in most of the cases, symmetric. A derived class could add new data members, and the class member functions that used these data members wouldn't apply to the base class.

#257 Explained Report Bookmark

Which among the following is the proper syntax for the template class?

- **A**
  template <typename T1, typename T2>;
- **B**
  template <typename T> T named(T x, T y){ }
- **C**
  Template <typename T1, typename T2>;
- **D**
  Template <typename T1, typename T2> T1 named(T1 x, T2 y){ }

Correct Answer :B

Explanation

```
template<class T>
class Base
{
public:
  void set(const T& val) { data = val; }

private:
  T data;
};

template<class T>
class Derived : public Base<T>
{
public:
  void set(const T& val);
};


template<class T>
void Derived<T>::set(const T& v)
{
   // Call base-class behaviour
   Base<T>::set(v);
   // …plus any derived-class behaviour
}
```

Override base class behaviour

#258 Explained Report Bookmark

In C++, Class object created statically(e.g. Movie obj;) and dynamically (Movie *obj = new Movie() ;) are stored in memory

null

- **A**
  Stack, stack
- **B**
  Heap, heap
- **C**
  Heap, stack
- **D**
  Stack, heap

Correct Answer :D

Explanation

Stack is used for static memory allocation and Heap for dynamic memory allocation, both stored in the computer's RAM. Variables allocated on the stack are stored directly to the memory and access to this memory is very fast, and it's allocation is dealt with when the program is compiled.

#259 Explained Report Bookmark

Which of the following is an abstract data type?

null

- **A**
  Class
- **B**
  Int
- **C**
  String
- **D**
  Double

Correct Answer :A

Explanation

: An abstract data type (or ADT) is a class that has a defined set of operations and values. In other words, you can create the starter motor as an entire abstract data type, protecting all of the inner code from the user.

#260 Explained Report Bookmark

If I want to have common functions in a class and want to defer implementations of some other functions to derived classes, then we need to use

null

- **A**
  An interface
- **B**
  An abstract class
- **C**
  A friend class
- **D**
  A static class

Correct Answer :B

Explanation

If I want to have common functions in a class and want to defer implementations of some other functions to derived classes, then we need to use an abstract class.

#261 Explained Report Bookmark

Overloaded functions in C++ oops are

null

- **A**
  Functionspreceding with virtual keywords
- **B**
  Functions inherited from base class to derived class.
- **C**
  Two or more functions having same name but different number of parameters or type.

- **D**
  None of above

Correct Answer :C

Explanation

C++ allows specification of more than one function of the same name in the same scope. These functions are called overloaded functions. Overloaded functions enable you to supply different semantics for a function, depending on the types and number of arguments.

#262 Explained Report Bookmark

. C++ abstract class can contain

null

- **A**
  Pure virtual function
- **B**
  Non-virtual function
- **C**
  Only pure virtual function
- **D**
  Both pure virtual and non-virtual function

Correct Answer :D

Explanation

Abstract class cannot be instantiated, but pointers and refrences of Abstract class type can be created. Abstract class can have normal functions and variables along with a pure virtual function. Abstract classes

are mainly used for Upcasting, so that its derived classes can use its interface.

#263 Explained Report Bookmark

Which function did not get this pointer?

null

- **A**
  Static function
- **B**
  Friend function
- **C**
  . Global function
- **D**
  All of the above

Correct Answer :D

Explanation

Friend functions do not have a this pointer, because friends are not members of a class.Because static member functions are not attached to an object, they have no *this* pointer! This makes sense when you think about it -- the *this* pointer always points to the object that the member function is working on. Static member functions do not work on an object, so the *this* pointer is not needed.

#264 Explained Report Bookmark

How structures and classes in C++ differ?

- **A**
  In Structures, members are private by default whereas, in Classes, they are public by default

- **B**
  In Structures, members are public by default whereas, in Classes, they are private by default
- **C**
  Structures by default hide every member whereas classes do not
- **D**
  Structures cannot have private members whereas classes can have

Correct Answer :B

Explanation

Structure members are public by default whereas, class members are private by default. Both of them can have private and public members.

#265 Explained Report Bookmark

Which is not feature of OOP in general definitions?

null

- **A**
  Code reusability
- **B**
  Modularity
- **C**
  Duplicate/Redundant data
- **D**
  Efficient Code

Correct Answer :C

Explanation

Duplicate/Redundant data is dependent on programmer and hence can’t be guaranteed by OOP. Code reusability is done using inheritance. Modularity is supported by using different code files and classes. Codes are more efficient because of features of OOP.

#266 Explained Report Bookmark

Which Feature of OOP illustrated the code reusability?

null

- **A**
  Polymorphism
- **B**
  Abstraction
- **C**
  Encapsulation
- **D**
  Inheritance

Correct Answer :D

Explanation

Using inheritance we can reuse the code already written and also can avoid creation of many new functions or variables, as that can be done one time and be reused, using classes.

#267 Explained Report Bookmark

How many classes can be defined in a single program?

- **A**
  Only 1
- **B**
  Only 100

- C
  As many as you want
- D
  Only 999

Correct Answer :C

Explanation

Any number of classes can be defined inside a program, provided that their names are different. In java, if public class is present then it must have the same name as that of file.

#268 Explained Report Bookmark

When OOP concept did first came into picture?

null

- A
  1970’s
- B
  1980’s
- C
  1993
- D
  1995

Correct Answer :A

Explanation

OOP first came into picture in 1970’s by Alan and his team. Later it was used by some programming languages and got implemented successfully,

SmallTalk was first language to use pure OOP and followed all rules strictly.

#269 Explained Report Bookmark

Which concept of OOP is false for C++?

null

- **A**
  Code can be written without using classes
- **B**
  Code must contain at least one class
- **C**
  A class must have member functions
- **D**
  At least one object should be declared in code

Correct Answer :B

Explanation

In C++, it’s not necessary to use classes, and hence codes can be written without using OOP concept. Classes may or may not contain member functions, so it’s not a necessary condition in C++. And, an object can only be declared in a code if its class is defined/included via header file.

#270 Explained Report Bookmark

Which header file is required in C++ to use OOP?

null

- **A**
  iostream.h

- B
  stdio.h
- C
  stdlib.h
- D
  OOP can be used without using any header file

Correct Answer :D

Explanation

We need not include any specific header file to use OOP concept in C++, only specific functions used in code need their respective header files to be included or classes should be defined if needed.

#271 Explained Report Bookmark

Which feature allows open recursion, among the following?

null

- A
  Use of this pointer
- B
  Use of pointers
- C
  Use of pass by value
- D
  Use of parameterized constructor

Correct Answer :A

Explanation

Use of this pointer allows an object to call data and methods of itself whenever needed. This helps us call the members of an object recursively, and differentiate the variables of different scopes.

#272 Explained Report Bookmark

Which definition best describes an object?

null

- **A**
  Instance of a class
- **B**
  Instance of itself
- **C**
  Child of a class
- **D**
  Overview of a class

Correct Answer :A

Explanation

An object is instance of its class. It can be declared in the same way that a variable is declared, only thing is you have to use class name as the data type.

#273 Explained Report Bookmark

Which among the following is false?

null

- **A**
  Object must be created before using members of a class

- **B**
  Memory for an object is allocated only after its constructor is called
- **C**
  Objects can’t be passed by reference
- **D**
  Objects size depends on its class data members

Correct Answer :C

Explanation

Objects can be passed by reference. Objects can be passed by value also. If the object of a class is not created, we can’t use members of that class.

#274 Explained Report Bookmark

Which of the following is incorrect?

null

- **A**
  class student{ }s;
- **B**
  class student{ }; student s;
- **C**
  class student{ }s[];
- **D**
  class student{ }; student s[5];

Correct Answer :C

Explanation

The array must be specified with a size. You can’t declare object array, or any other linear array without specifying its size. It’s a mandatory field.

#275 Explained Report Bookmark

The object can’t be _____

null

- **A**
  Passed by reference
- **B**
  Passed by value
- **C**
  Passed by copy
- **D**
  Passed as function

Correct Answer :D

Explanation

Object can’t be passed as function as it is an instance of some class, it’s not a function. Object can be passed by reference, value or copy. There is no term defined as pass as function for objects.

#276 Explained Report Bookmark

What is size of the object of following class (64 bit system)?

```
class student

{

    int rollno;
```

```
    char  name[20];

    static int studentno;

};
```

- **A**
  20
- **B**
  22
- **C**
  24
- **D**
  28

Correct Answer :C

Explanation

The size of any object of student class will be of size 4+20=24,

because static members are not really considered as property of a single object. So static variables size will not be added.

#277 Explained Report Bookmark

How members of an object are accessed?

- **A**
  Using scope resolution operator
- **B**
  Using member names directly
- **C**
  Using pointer only
- **D**
  Using dot operator/period symbol

Correct Answer :D

Explanation

Using dot operator after the name of object we can access its members. It is not necessary to use the pointers. We can't use the names directly because it may be used outside the class.

#278 Explained Report Bookmark

If a local class is defined in a function, which of the following is true for an object of that class?

null

- **A**
  Object is accessible outside the function
- **B**
  Object can be declared inside any other function
- **C**
  Object can be used to call other class members
- **D**
  Object can be used/accessed/declared locally in that function

Correct Answer :D

Explanation

For an object which belongs to a local class, it is mandatory to declare and use the object within the function because the class is accessible locally within the class only.

#279 Explained Report Bookmark

Which among the following is wrong?

- **A**
  class student{ }; student s;
- **B**
  abstract class student{ }; class toppers: public student{ }; topper t;
- **C**
  abstract class student{ }s[50000000];
- **D**
  abstract class student{ }; student s;

Correct Answer :D

Explanation

We can never create instance of an abstract class. Abstract classes doesn't have constructors and hence when an instance is created there is no facility to initialize its members. Option d is correct because topper class is inheriting the base abstract class student, and hence topper class object can be created easily.

#280 Explained Report Bookmark

Object declared in main() function _____

null

- **A**
  Can be used by any other function
- **B**
  Can be used by main() function of any other program
- **C**
  Can't be used by any other function
- **D**
  Can be accessed using scope resolution operator

Correct Answer :C

Explanation

The object declared in main() have local scope inside main() function only. It can't be used outside main() function. Scope resolution operator is used to access globally declared variables/objects.

#281 Explained Report Bookmark

When an object is returned_____

null

- **A**
  A temporary object is created to return the value
- **B**
  The same object used in function is used to return the value
- **C**
  The Object can be returned without creation of temporary object
- **D**
  Object are returned implicitly, we can't say how it happens inside program

Correct Answer :A

Explanation

A temporary object is created to return the value. It is created because the object used in function is destroyed as soon as the function is returned. The temporary variable returns the value and then gets destroyed.

#282 Explained Report Bookmark

Which among the following is correct?

null

- **A**
  class student{ }s1,s2; s1.student()=s2.student();
- **B**
  class student{ }s1; class topper{ }t1; s1=t1;
- **C**
  class student{ }s1,s2; s1=s2;
- **D**
  class student{ }s1; class topper{ }t1; s1.student()=s2.topper();

Correct Answer :C

Explanation

Only if the objects are of same class then their data can be copied from to another using assignment operator. This actually comes under operator overloading. Class constructors can’t be assigned any explicit value as in option class student{ }s1; class topper{ }t1; s1=t1; and class student{ }s1; class topper{ }t1; s1.student()=s2.topper();.

#283 Explained Report Bookmark

If a function can perform more than 1 type of tasks, where the function name remains same, which feature of OOP is used here?

null

- **A**
  Encapsulation
- **B**
  Inheritance
- **C**
  Polymorphism

- D
  Abstraction

Correct Answer :C

Explanation

For the feature given above, the OOP feature used is Polymorphism. Example of polymorphism in real life is a kid, who can be a student, a son, a brother depending on where he is.

#284 Explained Report Bookmark

If different properties and functions of a real world entity is grouped or embedded into a single element, what is it called in OOP language?

null

- A
  Inheritance
- B
  Polymorphism
- C
  Abstraction
- D
  Encapsulation

Correct Answer :D

Explanation

It is Encapsulation, which groups different properties and functions of a real world entity into single element. Abstraction, on other hand, is hiding of

functional or exact working of codes and showing only the things which are required by the user.

#285 Explained Report Bookmark

Which of the following is not a feature of pure OOP?

null

- **A**
  Classes must be used
- **B**
  Inheritance
- **C**
  Data may/may not be declared using object
- **D**
  Functions Overloading

Correct Answer :C

Explanation

Data must be declared using objects. Object usage is mandatory because it in turn calls its constructors, which in turn must have a class defined. If object is not used, it is a violation of pure OOP concept.

#286 Explained Report Bookmark

Which among the following doesn’t come under OOP concept?

null

- **A**
  Platform independent
- **B**
  Data binding

- C
  Message passing
- D
  Data hiding

Correct Answer :A

Explanation

Platform independence is not feature of OOP. C++ supports OOP but it’s not a platform independent language. Platform independence depends on programming language.

#287 Explained Report Bookmark

Which feature of OOP is indicated by the following code?

```
class student{  int marks;  };

class topper:public student{  int age;  topper(int age){
this.age=age; } };
```

- A
  Inheritance
- B
  Polymorphism
- C
  nheritance and polymorphism
- D
  Encapsulation and Inheritance

Correct Answer :D

Explanation

Encapsulation is indicated by use of classes. Inheritance is shown by inheriting the student class into topper class. Polymorphism is not shown here because we have defined the constructor in the topper class but that doesn’t mean that default constructor is overloaded.

#288 Explained Report Bookmark

How many basic features of OOP are required for a programming language to be purely OOP?

- **A**
  7
- **B**
  6
- **C**
  5
- **D**
  4

Correct Answer :A

Explanation

There are 7 basic features that define whether a programing language is pure OOP or not.

The 4 basic features are inheritance, polymorphism, encapsulation and abstraction.

Further, one is, object use is must, secondly, message passing and lastly, Dynamic binding.

#289 Explained Report Bookmark

The feature by which one object can interact with another object is _____

null

- **A** Data transfer
- **B** Data Binding
- **C** Message Passing
- **D** Message reading

Correct Answer :C

Explanation

The interaction between two object is called the message passing feature. Data transfer is not a feature of OOP. Also, message reading is not a feature of OOP.

#290 Explained Report Bookmark

Which feature in OOP is used to allocate additional function to a predefined operator in any language?

null

- A
  Operator Overloading
- B
  Function Overloading
- C
  Operator Overriding
- D
  Function Overriding

Correct Answer :A

Explanation

The feature is operator overloading. There is not a feature named operator overriding specifically. Function overloading and overriding doesn’t give addition function to any operator.

#291 Explained Report Bookmark

Which among doesn’t illustrates polymorphism?

null

- A
  Function overloading
- B
  Function overriding
- C
  Operator overloading
- D
  Virtual function

Correct Answer :B

Explanation

Function overriding doesn’t illustrate polymorphism because the functions are actually different and theirs scopes are different. Function and operator overloading illustrate proper polymorphism. Virtual functions show polymorphism because all the classes which inherit virtual function, define the same function in different ways.

#292 Explained Report Bookmark

Which among the following best describes polymorphism?

null

- **A**
  It is the ability for a message/data to be processed in more than one form
- **B**
  It is the ability for a message/data to be processed in only 1 form
- **C**
  It is the ability for many messages/data to be processed in one way
- **D**
  It is the ability for undefined message/data to be processed in at least one way

Correct Answer :A

Explanation

It is actually the ability for a message / data to be processed in more than one form. The word polymorphism indicates many-forms. So if a single entity takes more than one form, it is known as polymorphism.

#293 Explained Report Bookmark

Which class/set of classes can illustrate polymorphism in the following code?

```
abstract class student

{

   public : int marks;

   calc_grade();

}

class topper:public student

{

    public : calc_grade()

    {

        return 10;

    }

};

class average:public student

{

     public : calc_grade()

     {

         return 20;
```

```
    }

};

class failed{ int marks; };
```

- **A**
  Only class student can show polymorphism
- **B**
  Only class student and topper together can show polymorphism
- **C**
  All class student, topper and average together can show polymorphism
- **D**
  Class failed should also inherit class student for this code to work for polymorphism

Correct Answer :C

Explanation

Since Student class is abstract class and class topper and average are inheriting student, class topper and average must define the function named calc_grade(); in abstract class. Since both the definition are different in those classes, calc_grade() will work in different way for same input from different objects. Hence it shows polymorphism.

#294 Explained Report Bookmark

Which type of function among the following shows polymorphism?

- **A**
  Virtual function
- **B**
  Undefined functions

- C
  Class member functions
- D
  Inline function

Correct Answer :A

Explanation

Only virtual functions among these can show polymorphism. Class member functions can show polymorphism too but we should be sure that the same function is being overloaded or is a function of abstract class or something like this, since we are not sure about all these, we can’t say whether it can show polymorphism or not.

#295 Explained Report Bookmark

In case of using abstract class or function overloading, which function is supposed to be called first?

null

- A
  Local function
- B
  Function with highest priority in compiler
- C
  Global function
- D
  Function with lowest priority because it might have been halted since long time, because of low priority

Correct Answer :B

Explanation

Function with highest priority is called. Here, it’s not about the thread scheduling in CPU, but it focuses on whether the function in local scope is present or not, or if scope resolution is used in some way, or if the function matches the argument signature. So all these things define which function has the highest priority to be called in runtime. Local function could be one of the answer but we can’t say if someone have used pointer to another function or same function name.

#296 Explained Report Bookmark

W.hich among the following can’t be used for polymorphi

null

- A
  Static member functions
- B
  Member functions overloading
- C
  Predefined operator overloading
- D
  Constructor overloading

Correct Answer :A

Explanation

Static member functions are not property of any object. Hence it can’t be considered for overloading/overriding. For polymorphism, function must be property of object, not only of class.

#297 Explained Report Bookmark

Which among the following can show polymorphism?

null

- A
  Overloading ||
- B
  Overloading +=
- C
  Overloading <<
- D
  Overloading &&

Correct Answer :C

Explanation

Only insertion operator can be overloaded among all the given options. And the polymorphism can be illustrated here only if any of these is applicable of being overloaded. Overloading is type of polymorphism.

#298 Explained Report Bookmark

If data members are private, what can we do to access them from the class object?

null

- A
  Create public member functions to access those data members
- B
  Create private member functions to access those data members
- C
  Create protected member functions to access those data members
- D
  Private data members can never be accessed from outside the class

Correct Answer :A

Explanation

We can define public member functions to access those private data members and get their value for use or alteration. They can’t be accessed directly but is possible to be access using member functions. This is done to ensure that the private data doesn’t get modified accidentally.

#299 Explained Report Bookmark

Which feature can be implemented using encapsulation?

null

- A
  Inheritance
- B
  Abstraction
- C
  Polymorphism
- D
  Overloading

Correct Answer :B

Explanation

Data abstraction can be achieved by using encapsulation. We can hide the operation and structure of actual program from the user and can show only required information by the user.

#300 Explained Report Bookmark

Find which of the following uses encapsulation?

null

- A
  void main(){ int a; void fun(int a=10; cout<<a); fun(); }
- B
  class student{ int a; public: int b;};
- C
  class student{int a; public: void disp(){ cout<<a;} };
- D
  struct topper{ char name[10]; public : int marks; }

Correct Answer :C

Explanation

It is the class which uses both the data members and member functions being declared inside a single unit. Only data members can be there in structures also. And the encapsulation can only be illustrated if some data/operations are associated within class.

#301 Explained Report Bookmark

How can Encapsulation be achieved?

null

- A
  Using Access Specifiers
- B
  Using only private members
- C
  Using inheritance
- D
  Using Abstraction

Correct Answer :A

Explanation

Using access specifiers we can achieve encapsulation. Using this we can in turn implement data abstraction. It’s not necessary that we only use private access.

#302 Explained Report Bookmark

Which among the following would destroy the encapsulation mechanism if it was allowed in programming?

null

- **A**
  Using access declaration for private members of base class
- **B**
  Using access declaration for public members of base class
- **C**
  Using access declaration for local variable of main() function
- **D**
  Using access declaration for global variables

Correct Answer :A

Explanation

If using access declaration for private members of base class was allowed in programming, it would have destroyed whole concept of encapsulation. As if it was possible, any class which gets inherited privately, would have been able to inherit the private members of base class, and hence could access each and every member of base class.

#303 Explained Report Bookmark

Which among the following can be a concept against encapsulation rules?

null

- A
  Using function pointers
- B
  Using char* string pointer to be passed to non-member function
- C
  Using object array
- D
  Using any kind of pointer/array address in passing to another function

Correct Answer :D

Explanation

If we use any kind of array or pointer as data member which should not be changed, but in some case its address is passed to some other function or similar variable. There are chances to modify its whole data easily. Hence Against encapsulation.

#304 Explained Report Bookmark

Using encapsulation data security is _____

null

- A
  Not ensured
- B
  Ensured to some extent
- C
  Purely ensured
- D
  Very low

Correct Answer :B

Explanation

The encapsulation can only ensure data security to some extent. If pointer and addresses are misused, it may violate encapsulation. Use of global variables also makes the program vulnerable, hence we can’t say that encapsulation gives pure security.

#305 Explained Report Bookmark

Hiding the implementation complexity can _____

null

- **A**
  Make the programming easy
- **B**
  Make the programming complex
- **C**
  Provide more number of features
- **D**
  Provide better features

Correct Answer :A

Explanation

It can make programming easy. The programming need not know how the inbuilt functions are working but can use those complex functions directly in the program. It doesn’t provide more number of features or better features.

#306 Explained Report Bookmark

Class is _____ abstraction.

null

- A
  Object
- B
  Logical
- C
  Real
- D
  Hypothetical

Correct Answer :B

Explanation

Class is logical abstraction because it provides a logical structure for all of its objects. It gives an overview of the features of an object.

#307 Explained Report Bookmark

Object is _____ abstraction.

null

- A
  Object
- B
  Logical
- C
  Real
- D
  Hypothetical

Correct Answer :C

Explanation

Object is real abstraction because it actually contains those features of class. It is the implementation of overview given by class. Hence the class is logical abstraction and its object is real.

#308 Explained Report Bookmark

Which among the following can be viewed as combination of abstraction of data and code.

null

- A
  Class
- B
  Object
- C
  Inheritance
- D
  Interfaces

Correct Answer :B

Explanation

Object can be viewed as abstraction of data and code. It uses data members and their functioning as data abstraction. Code abstraction as use of object of inbuilt class.

#309 Explained Report Bookmark

Encapsulation and abstraction differ as _____

null

- A
  Binding and Hiding respectively

- B
  Hiding and Binding respectively
- C
  Can be used any way
- D
  Hiding and hiding respectively

Correct Answer :A

Explanation

Abstraction is hiding the complex code. For example, we directly use cout object in C++ but we don’t know how is it actually implemented. Encapsulation is data binding, as in, we try to combine a similar type of data and functions together.

#310 Explained Report Bookmark

If two classes combine some private data members and provides public member functions to access and manipulate those data members. Where is abstraction used?

null

- A
  Using private access specifier for data members
- B
  Using class concept with both data members and member functions
- C
  Using public member functions to access and manipulate the data members
- D
  Data is not sufficient to decide what is being used

Correct Answer :C

Explanation

It is the concept of hiding program complexity and actual working in background. Hence use of public member functions illustrates abstraction here.

#311 Explained Report Bookmark

A phone is made up of many components like motherboard, camera, sensors and etc. If the processor represents all the functioning of phone, display shows the display only, and the phone is represented as a whole. Which among the following have highest level of abstraction?

null

- **A**
  Motherboard
- **B**
  Display
- **C**
  Camera
- **D**
  Phone

Correct Answer :D

Explanation

Phone as a whole have the highest level of abstraction. This is because the phone being a single unit represents the whole system. Whereas motherboard, display and camera are its components.

#312 Explained Report Bookmark

Which among the following is called first, automatically, whenever an object is created?

null

- **A**
  Class
- **B**
  Constructor
- **C**
  New
- **D**
  Trigger

Correct Answer :B

Explanation

Constructors are the member functions which are called automatically whenever an object is created. It is a mandatory functions to be called for an object to be created as this helps in initializing the object to a legal initial value for the class.

#313 Explained Report Bookmark

Which among the following is not a necessary condition for constructors?

null

- **A**
  Its name must be same as that of class
- **B**
  It must not have any return type
- **C**
  It must contain a definition body

- **D**
  It can contains arguments

Correct Answer :C

Explanation

Constructors are predefined implicitly, even if the programmer doesn't define any of them. Even if the programmer declares a constructor, it's not necessary that it must contain some definition.

#314 Explained Report Bookmark

Which among the following is correct?

null

- **A**
  class student{ public: int student(){} };
- **B**
  class student{ public: void student (){} };
- **C**
  class student{ public: student{}{} };
- **D**
  class student{ public: student(){} };

Correct Answer :D

Explanation

The constructors must not have any return type. Also, the body may or may not contain any body. Defining default constructor is optional, if you are not using any other constructor.

#315 Explained Report Bookmark

In which access should a constructor be defined, so that object of the class can be created in any function?

null

- **A**
  Public
- **B**
  Protected
- **C**
  Private
- **D**
  Any access specifier will work

Correct Answer :A

Explanation

Constructor function should be available to all the parts of program where the object is to be created. Hence it is advised to define it in public access, so that any other function is able to create objects.

#316 Explained Report Bookmark

How many types of constructors are available for use in general (with respect to parameters)?

null

- **A**
  2
- **B**
  3

- **C**
  4
- **D**
  5

Correct Answer :A

Explanation

Two types of constructors are defined generally, namely, default constructor and parameterized constructor. Default constructor is not necessary to be defined always.

#317 Explained Report Bookmark

If a programmer defines a class and defines a default value parameterized constructor inside it. He has not defined any default constructor. And then he try to create the object without passing arguments, which among the following will be correct?

null

- **A**
  It will not create the object (as parameterized constructor is used)
- **B**
  It will create the object (as the default arguments are passed)
- **C**
  It will not create the object (as the default constructor is not defined)
- **D**
  It will create the object (as at least some constructor is defined)

Correct Answer :B

Explanation

It will create the object without any problem, because the default arguments use the default value if no value is passed. Hence it is equal to default constructor with zero parameters. But it will not create the object if signature doesn't match.

#318 Explained Report Bookmark

If class C inherits class B. And B has inherited class A. Then while creating the object of class C, what will be the sequence of constructors getting called?

null

- **A**
  Constructor of C then B, finally of A
- **B**
  Constructor of A then C, finally of B
- **C**
  Constructor of C then A, finally B
- **D**
  Constructor of A then B, finally C

Correct Answer :D

Explanation

While creating the object of class C, its constructor would be called by default. But, if the class is inheriting some other class, firstly the parent class constructor will be called so that all the data is initialized that is being inherited.

#319 Explained Report Bookmark

In multiple inheritance, if class C inherits two classes A and B as follows, which class constructor will be called first?

```
class A{ };

class B{ };

class C: public A, public B{  };
```

- **A**
  A()
- **B**
  B()
- **C**
  C()
- **D**
  Can’t be determined

Correct Answer :A

Explanation

Constructor of class A will be called first. This is because the constructors in multiple inheritance are called in the sequence in which they are written to be inherited. Here A is written first, hence it is called first.

#320 Explained Report Bookmark

If the object is passed by value to a copy constructor?

- **A**
  Only public members will be accessible to be copied
- **B**
  That will work normally
- **C**
  Compiler will give out of memory error
- **D**
  Data stored in data members won’t be accessible

Correct Answer :C

Explanation

Compiler runs out of memory. This is because while passing the argument by value, a constructor of the object will be called. That in turn called another object constructor for values, and this goes on. This is like a constructor call to itself, and this goes on infinite times, hence it must be passed by reference, so that the constructor is not called.

#321 Explained Report Bookmark

Which object will be created first?

```
class student

{

    int marks;

};

student s1, s2, s3;
```

- **A**
  s1 then s2 then s3
- **B**
  s3 then s2 then s1
- **C**
  s2 then s3 then s1
- **D**
  all are created at same time

Correct Answer :A

Explanation

The objects are created in the sequence of how they are written. This happens because the constructors are called in the sequence of how the objects are mentioned. This is done in sequence.

#322 Explained Report Bookmark

Copy constructor is a constructor which _____

- **A**
  Creates an object by copying values from first object created for that class
- **B**
  Creates an object by copying values from another object of another class
- **C**
  Creates an object by copying values from any other object of same class
- **D**
  Creates an object by initializing it with another previously created object of same class

Correct Answer :D

Explanation

The object that has to be copied to new object must be previously created. The new object gets initialized with the same values as that of the object mentioned for being copied. The exact copy is made with values.

#323 Explained Report Bookmark

The copy constructor can be used to _____

null

- A
  Initialize one object from another object of same type
- B
  Initialize one object from another object of different type
- C
  Initialize more than one object from another object of same type at a time
- D
  Initialize all the objects of a class to another object of another class

Correct Answer :A

Explanation

The copy constructor has the most basic function to initialize the members of an object with same values as that of some previously created object. The object must be of same class.

#324 Explained Report Bookmark

If two classes have exactly same data members and member function and only they differ by class name. Can copy constructor be used to initialize one class object with another class object?

null

- A
  Yes, possible
- B
  Yes, because the members are same
- C
  No, not possible
- D
  No, but possible if constructor is also same

Correct Answer :C

Explanation

The restriction for copy constructor is that it must be used with the object of same class. Even if the classes are exactly same the constructor won't be able to access all the members of another class. Hence we can't use object of another class for initialization.

#325 Explained Report Bookmark

The copy constructors can be used to _____

null

- **A**
  Copy an object so that it can be passed to a class
- **B**
  Copy an object so that it can be passed to a function
- **C**
  Copy an object so that it can be passed to another primitive type variable
- **D**
  Copy an object for type casting

Correct Answer :B

Explanation

When an object is passed to a function, actually its copy is made in the function. To copy the values, copy constructor is used. Hence the object being passed and object being used in function are different.

#326 Explained Report Bookmark

If programmer doesn’t define any copy constructor then _____

null

- **A**
  Compiler provides an implicit copy constructor
- **B**
  Compiler gives an error
- **C**
  The objects can’t be assigned with another objects
- **D**
  The program gives run time error if copying is used

Correct Answer :A

Explanation

The compiler provides an implicit copy constructor. It is not mandatory to always create an explicit copy constructor. The values are copied using implicit constructor only.

#327 Explained Report Bookmark

What is the syntax of copy constructor?

- **A**
  classname (classname &obj){ /*constructor definition*/ }
- **B**
  classname (const classname obj){ /*constructor definition*/ }
- **C**
  classname (const classname &obj){ /*constructor definition*/ }
- **D**
  classname (const &obj){ /*constructor definition*/ }

Correct Answer :C

Explanation

The syntax must contain the class name first, followed by the classname as type and &object within parenthesis. Then comes the constructor body. The definition can be given as per requirements.

#328 Explained Report Bookmark

The arguments to a copy constructor _____

null

- **A**
  Must be const
- **B**
  Must not be cosnt
- **C**
  Must be integer type
- **D**
  Must be static

Correct Answer :A

Explanation

The object should not be modified in the copy constructor. Because the object itself is being copied. When the object is returned from a function, the object must be a constant otherwise the compiler creates a temporary object which can die anytime.

#329 Explained Report Bookmark

Which constructor will be called from the object created in the code below?

```
class A

{

     int i;

     A()

     {

          i=0; cout&lt;&lt;i;

     }

     A(int x=0)

     {

          i=x;  cout&lt;&lt;I;

     }

};

A obj1;
```

- **A**
  Default constructor
- **B**
  Parameterized constructor
- **C**
  Compile time error
- **D**
  Run time error

Correct Answer :C

Explanation

When a default constructor is defined and another constructor with 1 default value argument is defined, creating object without parameter will create ambiguity for the compiler. The compiler won't be able to decide which constructor should be called, hence compile time error.

#330 Explained Report Bookmark

When a destructor is called?

- A
  Anytime in between object's lifespan
- B
  At end of whole program
- C
  Just before the end of object life
- D
  After the end of object life

Correct Answer :C

Explanation

The destructor is called just before the object go out of scope or just before its life ends. This is done to ensure that all the resources reserved for the object are used and at last, are made free for others.

#331 Explained Report Bookmark

If in multiple inheritance, class C inherits class B, and Class B inherits class A. In which sequence are their destructors called if an object of class C was declared?

- A
  ~A() then ~B() then ~C()
- B
  ~B() then ~C() then ~A()
- C
  ~C() then ~B() then ~A()
- D
  ~C() then ~A() then ~B()

Correct Answer :C

Explanation

The destructors are always called in the reverse order of how the constructors were called. Here class A constructor would have been created first if Class C object is declared. Hence class A destructor is called at last.

#332 Explained Report Bookmark

What happens when an object is passed by reference?

null

- A
  Destructor is not called
- B
  Destructor is called at end of function
- C
  Destructor is called when function is out of scope
- D
  Destructor is called when called explicitly

Correct Answer :A

Explanation

The destructor is never called in this situation. The concept is that when an object is passed by reference to the function, the constructor is not called, but only the main object will be used. Hence no destructor will be called at end of function.

#333 Explained Report Bookmark

If a function has to be called only by using other member functions of the class, what should be the access specifier used for that function?

null

- **A**
  Private
- **B**
  Protected
- **C**
  Public
- **D**
  Default

Correct Answer :A

Explanation

The function should be made private. In this way, the function will be available to be called only from the class member functions. Hence the function will be secure from the outside world.

#334 Explained Report Bookmark

Which among the following is correct for the code given below?

```
class student

{

     private: student()

     {

     }

     public : student( int x)

     {

          marks =x;

     }

};
```

- **A**
  The object can never be created
- **B**
  The object can be created without parameters
- **C**
  Only the object with only 1 parameter can be created
- **D**
  Only the object with some parameters can be created

Correct Answer :C

Explanation

For creating object without parameters, the default constructor must be defined in public access. But here, only parameterized constructor is public, hence the objects being created with only one parameter will only be allowed.

#335 Explained Report Bookmark

If private members have to be accessed directly from outside the class but the access specifier must not be changed, what should be done?

- A
  Specifier must be changed
- B
  Friend function should be used
- C
  Other public members should be used
- D
  It is not possible

Correct Answer :B

Explanation

For calling the function directly, we can’t use another function because that will be indirect call. Using friend function, we can access the private members directly.

#336 Explained Report Bookmark

Which access specifier is/are most secure during inheritance?

null

- A
  Private

- B
  Default
- C
  Protected
- D
  Private and default

Correct Answer :A

Explanation

The private members are most secure in inheritance. The default members can still be in inherited in special cases, but the private members can’t be accessed in any case.

#337 Explained Report Bookmark

Which option is false for the following code?

```
class A

{

     private : int sum(int x, int y)

     {

          return x+y;

     }

     public: A()
```

```
    {

    }

    A(int x, int y)

    {

        cout&lt;&lt;sum(x,y);

    }

};
```

- **A**
  Constructor can be created with zero argument
- **B**
  Constructor prints sum, if two parameters are passed with object creation
- **C**
  Constructor will give error if float values are passed
- **D**
  Constructor will take 0 as default value of parameters if not passed

Correct Answer :D

Explanation

Constructor is not having any default arguments hence no default value will be given to any parameters. Only integer values must be passed to the constructor if we need the sum as output, otherwise if float values are passed, type mismatch will be shown as error.

#338 Explained Report Bookmark

Which among the following is true for the code given below?

```
class A

{

     int marks;

     public : disp()

     {

          cout&lt;&lt;marks;

     }

}

class B: protected A

{

     char name[20];

}

A a; a.disp();

B b; b.disp();
```

- **A**
  Only object of class A can access disp() function
- **B**
  Only object of class B can access disp() function

- C
  Both instances can access disp() function
- D
  Accessing disp() outside class is not possible

Correct Answer :A

Explanation

The object of class A can access the disp() function. This is because the disp() function is public in definition of class A. But it can’t be accessed from instance of class B because the disp() function is protected in class B, since it was inherited as protected.

#339 Explained Report Bookmark

If the members have to be accessed from anywhere in the program and other packages also, which access specifier should be used?

- A
  Public
- B
  Private
- C
  Protected
- D
  Default

Correct Answer :A

Explanation

The access specifier must be public so as to access the members outside the class and anywhere within the program without using inheritance. This is a rule, predefined for the public members.

#340 Explained Report Bookmark

Which among the following have least security according to the access permissions allowed?

null

- A
  Private
- B
  Default
- C
  Protected
- D
  Public

Correct Answer :D

Explanation

The public members are available to the whole program. This makes the members most vulnerable to accidental changes, which may result in unwanted modification and hence unstable programming.

#341 Explained Report Bookmark

The size of an object or a type can be determined using which operator?

null

- A
  malloc

- B
  sizeof
- C
  malloc
- D
  calloc

Correct Answer :B

Explanation

The sizeof operator gives the size of the object or type.

#342 Explained Report Bookmark

Which is the correct statement about pure virtual functions?

null

- A
) They should be defined inside a base class
- B
  Pure keyword should be used to declare a pure virtual function
- C
  Pure virtual function is implemented in derived classes
- D
  Pure virtual function cannot implemented in derived classes

Correct Answer :C

Explanation

A pure virtual function does not have a definition corresponding to base class. All derived class may or may not have an implementation of a pure virtual function. there is no pure keyword in C++.

#343 Explained Report Bookmark

What is size of generic pointer in C++ (in 32-bit platform)?

null

- **A**
  2
- **B**
  4
- **C**
  8
- **D**
  0

Correct Answer :B

Explanation

Size of any type of pointer is 4 bytes in 32-bit platforms.

#344 Explained Report Bookmark

What will be the output of the following C++ code?

```
#include <iostream>

   using namespace std;

   int main()
```

```
{

    int a[2][4] = {3, 6, 9, 12, 15, 18, 21, 24};

    cout << *(a[1] + 2) << *(*(a + 1) + 2) << 2[1[a]];

    return 0;

}
```

- **A**
  15 18 21
- **B**
  21 21 21
- **C**
  24 24 24
- **D**
  Compile time error

Correct Answer :B

Explanation

a[1][2] means 1 * (4)+2 = 6th element of an array starting from zero.

#345 Explained Report Bookmark

What will be the output of the following C++ code?

```
#include <iostream>

   using namespace std;

   int main()
```

```
{

    int i;

    const char *arr[] = {"C", "C++", "Java", "VBA"};

    const char *(*ptr)[4] = &arr;

    cout << ++(*ptr)[2];

    return 0;

}
```

- **A**
  java
- **B**
  ava
- **C**
  c++
- **D**
  compile time error

Correct Answer :B

Explanation

In this program we are moving the pointer from first position to second position and printing the remaining value.

Output: ava

#346 Explained Report Bookmark

Which members are inherited but are not accessible in any case?

- **A**
  Public
- **B**
  Private
- **C**
  Protected
- **D**
  Both private and protected

Correct Answer :B

Explanation

Private members of a class are inherited to the child class but are not accessible from the child class.

#347 Explained Report Bookmark

Which of the following is a valid class declaration?

null

- **A**
  class A { int x; };
- **B**
  class B { }
- **C**
  public class A { }
- **D**
  object A { int x; };

Correct Answer :A

Explanation

A class declaration terminates with semicolon and starts with class keyword. only option (a) follows these rules therefore class A { int x; }; is correct.

#348 Explained Report Bookmark

The data members and functions of a class in C++ are by default _____

null

- **A**
  protected
- **B**
  private
- **C**
  public
- **D**
  public & protected

Correct Answer :B

Explanation

By default all the data members and member functions of class are private.

#349 Explained Report Bookmark

How many types of templates are there in c++?

null

- A
  1
- B
  2
- C
  3
- D
  4

Correct Answer :B

Explanation

There are two types of templates. They are function template and class template.

#350 Explained Report Bookmark

Which statement is incorrect about virtual function.

null

- A
  They are used to achieve runtime polymorphism
- B
  They are used to hide objects
- C
  Each virtual function declaration starts with the virtual keyword
- D
  All of the mentioned

Correct Answer :B

Explanation

Virtual function are used to achieve runtime polymorphism by calling the right function during runtime. Their declaration starts with a virtual keyword.

#351 Explained Report Bookmark

Which among the following is the correct syntax to access static data member without using member function?

- A className -> staticDataMember;
- B className :: staticDataMember;
- C className : staticDataMember;
- D className . staticDataMember;

Correct Answer :B

Explanation

For accessing the static data members without using the static member functions, the class name can be used. The class name followed by scope resolution, indicating that static data members is member of this class, and then the data member name.

#352 Explained Report Bookmark

Which of the following is universal class for exception handling?

- A Object
- B Errors
- C Exceptions

- DMaths

Correct Answer :C

Explanation

Exceptions is universal class for exception handling

#353 Explained Report Bookmark

Which among the following is true?

- A
  Static member functions can be overloaded
- B
  Static member functions can’t be overloaded
- C
  Static member functions can be overloaded using derived classes
- D
  Static member functions are implicitly overloaded

Correct Answer :B

Explanation

The static member functions can’t be overloaded because the definition must be the same for all the instances of a class. If an overloaded function have many definitions, none of them can be made static.

#354 Explained Report Bookmark

Consider the following code and select the correct option.

```
class student

{

     int marks;

     public : int* fun()

     {

          return &marks;

     }

};

main()

{

    student s;

    int *ptr=c.fun();

    return 0;

}
```

- A This code is good to go
- B This code may result in undesirable conditions
- C This code will generate error
- D This code violates encapsulation

Correct Answer :D

Explanation

This code violates the encapsulation. By this code we can get the address of the private member of the class, hence we can change the value of private member, which is against the rules.

#355 Explained Report Bookmark

Object declared in main() function _____

- A Can be used by any other function
- B Can’t be used by any other function
- C Can be used by main() function of any other program
- D Can be accessed using scope resolution operator

Correct Answer :B

Explanation

The object declared in main() have local scope inside main() function only. It can't be used outside main() function. Scope resolution operator is used to access globally declared variables/objects.

#356 Explained Report Bookmark

If same message is passed to objects of several different classes and all of those can respond in a different way, what is this feature called?

- A Inheritance
- B Overloading
- C Polymorphism
- D Overriding

Correct Answer :C

Explanation

The feature defined in question defines polymorphism features. Here the different objects are capable of responding to the same message in different ways, hence polymorphism.

#357 Explained Report Bookmark

Which among the following can show polymorphism?

- A Overloading ||
- B Overloading +=
- C Overloading <<
- D Overloading &&

Correct Answer :C

Explanation

Only insertion operator can be overloaded among all the given options. And the polymorphism can be illustrated here only if any of these is applicable of being overloaded. Overloading is type of polymorphism.

#358 Explained Report Bookmark

Which of the following is correct for copy constructor?

- A The argument object is passed by reference
- B It can be defined with zero arguments
- C Used when an object is passed by value to a function
- D Used when a function returns an object

Correct Answer :B

Explanation

It can be defined with zero arguments

#359 Explained Report Bookmark

Which among following is correct for initializing the class below?

```
class student{

int marks;

int cgpa;

public: student(int i, int  j){
```

```
marks=I;

cgpa=j

}

};
```

- Astudent s[3]={ s(394, 9); s(394, 9); s(394,9); };
- Bstudent s[2]={ s(394,9), s(222,5) };
- Cstudent s[2]={ s1(392,9), s2(222,5) };
- Dstudent s[2]={ s[392,9], s2[222,5] };

Correct Answer :B

Explanation

It is the way we can initialize the data members for an object array using parameterized constructor. We can do this to pass our own intended values to initialize the object array data.

#360 Explained Report Bookmark

Which is most appropriate comment on following class definition?

```
class Student

{

    int a;

    public : float a;
```

```
};
```

- A Error : same variable name can’t be used twice
- B Error : Public must come first
- C Error : data types are different for same variable
- D It is correct

Correct Answer :A

Explanation

Same variable can’t be defined twice in same scope. Even if the data types are different, variable name must be different. There is no rule like Public member should come first or last.

#361 Explained Report Bookmark

A phone is made up of many components like motherboard, camera, sensors and etc. If the processor represents all the functioning of phone, display shows the display only, and the phone is represented as a whole. Which among the following have highest level of abstraction?

- A Motherboard
- B Display
- C Camera
- D Phone

Correct Answer :D

Explanation

Phone as a whole have the highest level of abstraction. This is because the phone being a single unit represents the whole system. Whereas motherboard, display and camera are its components.

#362 Explained Report Bookmark

Which of the following is not related to OOPS?

- A Class and Object
- B Constructor and Destructor
- C Structure and Union
- D Inheritance and Polymorphism

Correct Answer :C

Explanation

Structure and Union is not related to OOPS

#363 Explained Report Bookmark

How many classes can be defined in a single program?

- A Only 1
- B Only 100
- C Only 999
- D As many as you want

Correct Answer :D

Explanation

As many as you want

#364 Explained Report Bookmark

Which is private member functions access scope?

- A Member functions which can used outside the class
- B Member functions which are accessible in derived class
- C Member functions which can only be used within the class
- D Member functions which can’t be accessed inside the class

Correct Answer :C

Explanation

The member functions can be accessed inside the class only if they are private. The access is scope is limited to ensure the security of the private members and their usage.

#365 Explained Report Bookmark

Which among the following can’t be used to access the members in any way?

- AScope resolution
- BArrow operator
- CSingle colon
- DDot operator

Correct Answer :C

Explanation

The single colon can’t be used in any way in order to access the static members of a class. Other symbols can be used according to the code and need.

#366 Explained Report Bookmark

Which of the following concepts means determining at runtime what method to invoke?

- A
  Data hiding
- B
  Dynamic Typing
- C
  Dynamic binding
- D
  Dynamic loading

Correct Answer :C

Explanation

Dynamic binding also called dynamic dispatch is the process of linking procedure call to a specific sequence of code (method) at run-time. It means that the code to be executed for a specific procedure call is not known until run-time. Dynamic binding is also known as late binding or run-time binding.

#367 Explained Report Bookmark

Which feature in OOP is used to allocate additional function to a predefined operator in any language?

- A Operator Overloading
- B Function Overloading
- C Operator Overriding
- D Function Overriding

Correct Answer : A

Explanation

The feature is operator overloading. There is not a feature named operator overriding specifically. Function overloading and overriding doesn't give addition function to any operator.

#368 Explained Report Bookmark

What will happen if a class is not having any name?

- A
  can have a destructor
- B
  It cannot have a constructor.

- C
  It cannot have a destructor.
- D
  Both A and B.

Correct Answer :D

Explanation

Anonymous class is a class which has no name given to it. C++ supports this feature. These classes cannot have a constructor but can have a destructor. These classes can neither be passed as arguments to functions nor can be used as return values from functions

#369 Explained Report Bookmark

Which of the following factors supports the statement that reusability is a desirable feature of a language?

- A
  It decreases the testing time.
- B
  It lowers the maintenance cost.
- C
  It reduces the compilation time.
- D
  Both A and B.

Correct Answer :D

Explanation

An *inheritance* leads to less development and *maintenance costs*. *With inheritance*, we will be able to override the methods of the base class so that the meaningful implementation of the base class method can be designed in the derived class. An *inheritance* leads to less development and *maintenance costs*.

Encapsulation and abstraction are meant to hide/group data into one element.

Polymorphism is to indicate different tasks performed by a single entity.

#370 Explained Report Bookmark

What is default access specifier for data members or member functions declared within a class without any specifier, in C++?

- A Private
- B Protected
- C Public
- D Depends on compiler

Correct Answer :A

Explanation

The data members and member functions are Private by default in C++ classes, if none of the access specifier is used. It is actually made to increase the privacy of data.

#371 Explained Report Bookmark

Which of the following provides a reuse mechanism?

- A
  Abstraction
- B
  Inheritance
- C
  Dynamic binding
- D
  Encapsulation

Correct Answer :B

Explanation

Inheritance is the OOPS concept that can be used as reuse mechanism

#372 Explained Report Bookmark

Which of the following concepts provides facility of using object of one class inside another class?

- A
  Encapsulation
- B
  Abstraction
- C
  Composition
- D
  Inheritance

Correct Answer :C

Explanation

Composition is one of the fundamental concepts in object-oriented programming. It describes a class that references one or more objects of other classes in instance variables. This allows you to model a has-a association between objects. You can find such relationships quite regularly in the real world

#373 Explained Report Bookmark

Why reference is not the same as a pointer?

- **A**
  A reference can never be null.
- **B**
  A reference once established cannot be changed.
- **C**
  Reference doesn't need an explicit dereferencing mechanism.
- **D**
  All of the above.

Correct Answer :D

Explanation

References cannot have a null value assigned but pointer can

A reference variable can be referenced by pass by value whereas a pointer can be referenced but pass by reference

Reference doesn't need an explicit dereferencing mechanism

Once a reference is created, it cannot be later made to reference another object.

They are entirely different concept. Pointer stores the address whereas Reference is an alias of some variable.

#374 Explained Report Bookmark

Which of the following is the correct way of declaring a function as constant?

- **A**
  const int ShowData(void) { /* statements */ }
- **B**
  int const ShowData(void) { /* statements */ }
- **C**
  int ShowData(void) const { /* statements */ }
- **D**
  Both A and B

Correct Answer :C

Explanation

A "const function", denoted with the keyword `const` after a function declaration,makes it a compiler error for this class function to change a member variable of the class. However, reading of a class variables is okay inside of the function, but writing inside of this function will generate a compiler error

#375 Explained Report Bookmark

Which of the following type of class allows only one object of it to be created?

- A
  Virtual class
- B
  Abstract class
- C
  Singleton class
- D
  Friend class

Correct Answer :C

Explanation

Singleton class allows the programmer to declare only one object of it, If one tries to declare more than one object the program results into error

#376 Explained Report Bookmark

Which is not feature of OOP in general definitions?

- A Code reusability
- B Modularity
- C Duplicate/Redundant data
- D Efficient Code

Correct Answer :C

Explanation

Duplicate/Redundant data is dependent on programmer and hence can’t be guaranteed by OOP. Code reusability is done using inheritance. Modularity is supported by using different code files and classes. Codes are more efficient because of features of OOP.

#377 Explained Report Bookmark

Which feature of OOP is indicated by the following code?

```
class student{  int marks;  };

class topper:public student{  int age;  topper(int age){
this.age=age; } };
```

- A Inheritance
- B Polymorphism
- C Encapsulation and Inheritance
- D inheritance and polymorphism

Correct Answer :C

Explanation

Encapsulation is indicated by use of classes. Inheritance is shown by inheriting the student class into topper class. Polymorphism is not shown here because we have defined the constructor in the topper class but that doesn’t mean that default constructor is overloaded.

#378 Explained Report Bookmark

Which of the following is not type of class?

- AAbstract Class
- BFinal Class
- CStart Class
- DString Class

Correct Answer :C

Explanation

Only 9 types of classes are provided in general, namely, abstract, final, mutable, wrapper, anonymous, input-output, string, system, network. We may further divide the classes into parent class and subclass if inheritance is used.

#379 Explained Report Bookmark

If same message is passed to objects of several different classes and all of those can respond in a different way, what is this feature called?

- AInheritance
- BOverloading
- CPolymorphism
- DOverriding

Correct Answer :C

Explanation

The feature defined in question defines polymorphism features. Here the different objects are capable of responding to the same message in different ways, hence polymorphism.

#380 Explained Report Bookmark

Which is most appropriate comment on following class definition?

```
class Student

{

    int a;

    public : float a;

};
```

- AError : same variable name can’t be used twice
- BError : Public must come first
- CError : data types are different for same variable
- DIt is correct

Correct Answer :A

Explanation

Same variable can’t be defined twice in same scope. Even if the data types are different, variable name must be different. There is no rule like Public member should come first or last.

#381 Explained Report Bookmark

The object can’t be _____

- APassed by reference
- BPassed by value
- CPassed by copy
- DPassed as function

Correct Answer :D

Explanation

Object can’t be passed as function as it is an instance of some class, it’s not a function. Object can be passed by reference, value or copy. There is no term defined as pass as function for objects.

#382 Explained Report Bookmark

Which class can have member functions without their implementation?

- A Default class
- B String class
- C Template class
- D Abstract class

Correct Answer :D

Explanation

Abstract classes can have member functions with no implementation, where the inheriting subclasses must implement those functions.

#383 Explained Report Bookmark

Which of the two features match each other?

- A Inheritance and Encapsulation
- B Encapsulation and Polymorphism
- C Encapsulation and Abstraction
- D Abstraction and Polymorphism

Correct Answer :C

Explanation

Encapsulation and Abstraction are similar features. Encapsulation is actually binding all the properties in a single class or we can say hiding all the features of object inside a class. And Abstraction is hiding unwanted data (for user) and showing only the data required by the user of program.

#384 Explained Report Bookmark

Which among the following can’t be used for polymorphism?

- A Static member functions
- B Member functions overloading
- C Predefined operator overloading
- D Constructor overloading

Correct Answer :A

Explanation

Static member functions are not property of any object. Hence it can’t be considered for overloading/overriding. For polymorphism, function must be property of object, not only of class

#385 Explained Report Bookmark

Which among the following doesn’t come under OOP concept?

- A Platform independent
- B Data binding
- C Message passing

- DData hiding

Correct Answer :A

Explanation

Platform independence is not feature of OOP. C++ supports OOP but it’s not a platform independent language. Platform independence depends on programming language.

#386 Explained Report Bookmark

Which among the following can show polymorphism?

- AOverloading ||
- BOverloading +=
- COverloading <<
- DOverloading &&

Correct Answer :C

Explanation

Only insertion operator can be overloaded among all the given options. And the polymorphism can be illustrated here only if any of these is applicable of being overloaded. Overloading is type of polymorphism.

#387 Explained Report Bookmark

What is the additional feature in classes that was not in structures?

- AData members
- BMember functions

- C Static data allowed
- D Public access specifier

Correct Answer :B

Explanation

Member functions are allowed inside a class but were not present in structure concept. Data members, static data and public access specifiers were present in structures too.

#388 Explained Report Bookmark

Which of the following operators cannot be overloaded?

- A
  >>
- B
  ?:
- C
  . (dot)
- D
  Both (b) and (c)

Correct Answer :D

Explanation

Operators which cannot be overloaded

?: (conditional)

. (member selection)

.* (member selection with pointer-to-member)

:: (scope resolution)

sizeof (object size information)

typeid (object type information)

#389 Explained Report Bookmark

Which of the following specifiers need not be honored by the compiler?

- A register
- B inline
- C static
- D Both (a) & (b)

Correct Answer :D

Explanation

Register and inline are not compiler directives but rather request to the compiler. These requests need not be honored by the compiler.

#390 Explained Report Bookmark

Which among the following can’t be used for polymorphism?

- A Static member functions
- B Member functions overloading
- C Predefined operator overloading
- D Constructor overloading

Correct Answer :A

Explanation

Static member functions are not property of any object. Hence it can’t be considered for overloading/overriding. For polymorphism, function must be property of object, not only of class.

#391 Explained Report Bookmark

At what point of time a variable comes into existence in memory is determined by its

- A
  scope
- B
  storage class
- C
  data type
- D
  all of the above

Correct Answer :B

Explanation

At what point of time a variable comes into existence in memory is determined by its storage class. Storage class is a common feature in object oriented programming languages such as C++. The storage class for a variable determines its visibility and lifetime.

#392 Explained Report Bookmark

Overloading is otherwise called as

- A
  virtual polymorphism
- B
  transient polymorphism
- C
  pseudo polymorphism
- D
  ad-hoc polymorphism

Correct Answer :D

Explanation

When people talk about polymorphism in C++ they usually mean the thing of using a derived class through the base class pointer or reference, which is called subtype polymorphism. ... Ad-hoc polymorphism is also known as overloading. Coercion is also known as (implicit or explicit) casting.

#393 Explained Report Bookmark

Which concept of OOP is false for C++?

- A
  Code can be written without using classes
- B
  Code must contain at least one class

- C
  A class may or may not have member functions
- D
  the object can only be declared in a code if its class is defined

Correct Answer :B

Explanation

In C++, it’s not necessary to use classes, and hence codes can be written without using OOP concept. Classes may or may not contain member functions, so it’s not a necessary condition in C++. And, an object can only be declared in a code if its class is defined/included via header file.

#394 Explained Report Bookmark

Which feature allows open recursion, among the following?

- AUse of this pointer
- BUse of pointers
- CUse of pass by value
- DUse of parameterized constructor

Correct Answer :A

Explanation

Use of this pointer allows an object to call data and methods of itself whenever needed. This helps us call the members of an object recursively, and differentiate the variables of different scopes.

#395 Explained Report Bookmark

Consider the declarations

char a;

const char aa = 'h';

char *na;

const char *naa;

Which of the following statements is/are illegal?

Statement I: aa = a;

Statement II: na = &a;

Statement III: na = &aa;

- **A**
  Only I and II
- **B**
  Only II and III
- **C**
  Only I and III
- **D**
  All the three statements are illegal

Correct Answer :C

Explanation

- Why first statement is wrong is we can't change the vlaues of the constant variable,if we true that it is showing error.
- In second statement we just assigning address of the a variable to pointer variable na it is acceptable, to store .
- In 3rd statement we doing the same as 2nd statement but here we assigning constant variable address(&aa) to normal char pointer variable but this will give error as cannot convert const char * to char *.

#396 Explained Report Bookmark

Use of preprocessor directive in OOP

- **A**
  for conditional compilation
- **B**
  for macro expansion
- **C**
  error and warning reporting
- **D**
  all the above

Correct Answer :D

Explanation

Preprocessor is a program that processes an input to produce an output which is used as another programs input. The preprocessor directives are generally invoked by the compiler to process before compilation. The preprocessor directive are capable of performing simple textual substitutions, macro expansion, conditional compilation, warning and error reporting. It begins with special character # followed by directive name. Preprocessor directive statement does not end with a semi colon. Few examples of preprocessor directives are: #include, #define, #undef, #if,

#elif, #else, #endif, #line etc. It is used in various OOP languages like C, C++, C#.

#397 Explained Report Bookmark

Which statements are true about an abstract class

- **A**
  Abstract class has at least one pure virtual function.
- **B**
  Pointer for an abstract class can be created
- **C**
  Object of an abstract class cannot be created.
- **D**
  All are correct.

Correct Answer :D

Explanation

Abstract classes are classes that contain one or more abstract methods.

An abstract method is a method that is declared, but contains no implementation.

Abstract classes may not be instantiated, and require subclasses to provide implementations for the abstract methods.

we can create a pointer to an abstract class, which could be actually pointing to the objects of its derived classes. In this way, a container of base class pointers can be created which can also store derived class objects

An abstract class contains at least one pure virtual function. You declare a pure virtual function by using a pure specifier (= 0) in the declaration of a virtual member function in the class declaration.

#398 Explained Report Bookmark

Not using virtual destructor feature in a C++ object oriented programing can cause

- A Memory leak
- B An Issue in creating object of the class
- C An issue in calling base class destructor
- D Nothing

Correct Answer :A

Explanation

Virtual destructor is used to maintain the hierarchy of destructor calls for polymorphic classes in inheritance. If we don’t use it then it may cause resource leak or memory leak.

#399 Explained Report Bookmark

A copy constructor is called

- A when an object is returned by value
- B when an object is passed by value as an argument
- C when compiler generates a temporary object
- D all the above

Correct Answer :D

Explanation

Copy constructor is a type of constructor which is used for creating a new object that is an exact copy of the existing object. In general compiler creates a copy constructor for each and every class on its own but in special case(when object has pointers involved) copy constructor is created by the programmer. It is called in all the above cases.

#400 Explained Report Bookmark

Polymorphism types is/are

- A Compile time
- B Run time
- C Both
- D None

Correct Answer :C

Explanation

Two types of Polymorphism are available in C++ object oriented programming i.e. compile time and run time polymorphism. Also, known as early binding and late binding respectively. Compiler time polymorphism features in C++ language are function overloading, constructor and operator overloading etc. and run time polymorphism is function overriding in inheritance relationship.

#401 Explained Report Bookmark

Operators can be overloaded in C++ is/are

- A
  New
- B
  Delete

- C
  ++
- D
  All can be overloaded

Correct Answer :D

Explanation

List of operators that can be overloaded are:

```
+     -     *     /       %          ^

&     |     ~     !,          =

      =        ++          --

      ==      !=        &&          ||

+=     -=     /=     %=        ^=          &=

|=     *=     =        []          ()

->     ->*     new     new []       delete     delete []
```

#402 Explained Report Bookmark

Correct way of creating an object of a class called Car is

- ACar obj;
- BCar *obj = new Car();
- COnly B
- DA & B both

Correct Answer :D

Explanation

Both Car obj; and Car *obj = new Car() are valid way to create an object of the class.

#403 Explained Report Bookmark

Which of the following cannot be overloaded in C++?

- A Increment operator
- B Constructor
- C Destructor
- D New and delete operator

Correct Answer :C

Explanation

Destructor of a class cannot be overloaded in C++ programming. Increment operator, constructor and new and delete can be overloaded.

#404 Explained Report Bookmark

In C++ programming, cout is a/an

- A
  Function
- B
  Operator
- C
  Object

- D
  macro

Correct Answer :C

Explanation

The cout object in C++ is an object of class ostream. It is used to display the output to the standard output device i.e. monitor. It is associated with the standard C output stream stdout.

#405 Explained Report Bookmark

Which function cannot be overloaded in C++

- AConstructor
- BClass destructor
- CBoth a & b
- Dnull

Correct Answer :B

Explanation

None

#406 Explained Report Bookmark

For a method to be an interface between the outside world and a class, it has to be declared

- A
  private

- **B**
  protected
- **C**
  public
- **D**
  external

Correct Answer :C

Explanation

C++ has no built-in concepts of interfaces. You can implement it using abstract classes which contains only pure virtual functions. Since it allows multiple inheritance, you can inherit this class to create another class which will then contain this interface (I mean, object interface :)) in it.

The public interface of a class are its public properties (variables or fields you can read the values of or assign to) and methods (functions you can call). So, the assignment is to create something that is not a subclass of LinkedList . Creating a subclass would give you access to protected methods

#407 Explained Report Bookmark

A pointer pointing to a variable that is not initialized is called ____

- **A**
  Void Pointer
- **B**
  Null Pointer
- **C**
  Empty Pointer
- **D**
  Wild Pointer

Correct Answer :B

Explanation

just like normal variables, pointers are not initialized when they are instantiated. ... A null value is a special value that means the pointer is not pointing at anything. A pointer holding a null value is called a null pointer.

#408 Explained Report Bookmark

A class whos objects can not be created is known as _____

- **A**
  Base Class
- **B**
  Derived Class
- **C**
  Super Class
- **D**
  Abstract Class

Correct Answer :D

Explanation

Instance of abstract class can't be created as it will not have any constructor of its own, hence while creating an instance of class, it can't initialize the object members.

#409 Explained Report Bookmark

Which class allows only one object to be created.

- **A**
  Virtual class

- B
  Abstruct Class
- C
  Sigleton Class
- D
  Friend class

Correct Answer :C

Explanation

Singleton class allows the programmer to declare only one object of it, If one tries to declare more than one object the program results into error

#410 Explained Report Bookmark

A class having no name (Anonymous class) in c++ , which of the statement is true

- A
  cannot be passed as an argument
- B
  cannot have a constructor
- C
  cannot have a destructor
- D
  Both Option 1 & 2

Correct Answer :D

Explanation

Anonymous class is a class which has no name given to it. C++ supports this feature.

These classes cannot have a constructor but can have a destructor.

These classes can neither be passed as arguments to functions nor can be used as return values from functions

#411 Explained Report Bookmark

The following program

```
void main ( )

{

int x=10;

int &p=x;

cout << &p <<  &x;

}
```

- A prints 10 and the address of x
- B prints the address of p twice
- C prints the address of x twice
- D Both (b) & (c)

Correct Answer :D

Explanation

int &p=x aliases p to x. This means they refer to the same memory location. So, the address of x will be same as that of p.

#412 Explained Report Bookmark

An exceptio in C++ can be generated using which keywords.

- **A**
  thrown
- **B**
  threw
- **C**
  throw
- **D**
  throws

Correct Answer :C

Explanation

C++ exception handling is built upon three keywords: try, catch, and throw. throw − A program throws an exception when a problem shows up. This is done using a throw keyword. catch − A program catches an exception with an exception handler at the place in a program where you want to handle the problem.

#413 Explained Report Bookmark

A function abc is defined as

```
void abc (int x=0, int y=0)
{
cout << x << y;
}
```

Which of the following function calls is/are illegal? (Assume h, g are declared as integers)

- Aabc () ;
- Babc (h) ;
- Cabc (g, h) ;
- DNone of the above

Correct Answer :D

Explanation

Both the arguments are optional. All the calls are legal.

#414 Explained Report Bookmark

The statements

int a = 5;

cout << "FIRST" << (a <<2) << "SECOND";

outputs

- AFIRST52SECOND
- BFIRST20SECOND
- CSECOND25FIRST
- Dan error message

Correct Answer :B

Explanation

The symbol << has a context sensitive meaning. The << in (a << 2) means shifting a by 2 bits to the left, which is nothing but multiplying it by 4. So, a < <2 will be 5 x 4 = 20 and hence the output will be FIRST20SECOND.

#415 Explained Report Bookmark

Default constructor has ____ arguments.

- **A**
  No argument
- **B**
  One Argument
- **C**
  Two Argument
- **D**
  None of these

Correct Answer :A

Explanation

A default constructor is a constructor that either has no parameters, or if it has parameters, all the parameters have default values. If no user-defined constructor exists for a class A and one is needed, the compiler implicitly declares a default parameterless constructor A::A() .

#416 Explained Report Bookmark

Which among the following best defines single level inheritance?

- AA class inheriting a derived class
- BA class inheriting a base class
- CA class inheriting a nested class
- DA class which gets inherited by 2 classes

Correct Answer :B

Explanation

A class inheriting a base class defines single level inheritance. Inheriting an already derived class makes it multilevel inheritance. And if base class is inherited by 2 other classes, it is multiple inheritance.

#417 Explained Report Bookmark

Which of the following is an invalid visibility label while inheriting a class?

- A
  public
- B
  private
- C
  protected
- D
  friend

Correct Answer :D

Explanation

In C++, there are three access specifiers: public - members are accessible from outside the class. private - members cannot be accessed (or viewed) from outside the class. protected - members cannot be accessed from outside the class, however, they can be accessed in inherited classes.

#418 Explained Report Bookmark

Which of the following is correct about class and structure?

- **A**
  class can have member functions while structure cannot.
- **B**
  class data members are public by default while that of structure are private.
- **C**
  Pointer to structure or classes cannot be declared.
- **D**
  class data members are private by default while that of structure are public by default.

Correct Answer :D

Explanation

In C++, a structure is a class defined with the struct keyword. Its members and base classes are public by default. A class defined with the class keyword has private members and base classes by default. This is the only difference between structs and classes in C++.

#419 Explained Report Bookmark

Which of the following statement is correct?

- **A**
  Class is an instance of object.
- **B**
  Object is an instance of a class.

- C
  Class is an instance of data type.
- D
  Object is an instance of data type.

Correct Answer :B

Explanation

An object is an instance of a class, and may be called a class instance or class object; instantiation is then also known as construction. Not all classes can be instantiated – abstract classes cannot be instantiated, while classes that can be instantiated are called concrete classes.

#420 Explained Report Bookmark

Which one of the following is correct about the statements given below?

1. All function calls are resolved at compile-time in Procedure Oriented Programming.
2. All function calls are resolved at compile-time in OOPS.

- A
  Only II is correct.
- B
  Both I and II are correct.
- C
  Only I is correct.
- D
  Both I and II are incorrect.

Correct Answer :C

Explanation

The binding of functional call and choosing the correct function declaration is done by compiler at the compile time.It provides fast execution because the method that needs to be executed is known early at the compile time.

#421 Explained Report Bookmark

If same message is passed to objects of several different classes and all of those can respond in a different way, what is this feature called?

- A Inheritance
- B Overloading
- C Polymorphism
- D Overriding

Correct Answer :C

Explanation

The feature defined in question defines polymorphism features. Here the different objects are capable of responding to the same message in different ways, hence polymorphism.

#422 Explained Report Bookmark

Which of the following concepts means determining at runtime what method to invoke?

- A
  Data hiding
- B
  early Binding
- C
  Late binding
- D
  late loading

Correct Answer :C

Explanation

Dynamic binding also called dynamic dispatch is the process of linking procedure call to a specific sequence of code (method) at run-time. It means that the code to be executed for a specific procedure call is not known until run-time. Dynamic binding is also known as late binding or run-time binding.

#423 Explained Report Bookmark

Which among the following is correct?

- **A**
  class student{ public: int student(){} };
- **B**
  class student{ public: void student (){} };
- **C**
  class student{ public: student{}{} };
- **D**
  class student{ public: student(){} };

Correct Answer :D

Explanation

The constructors must not have any return type. Also, the body may or may not contain any body. Defining default constructor is optional, if you are not using any other constructor.

#424 Explained Report Bookmark

Which among the following can show polymorphism?

- AOverloading ||
- BOverloading +=
- COverloading <<
- DOverloading &&

Correct Answer :C

Explanation

Only insertion operator can be overloaded among all the given options. And the polymorphism can be illustrated here only if any of these is applicable of being overloaded. Overloading is type of polymorphism.

#425 Explained Report Bookmark

Which of the following is not a type of constructor?

- A
  Copy constructor
- B
  Friend constructor
- C
  Default constructor
- D
  Parameterized constructor

Correct Answer :B

Explanation

Constructors are of three types: Default Constructor. Parametrized Constructor. Copy COnstructor.

#426 Explained Report Bookmark

Which of the following is associated with objects?

- **A**
  State
- **B**
  Behaviour
- **C**
  Identity
- **D**
  All of the above

Correct Answer :D

Explanation

An entity that has state and behaviour is known as an object e.g. chair, bike, marker, pen, table, car etc. It can be physical or logical (tangible and intangible). The example of the intangible object is the banking system.

An object has three characteristics:

State: represents data (value) of an object.

Behaviour: represents the behaviour (functionality) of an object such as deposit, withdraw etc.

Identity: Object identity is typically implemented via a unique ID. The value of the ID is not visible to the external user. But, it is used internally by the JVM to identify each object uniquely.

#427 Explained Report Bookmark

An Object is a/an _____ of a class.

- A type
- B prototype
- C instance
- D object

Correct Answer :C

Explanation

An object is said to be an instance of its class. It is of the type defined by the class.

#428 Explained Report Bookmark

The practice of separating the user from the true inner workings of an application through well-known interfaces is known as ____

- A
  Polymorphism
- B
  Inheritance
- C
  Abstraction
- D
  Encapsulation

Correct Answer :D

Explanation

In object-oriented prorgramming , encapsulation is a concept of wrapping up or binding up the data members and methods in a single module

#429 Explained Report Bookmark

Which one of the following options is correct about the statement given below

statement :

The compiler checks the type of reference in the object and not the type of object.

- **A** Inheritance
- **B** Polymorphism
- **C** Abstraction
- **D** Encapsulation

Correct Answer :B

Explanation

Polymorphism means one name and multiple implementations. We have two types of polymorphism early binding and late binding in early binding we are able to know at compile time but in late binding we will know at run time what actually the function will be called.

#430 Explained Report Bookmark

Which of the following is the correct class of the object cout?

- **A** iostream
- **B** istream
- **C** ostream

- **D**
  ifstream

Correct Answer :C

Explanation

C++ Standard Output Stream (cout)? std::cout is an object of class ostream that represents the standard output stream oriented to narrow characters (of type char). It corresponds to the C stream stdout. The standard output stream is the default destination of characters determined by the environment

#431 Explained Report Bookmark

Which one of the following are essential features of object oriented language?

A. Abstraction and encapsulation

B. Strictly-typed

C. Type-safe property coupled with sub-type rule

D. Polymorphism in the presence of inheritance

- **A**
  A and B only
- **B**
  A, D and B only
- **C**
  A and D only
- **D**
  A, C and D only

Correct Answer :C

Explanation

Object Oriented Programming (OPP) is a programming paradigm. The language is object oriented as it use objects. Objects are the data structures that contain data fields and methods together with their interactions.

The main features of the Programming techniques are

1. data abstraction 2. encapsulation 3. modularity 4. polymorphism 5. inheritance

#432 Explained Report Bookmark

Which inheritance type is used in the class given below?

class A : public X, public Y

{}

- **A**
  Multilevel inheritance
- **B**
  Multiple inheritance
- **C**
  Hybrid inheritance
- **D**
  Hierarchical Inheritance

Correct Answer :B

Explanation

Multiple inheritance is a feature of some object-oriented computer programming languages in which an object or class can inherit characteristics and features from more than one parent object or parent class.

#433 Explained Report Bookmark

Which of the following advanced OOP features is/are not supported by PHP?

1. Method overloading

2. Multiple Inheritance

3. Namespaces

4. Object Cloning

- A All of the mentioned
- B None of the mentioned
- C 1 and 2
- D 3 and 4

Correct Answer :C

Explanation

The advanced OOP features are: Object cloning, Inheritance, Interfaces, Abstract classes, and Namespaces.

#434 Explained Report Bookmark

The feature in object-oriented programming that allows the same operation to be carried out differently, depending on the object, is:

- A
  Inheritance
- B
  Polymorphism
- C
  Overfunctioning
- D
  Overriding

Correct Answer :B

Explanation

Polymorphism is the ability of an object to take on many forms. The most common use of polymorphismin OOP occurs when a parent class reference is used to refer to a child class object. Any Java object that can pass more than one IS-A test is considered to be polymorphic

#435 Explained Report Bookmark

Which one of the following is not a valid class name?

- AShopProduct
- BShopproduct
- CShopproduct1
- D1shopproduct

Correct Answer :D

Explanation

You declare a class with the class keyword and an arbitrary class name. Class names can be any combination of numbers and letters, although they must not begin with a number.

#436 Explained Report Bookmark

Features not available in C++ object oriented programming is

- A Virtual destructor
- B Virtual constructor
- C Virtual function
- D All

Correct Answer :B

Explanation

There is no concept of virtual constructor available in C++. However, virtual destructor is available in C++ language to maintain the destructor call from derived to base class. In polymorphic classes, if we don’t use virtual destructor in base class then the derived class destructor will not be called that may cause resource leaks.

#437 Explained Report Bookmark

IS A relationship in C++ is

- A
  Inheritance
- B
  Encapsulation

- C
  Composition
- D
  None

Correct Answer :A

Explanation

In OOP, IS-A relationship is completely inheritance. This means, that the child class is a type of parent class. ... A HAS-A relationship is dynamic (run time) binding while inheritance is a static (compile time) binding. If you just want to reuse the code and you know that the two are not of same kind use composition

#438 Explained Report Bookmark

Encapsulation helps

- Ainformation hiding
- Bin providing low coupling
- Cin providing high cohesion
- DAll the above

Correct Answer :D

Explanation

Encapsulation works on providing interactions through function calling only. Using keyword private or protected stops the use of data members outside class. The data thus remains accessible to functions inside class. Encapsulation is also called information hiding as it restricts the use of data inside class. Coupling is the degree of interdependence between different

blocks(modules) of program code. Low coupling helps in keeping data safe inside the class. Coupling contrasts cohesion as with high cohesion comes reliability and reusability in a code.

#439 Explained Report Bookmark

If you want to write multiple functions in a class with same name, then what C++ feature will you use?

- A Function overriding
- B Encapsulation
- C Function overloading
- D None

Correct Answer :C

Explanation

Compile time polymorphism feature is used i.e. function overloading in C++. Function overloading means, in a class, multiple functions with same name can be written with different signatures, return types etc.

#440 Explained Report Bookmark

Break statement in switch case

- A prevents from fallthrough
- B causes an exit from innermost loop
- C both a and b
- D none

Correct Answer :C

Explanation

Break statement is an important statement which alters the normal flow of a program. It helps in exiting the switch case block . It is frequently used to terminate the processing of a particular case within the switch statement. Control passes to the statement that follows the terminated statement. It also prevents fallthrough which occurs in the absence of break in the switch case. Fallthrough is a situation that leads to execution of all the cases which is possible in absence of break statement

#441 Explained Report Bookmark

Which C++ oops feature is related to re-usability?

- A Encapsulation
- B Inheritance
- C Abstraction
- D None

Correct Answer :B

Explanation

nheritance feature is used for concept of code re-usability as in inheritance a class can inherit properties and functions of existing well written class. Abstraction : Provide only necessary information to client code. Encapsulation: Hide complexity. e.g. by wrapping private class data members by functions. By making internal function of a class private and using interfaces etc.

#442 Explained Report Bookmark

If I want to have common functions in a class and want to defer implementations of some other functions to derived classes, then we need to use

- AAn interface
- BAn abstract class
- CA friend class
- DA static class

Correct Answer :B

Explanation

In C++ object oriented programming, abstract class is used for the same, in which we have common or say generalized function in abstract base class and also may have pure virtual function in this class that forces derived classes to implement it.

#443 Explained Report Bookmark

In a class, encapsulating an object of another class is called

- AComposition
- BInheritance
- CEncapsulation
- DNone

Correct Answer :A

Explanation

n simple word, if a class contains an object of another class as a data member, then it is known as composition. For example, Class Y, have a class X’s object as data member. Means, Y is composed of X. for example,we can take that a house is composed of windows, door and bricks etc. So, class House will look like below class Door { }; class

Windows { }; class Bricks { }; class House{ Door _d; Windows _w; Bricks _b; public: void showHouse() { } };

#444 Explained Report Bookmark

The OOPs concept in C++, exposing only necessary information to users or clients is known as

- A
  Abstraction
- B
  Encapsulation
- C
  Data hiding
- D
  Hiding complexity

Correct Answer : A

Explanation

The OOPs concept in C++, exposing only necessary information to users or clients is known as Abstraction.

Abstraction means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation We can implement Abstraction in C++ using classes

#445 Explained Report Bookmark

The keyword 'this' is used

- AAs reference to current object
- BExplicit constructor invocation

- C In open recursion
- D All the above

Correct Answer :D

Explanation

In object oriented programming the keyword 'this' is used to refer to objects, classes or any other entity that is part of currently running code. Different languages use it in different ways. In some languages, 'this' is the only way to access data and methods in current object. The concept, however, is similar in all languages using 'this'.It is usually a fixed reference or pointer which refers to current object. In some languages it is used explicitly while others use lexical scoping(range of functionality) to use it implicitly to make symbols within their class visible. The dispatch semantics of 'this' that method calls on 'this'are dynamically dispatched which means that these methods can be overriden by derived classes or objects.

#446 Explained Report Bookmark

Which of the property of a object encompasses all of the (usually static) properties of the object plus the current (usually dynamic) values of each of these.

- A
  Identity
- B
  Behaviour
- C
  State
- D
  All of the above

Correct Answer :C

Explanation

The state of an object encompasses all of the (usually static) properties of the object plus the current (usually dynamic) values of each of these properties.The fact that every object has state implies that every object takes up some amount of space, be it in the physical world or in computer memory

#447 Explained Report Bookmark

Abstraction and encapsulation are fundamental principles that underlie the object oriented approach to software development. What can you say about the following two statements ?

I. Abstraction allows us to focus on what something does without considering the complexities of how it works.

II. Encapsulation allows us to consider complex ideas while ignoring irrelevant detail that would confuse us.

- A Neither I nor II is correct
- B Neither I nor II is correct
- C Only II is correct
- D Only I is correct

Correct Answer :A

Explanation

Encapsulation allows us to focus on what something does without considering the complexities of how it works. Abstraction allows us to consider complex ideas while ignoring irrelevant detail that would confuse us. So, option (A) is correct.

#448 Explained Report Bookmark

Consider the following two statements: (a)A publicly derived class is a subtype of its base class. (b)Inheritance provides for code reuse.

- A Both the statements (a) and (b) are correct
- B Neither of the statements (a) and (b) are correct
- C Statement (a) is correct and (b) is incorrect
- D Statement (a) is incorrect and (b) is correct.

Correct Answer : A

Explanation

A publicly derived class is a subtype of its base class. Inheritance provides for code reuse. So, option (A) is correct.

#449 Explained Report Bookmark

Which of the following statements regarding the features of the object-oriented approach to databases are true?

(a)The ability to develop more realistic models of the real world.

(b)The ability to represent the world in a non-geometric way.

(c)The ability to develop databases using natural language approaches.

(d)The need to split objects into their component parts.

(e)The ability to develop database models based on location rather than state and behaviour. Codes:

- A (a), (b) and (c)
- B (b), (c) and (d)

- C(a), (d) and (e)
- D(c), (d) and (e)

Correct Answer :A

Explanation

Features of the object-oriented approach to databases: The ability to develop more realistic models of the real world. The ability to represent the world in a non-geometric way. The ability to develop databases using natural language approaches. So, option (A) is correct.

#450 Explained Report Bookmark

Converting a primitive type data into its corresponding wrapper class object instance is called

- ABoxing
- BWrapping
- CInstantiation
- DAutoboxing

Correct Answer :D

Explanation

Converting a primitive type data into its corresponding wrapper class object instance is called Autoboxing. Instantiation is creation or a real instance or a particular realization of an abstraction or template. Wrapping encapsulates and hides the underlying complexity of another entity. Boxing is the process of converting a value type to the type object or to any interface type implemented by this value type So, option (D) is correct.

#451 Explained Report Bookmark

Choose the Object-oriented programming language from below.

- **A**
  C++
- **B**
  Small talk
- **C**
  Simula
- **D**
  all

Correct Answer :D

Explanation

C++ is called object-oriented programming (OOP) language because C++ language views a problem in terms of objects involved rather than the procedure for doing it

Smalltalk is an object-oriented, dynamically typed reflective programming language.

Simula is considered the first *object-oriented programming* language.

#452 Explained Report Bookmark

When a class serves as base class for many derived classes, the situation is called:

- **A**
  polymorphism

- **B**
  hierarchical inheritance
- **C**
  hybrid inheritance
- **D**
  multipath inheritance

Correct Answer :B

Explanation

Hierarchical inheritance is a kind of inheritance where more than one class is inherited from a single parent or base class

#453 Explained Report Bookmark

Multiple inheritance leaves room for a derived class to have _____ members.

- **A**
  dynamic
- **B**
  private
- **C**
  public
- **D**
  ambiguous

Correct Answer :D

Explanation

The ambiguity that arises when using multiple inheritance refers to a derived class having more than one parent class that defines property[s] and/or method[s] with the same name. For example, if 'C' inherits from both

'A' and 'B' and classes 'A' and 'B', both define a property named x and a function named getx().

#454 Explained Report Bookmark

Which of the following is an abstract data type?

- A
  Class
- B
  Int
- C
  String
- D
  Double

Correct Answer :A

Explanation

Modern object-oriented languages, such as C++ and Java, support a form of abstract data types. When a class is used as a type, it is an abstract type that refers to a hidden representation. In this model an ADT is typically implemented as a class, and each instance of the ADT is usually an object of that class.

#455 Explained Report Bookmark

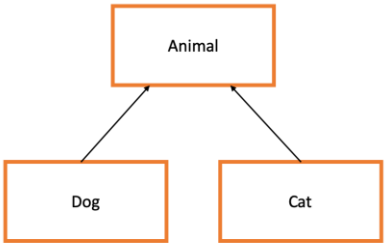
For Cat and Animal class, correct way of inheritance is

- A
  class Cat: public Animal
- B
  class Animal: public Cat
- C
  Both are correct way

- D
  None is correct way

Correct Answer :A

Explanation



In words, this means that a Dog *is an* Animal and a cat *is an* Animal. Let's implement this by modifying our class declarations

```
class Dog { → class Dog : public Animal {
```

```
class Cat { → class Cat : public Animal {
```

The *public* keyword in that syntax is called an access specifier. Ignore it for now - we'll discuss it later. Also, you probably need to re-organize your code so that Animal is declared before Dog and Cat,

#456 Explained Report Bookmark

If I want to have common functions in a class and want to defer implementations of some other functions to derived classes, then we need to use

- AAn interface

- BAn abstract class
- CA friend class
- DA static class

Correct Answer :B

Explanation

In object oriented programming, abstract class is used for the same, in which we have common or say generalized function in abstract base class and also may have pure virtual function in this class that forces derived classes to implement it.

#457 Explained Report Bookmark

Which public member of a base class cannot be inherited?

- AConstructor
- BDestructor
- CBoth A & B
- DOnly B

Correct Answer :C

Explanation

In C++ constructor and destructor both cannot be inherited to child class.

#458 Explained Report Bookmark

The following c++ code results in

#include "iostream.h"

```
void main (void)

{

cout << (int j=8) << (int k=9);

}
```

- **A**
  compilation error
- **B**
  run time error
- **C**
  link time error
- **D**
  none of the above

Correct Answer :A

Explanation

‘::main’ must return ‘int’ and expected primary-expression before ‘int’ for the cout statement. An expression enclosed in parentheses is a primary expression. Its type and value are identical to the type and value of the unparenthesized expression. It's an l-value if the unparenthesized expression is an l-value.

#459 Explained Report Bookmark

Compile time polymorphism in C++ language are

- **A**
  Operator overloading
- **B**
  Function overloading

- C
  Function overriding
- D
  A & B

Correct Answer :D

Explanation

Compile time polymorphism: This type of polymorphism is achieved by function overloading or operator overloading. Function Overloading: When there are multiple functions with same name but different parameters then these functions are said to be overloaded.

#460 Explained Report Bookmark

Run time polymorphism in C++ Program is

- A
  New and delete operator overloading
- B
  ++ and – – operator overloading
- C
  :: operator overloading
- D
  None

Correct Answer :D

Explanation

Runtime polymorphism is also known as dynamic polymorphism or late binding. In runtime polymorphism, the function call is resolved at run time. In contrast, to compile time or static polymorphism, the compiler deduces

the object at run time and then decides which function call to bind to the object

#461 Explained Report Bookmark

Which of the following concepts is used to implement late binding?

- A
  Virtual function
- B
  Operator functions
- C
  Const function
- D
  Static function

Correct Answer :A

Explanation

Virtual functions are used to implement the concept of late binding i.e. binding actual functions to their calls.

#462 Explained Report Bookmark

Consider the following program segment.

static char X [3] = "1234" ;

cout << X ;

A complete c++ program with these two statements

- Aprints 12 3 4
- Bprints 12 3
- Cprints 1234 followed by some junk

- D will give a compilation error

Correct Answer :D

Explanation

c++ forbids initialization with strings, whose length is more than the size of the array. C compiler permits.

#463 Explained Report Bookmark

Choose the correct statements.

- A
  A destructor is not inherited
- B
  A constructor cannot be called explicitly
- C
  A constructor is not inherited
- D
  All of these

Correct Answer :D

Explanation

Destructors are not inherited. If a class doesn't define one, the compiler generates one. ... But if a class has members with destructors, the generated destructor calls destructors for those members before calling the base class' destructor. That's something that an inherited function would not do

you cannot call a constructor from a method. The only place from which you can invoke constructors using “this()” or, “super()” is the first line of another constructor. If you try to invoke constructors explicitly elsewhere, a compile time error will be generated

A subclass inherits all the members (fields, methods, and nested classes) from its superclass. Constructors are not members, so they are not inherited by subclasses, but the constructor of the superclass can be invoked from the subclass.

#464 Explained Report Bookmark

A constructor is called whenever

- **A**
  an object is declared
- **B**
  an object is used
- **C**
  a class is declared
- **D**
  a class is used

Correct Answer :A

Explanation

The constructors are special type of functions which are called whenever an object is created. This is to initialize the data members of the class. The constructor allocates memory space for all the data members. ... Explanation: The constructors must contain only the class name

#465 Explained Report Bookmark

Following keyword is used before a function in a base class to be overridden in derived class in C++

- **A**
  override
- **B**
  virtual
- **C**
  void
- **D**
  none

Correct Answer :B

Explanation

The virtual keyword can be used when declaring overriding functions in a derived class, but it is unnecessary; overrides of virtual functions are always virtual. Virtual functions in a base class must be defined unless they are declared using the pure-specifier.

#466 Explained Report Bookmark

Which of the following remarks about the differences between constructors and destructors are correct ?

- **A**
  Constructors can take arguments but destructors cannot.
- **B**
  Constructors can be overloaded but destructors cannot be overloaded.
- **C**
  Destructors can take arguments but constructors cannot.
- **D**
  Both (a) and (b)

Correct Answer :D

Explanation

Constructors can take arguments but destructors cannot

Constructors can be overloaded but destructors cannot be overloaded.

#467 Explained Report Bookmark

Delaration a pointer more than once may cause __

- **A**
  Error
- **B**
  Abort
- **C**
  Trap
- **D**
  Null

Correct Answer :C

Explanation

A trap is when the program detects an error condition and takes some action accordingly

#468 Explained Report Bookmark

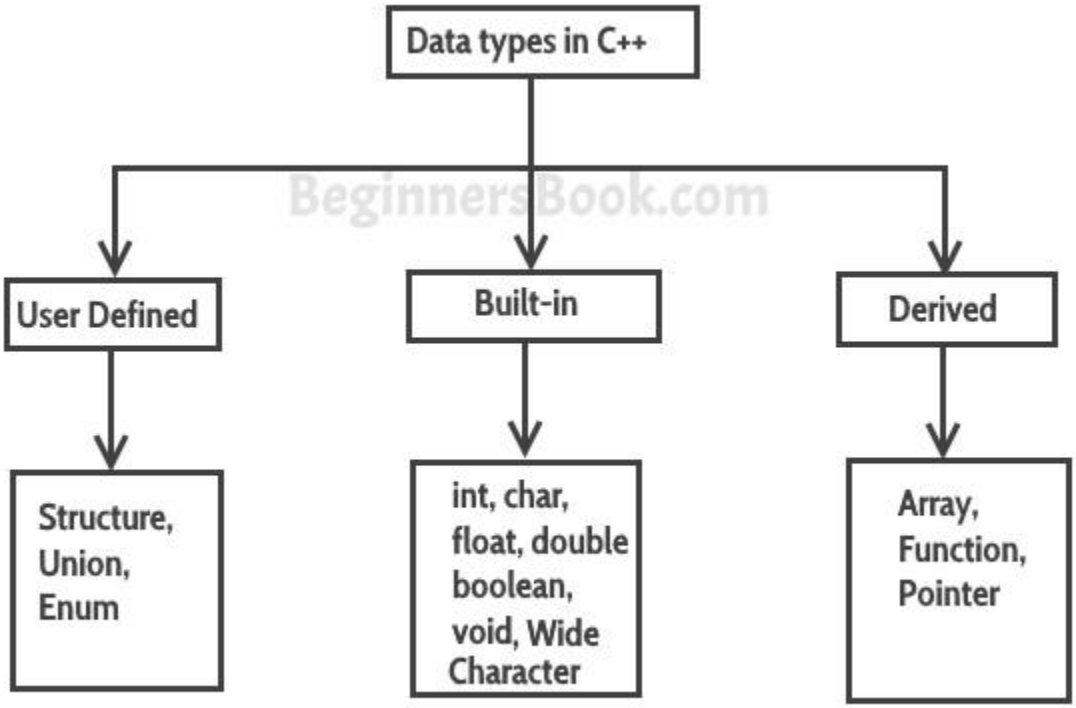
Which one is not a correct variable type in C++?

- **A**
  float

- B
  real
- C
  int
- D
  double

Correct Answer :B

Explanation



#469 Explained Report Bookmark

Which operation is used as Logical 'AND'

- A
  Operator-&

- B
  Operator-||
- C
  Operator +
- D
  Operator-&&

Correct Answer :D

Explanation

The AND logic operation returns true only if either of its inputs are true. If either of the inputs is false, the output is also false. In computer programming, the AND operation is usually written as && (two ampersands).

#470 Explained Report Bookmark

A C++ code line ends with ___

- A
  A Semicolon (;)
- B
  A Fullstop(.)
- C
  A Comma (,)
- D
  A Slash (/)

Correct Answer :A

Explanation

All C++ statements must end with a semicolon character. One of the most common syntax errors in C++ is forgetting to end a statement with a semicolon. You may have noticed that not all the lines of this program perform actions when the code is executed.

#471 Explained Report Bookmark

____ function is used to allocate space for array in memory

.

- **A**
  malloc()
- **B**
  realloc()
- **C**
  alloc()
- **D**
  calloc()

Correct Answer :D

Explanation

The <stdlib.h> library has functions responsible for Dynamic Memory Management.

Function	Purpose
malloc()	Allocates the memory of requested size and returns the pointer to the first byte of allocated space.

calloc()	Allocates the space for elements of an array. Initializes the elements to zero and returns a pointer to the memory.
realloc()	It is used to modify the size of previously allocated memory space.
Free()	Frees or empties the previously allocated memory space.

#472 Explained Report Bookmark

The following program fragment

```
int i=10;
void main ( )
{
int i=20;
{
int i=30.;
cout << i << :: i ;
}
}
```

- A prints 3010
- B prints 3020
- C will result in a run time error
- D none of the above

Correct Answer :A

Explanation

:: is basically meant to manipulate a global variable, in case a local variable also has the same name.

#473 Explained Report Bookmark

The following program

```
void abc (int &p)

{

cout  << p ;

}

void main (void)

{

float m = 1.23;

abc (m) ;
```

cout << m;

}

- **A**
  results in compilation error
- **B**
  results in run time error
- **C**
  prints 1. 23
- **D**
  prints I

Correct Answer :A

Explanation

'cout' was not declared in this scope and invalid initialization of non-const reference of type 'int&' from an rvalue of type 'int'

#474 Explained Report Bookmark

Which of the following is not a storage class supported by C++ ?

- **A**
  register
- **B**
  auto
- **C**
  mutable
- **D**
  dynamic

Correct Answer :D

Explanation

C++ uses 5 storage classes, namely:

- auto.
- register.
- extern.
- static.
- mutable.

#475 Explained Report Bookmark

The following program segment

int a = 10 ;

int const &b = a ;

a = l1 ;

cout << a << b ;

- Aresults in compile time error
- Bresults in run time error
- Cprints 1111
- Dnone of the above

Correct Answer :C

Explanation

The very idea of declaring b as a constant integer is to protect it from changes. However, since it is aliased to a variable whose attribute is not

const, the value of b can be indirectly changed by changing the value of a. This is a bad programming practice.

#476 Explained Report Bookmark

The following program fragment

```
void main ( )

{

int x=10;

int &p=x;

cout << &p << &x;

}
```

- A prints 10 and the address of x
- B prints the address of p twice
- C prints the address of x twice
- D Both (b) & (c)

Correct Answer :D

Explanation

int &p=x aliases p to x. This means they refer to the same memory location. So, the address of x will be same as that of p.

#477 Explained Report Bookmark

What will be the output of the following code?

```
import java.awt.Point;
class Testing
{
     public static void main(String[] args)
     {
          Point p1,p2;
          p1=new Point(100,100);
          p2=p1;
          p1.x=200;
          p1.y=200;
          System.out.println("Point 1: " + p1.x + ", " +
p1.y);
          System.out.println("Point 2: " + p2.x + ", " +
p2.y);
     }

}
```

- A Point 1: 200, 200 Point 2: 100, 100
- B Point 1: 100, 100 Point 2: 200, 200
- C Point 1: 200, 200 Point 2: 200, 200
- D Point 1: 100, 100 Point 2: 100, 100

Correct Answer :C

Explanation

The expected output would be like p2 with values 100, 100. But this is not the case. The tricky part is assignment used (p2=p1;). Here a reference is created from object p1 to p2, and not any new object that would copy p1's values. Hence when we change the values of p1 object members. There changes are reflected to the object p2 also.

#478 Explained Report Bookmark

Which among the following is false, for member function of a class ?

- A All member functions must be defined
- B Member functions can be defined inside or outside the class body
- C Member functions need not be declared inside the class definition
- D Member functions can be made friend to another class using friend keyword

Correct Answer :C

Explanation

Member functions must be declared inside class body, thought the definition can be given outside the class body. There is no way to declare the member functions inside the class.

#479 Explained Report Bookmark

Which of the following best defines a class ?

- A
  Parent of an object
- B
  Instance of an object
- C
  Blueprint of an object
- D
  Scope of an object

Correct Answer :C

Explanation

A class is Blueprint of an object which describes/ shows all the functions and data that are provided by an object of a specific class. It can't be called as parent or instance of an object. Class in general describes all the properties of an object.

#480 Explained Report Bookmark

Which Feature of OOP illustrated the code reusability ?

- A Polymorphism
- B Abstraction
- C Encapsulation
- D Inheritance

Correct Answer :D

Explanation

Using inheritance we can reuse the code already written and also can avoid creation of many new functions or variables, as that can be done one time and be reused, using classes.

#481 Explained Report Bookmark

Which concept of OOP is false for C++ ?

- A Code can be written without using classes
- B Code must contain at least one class
- C A class must have member functions
- D At least one object should be declared in code

Correct Answer :B

Explanation

In C++, it’s not necessary to use classes, and hence codes can be written without using OOP concept. Classes may or may not contain member functions, so it’s not a necessary condition in C++. And, an object can only be declared in a code if its class is defined/included via header file.

#482 Explained Report Bookmark

Which among the following is correct ?

- A class student{ public: student(){} };
- B class student{ public: student{}{} };
- C class student{ public: int student(){} };
- D class student{ public: void student (){} };

Correct Answer :A

Explanation

The constructors must not have any return type. Also, the body may or may not contain any body. Defining default constructor is optional, if you are not using any other constructor.

#483 Explained Report Bookmark

Which is most appropriate comment on following class definition ?

```
class Student
{
    int a;
    public : float a;

};
```

- AError : same variable name can’t be used twice
- BError : data types are different for same variable
- CError : Public must come first
- DIt is correct

Correct Answer :A

Explanation

Same variable can’t be defined twice in same scope. Even if the data types are different, variable name must be different. There is no rule like Public member should come first or last.

#484 Explained Report Bookmark

What is the output of following code?

```
int n=10;       // global
class A()
{
      private : int n;
      public : int m;
      A()
      {
            n=100; m=50;
      }
void disp()
{
      cout<<"n"<<m<<n;

};
```

- A 1050100
- B 1005010
- C n5010
- D n50100

Correct Answer :D

Explanation

n cout we have specified n as a string to be printed. And m is a variable so its value gets printed. And global variable will not be used since local variable have more preference.

#485 Explained Report Bookmark

How many basic features of OOP are required for a programming language to be purely OOP ?

- A 5
- B 6
- C 7
- D 8

Correct Answer :C

Explanation

There are 7 basic features that define whether a programing language is pure OOP or not. The 4 basic features are inheritance, polymorphism, encapsulation and abstraction. Further, one is, object use is must, secondly, message passing and lastly, Dynamic binding.

#486 Explained Report Bookmark

Which class/set of classes can illustrate polymorphism in the following code?

```
abstract class student

{

   public : int marks;

   calc_grade();
```

```
}
class topper:public student
{
    public : calc_grade()
    {
        return 10;
    }
};
class average:public student
{
     public : calc_grade()
     {
         return 20;
     }
};
class failed{ int marks; };
```

- AOnly class student can show polymorphism

- B
  Only class student and topper together can show polymorphism
- CClass failed should also inherit class student for this code to work for polymorphism
- DAll class student, topper and average together can show polymorphism

Correct Answer :D

Explanation

Since Student class is abstract class and class topper and average are inheriting student, class topper and average must define the function named calc_grade(); in abstract class. Since both the definition are different in those classes, calc_grade() will work in different way for same input from different objects. Hence it shows polymorphism.

#487 Explained Report Bookmark

In multiple inheritance, if class C inherits two classes A and B as follows, which class constructor will be called first ?

```
class A{ };

class B{ };

class C: public A, public B{  };
```

- AA()
- BB()
- CC()
- DCan’t be determined

Correct Answer :A

Explanation

Constructor of class A will be called first. This is because the constructors in multiple inheritance are called in the sequence in which they are written to be inherited. Here A is written first, hence it is called first.

#488 Explained Report Bookmark

If a function can perform more than 1 type of tasks, where the function name remains same, which feature of OOP is used here ?

- A Encapsulation
- B Polymorphism
- C Inheritance
- D Abstraction

Correct Answer :B

Explanation

For the feature given above, the OOP feature used is Polymorphism. Example of polymorphism in real life is a kid, who can be a student, a son, a brother depending on where he is.

#489 Explained Report Bookmark

What do you call the languages that support classes but not polymorphism?

- A Class based language
- B Procedure Oriented language
- C Object-based language
- D If classes are supported, polymorphism will always be supported

Correct Answer :C

Explanation

The languages which support classes but doesn’t support polymorphism, are known as object-based languages. Polymorphism is such an important feature, that is a language doesn’t support this feature, it can’t be called as a OOP language.

#490 Explained Report Bookmark

Virtual keyword is used

- A to remove static linkages
- B to call function based on kind of object it is being called for
- C to call the methods that don't exist at compile time
- D all the above

Correct Answer :D

Explanation

All the options are correct as virtual is the keyword used with function of base class then all the functions with same name in derived class, having different implementations are called instead of base class function being called each time. Compiler creates vtables for each class with virtual function so whenever an object for the virtual classes is created compiler inserts a pointer pointing to vtable for that object. Hence when the function is called the compiler knows which function is to be called actually. Example: class geom { public: virtual void show( ) { cout << ” the area is: x” << endl; } }; class rect: public geom { void show( ) { cout << ”the area is: y” << endl; } }; int main( ) { geom* a; rect b; a= &b; a - > show( ); } output: the area is: y

#491 Explained Report Bookmark

Choose the incorrect statements.

- **A**
  A destructor can take arguments
- **B**
  A constructor cannot be called explicitly
- **C**
  A constructor is not inherited
- **D**
  All of these

Correct Answer :A

Explanation

Destructor function is automatically invoked when the objects are destroyed. It cannot be declared static or const. The destructor does not have arguments

#492 Explained Report Bookmark

Which of the following statement is correct?

- **A**
  A constructor has a different name than the class in which it is present.
- **B**
  A constructor always returns an integer.
- **C**
  Destructors can take arguments but constructors cannot.
- **D**
  A constructor has the same name as the class in which it is present.

Correct Answer :D

Explanation

the constructor should always have the same name as the class. Constructor looks like method but it is not. It does not have a return type and its name is same as the class name. Mostly it is used to instantiate the instance variables of a class.

#493 Explained Report Bookmark

The following program fragment

```
void main ( )

{

int x=10;

int &p=x;

cout << &p << &x;

}
```

- A prints 10 and the address of x
- B prints the address of p twice
- C prints the address of x twice
- D Both (b) & (c)

Correct Answer :D

Explanation

int &p=x aliases p to x. This means they refer to the same memory location. So, the address of x will be same as that of p.

#494 Explained Report Bookmark

If many functions-have the same name, which of the following information, if present, will be used by the compiler to invoke the correct function to be used?

- **A**
  The operator : :
- **B**
  The return value of the function
- **C**
  Function signature
- **D**
  Both (a) & (c)

Correct Answer :D

Explanation

An operator is a symbol that tells the compiler to perform specific mathematical or logical functions. A function signature (or type signature, or method signature) defines input and output of functions or methods.

#495 Explained Report Bookmark

A function abc is defined as

void abc(int x=0, int y, int z=0)

{

cout << x << y << z;

}

Which of the following function calls is/are illegal? (Assume h, g are declared as integers)

- A abc () ;
- B abc (h) ;
- C abc (h, h) ;
- D Both (a) & (b)

Correct Answer :D

Explanation

Since the second argument is mandatory, any call should have at least the first two parameters. Some compilers expect the optional parameters to follow the others. Such compilers give a compilation error.

#496 Explained Report Bookmark

Reference is not same as pointer because

- A
  a reference can never be null
- B
  a reference once established cannot be changed.
- C
  reference doesn't need an explicit dereferencing mechanism.
- D
  All of these

Correct Answer :D

Explanation

When you create a reference, you only tell the compiler that you assign another name to the pointer variable; that's why references cannot "point to null", because a variable cannot be, and not be. ... The important thing is that a pointer has a value, while a reference only has a variable that it is referencing.

#497 Explained Report Bookmark

How many arguments a ternary operator takes ?

- **A**
  1
- **B**
  2
- **C**
  3
- **D**
  4

Correct Answer :C

Explanation

a ternary operator is an operator that takes three arguments (or operands). The arguments and result can be of different types. Many programming languages that use C-like syntax feature a ternary operator, ?: , which defines a conditional expression.

#498 Explained Report Bookmark

In argument value passed will get modified only inside the function is called

- **A**
  Call by Reference
- **B**
  Call by Value

- C
  Call by function
- D
  Call by Variable

Correct Answer :B

Explanation

Pass by value means that a copy of the actual parameter's value is made in memory, i.e. the caller and callee have two independent variables with the same value. If the callee modifies the parameter value, the effect is not visible to the caller.

#499 Explained Report Bookmark

Which operators cannot be overloaded?

- A
  . (Member Access or Dot operator)
- B
  ?: (Ternary or Conditional Operator)
- C
  :: (Scope Resolution Operator)
- D
  All of the above

Correct Answer :D

Explanation

List of operators that can be overloaded are:

+ - * / % ^

```
&     |     ~     !,          =

      =        ++          --

      ==      !=        &&          ||

+=      -=      /=      %=        ^=          &=

|=      *=      =        []          ()

->      ->*      new      new []        delete      delete []
```

#500 Explained Report Bookmark

If I want to have common functions in a class and want to defer implementations of some other functions to derived classes, then we need to use

- AAn interface
- BAn abstract class
- CA friend class
- DA static class

Correct Answer :B

Explanation

In C++ object oriented programming, abstract class is used for the same, in which we have common or say generalized function in abstract base class

and also may have pure virtual function in this class that forces derived classes to implement it.

#501 Explained Report Bookmark

A class is made abstract by declaring at least one of its functions as?

- A
  abstract classes
- B
  abstract functions
- C
  Interface
- D
  pure virtual function

Correct Answer :D

Explanation

A class is made abstract by declaring at least one of its functions as pure virtual function.

#502 Explained Report Bookmark

In a class, encapsulating an object of another class is called

- AComposition
- BInheritance
- CEncapsulation
- DNone

Correct Answer :A

Explanation

In simple word, if a class contains an object of another class as a data member, then it is known as composition. For example, Class Y, have a class X's object as data member. Means, Y is composed of X. class X { public: void f1() { } }; class Y{ X obj;//class object as a data member public: void f2() { } };

#503 Explained Report Bookmark

Which is private member functions access scope?

- **A**
  Member functions which can only be used within the class
- **B**
  Member functions which canâ€™t be accessed inside the class
- **C**
  Member functions which can used outside the class
- **D**
  Member functions which are accessible in derived class

Correct Answer :A

Explanation

The member functions can be accessed inside the class only if they are private. The access is scope is limited to ensure the security of the private members and their usage.

#504 Explained Report Bookmark

If you want to write multiple functions in a class with same name, then what C++ feature will you use?

- AFunction overriding
- BEncapsulation
- CFunction overloading
- DNone

Correct Answer :C

Explanation

Compile time polymorphism feature is used i.e. function overloading in C++. Function overloading means, in a class, multiple functions with same name can be written with different signatures, return types etc.

#505 Explained Report Bookmark

Which of the following function declaration using default arguments is correct?

- Aint foo(int x, int y =5, int z=10)
- Bint foo(int x=5, int y =10, int z)
- Cint foo(int x=5, int y, int z=10)
- Dall are correct

Correct Answer :A

Explanation

Default arguments in a function in C++ program is initialized from right to left.

#506 Explained Report Bookmark

Which C++ oops feature is related to re-usability?

- AEncapsulation
- BInheritance
- CAbstraction
- DNone

Correct Answer :B

Explanation

Inheritance feature is used for concept of code re-usability as in inheritance a class can inherit properties and functions of existing well written class. Abstraction : Provide only necessary information to client code. Encapsulation: Hide complexity. e.g. by wrapping private class data members by functions. By making internal function of a class private and using interfaces etc.

#507 Explained Report Bookmark

Can main() function be made private?

- A
  No, never
- B
  Yes, always
- C
  Yes, if program doesn't contain any classes
- D
  No, because main function is user defined

Correct Answer :A

Explanation

The reason given in c option is wrong. The proper reason that the main function should not be private is that it should be accessible in whole program. This makes the program flexible.

#508 Explained Report Bookmark

Which of the following functions are provided by compiler by default if we don’t write in a C++ class?

- A Copy constructor
- B Assignment
- C Constructor
- D All the above

Correct Answer :D

Explanation

The all functions are provided by the compiler, if we don’t write in a class.

#509 Explained Report Bookmark

Which is private member functions access scope?

- A Member functions which can used outside the class
- B Member functions which are accessible in derived class
- C Member functions which can only be used within the class
- D Member functions which can’t be accessed inside the class

Correct Answer :C

Explanation

The member functions can be accessed inside the class only if they are private. The access is scope is limited to ensure the security of the private members and their usage.

#510 Explained Report Bookmark

Which of the following are valid characters for a numeric literal constant?

- **A**
  decimal point
- **B**
  the letter e
- **C**
  a minus sign
- **D**
  All of the above

Correct Answer :D

Explanation

Constants refer to fixed values that the program may not alter and they are called literals. Constants can be of any of the basic data types and can be divided into Integer Numerals, Floating-Point Numerals, Characters, Strings and Boolean Values.

#511 Explained Report Bookmark

An expression A.B in C++ means _____

- **A**
  A is member of object B
- **B**
  B is member of Object A
- **C**
  Product of A and B

- **D**
  None of these

Correct Answer :B

Explanation

The data members and member functions of class can be accessed using the dot('.') operator with the object. For example if the name of object is *obj* and you want to access the member function with the name *printName()* then you will have to write *obj.printName()* .

#512 Explained Report Bookmark

Functions that returns information about an object's state can be classified as _____

- **A**
  inspector functions
- **B**
  mutator functions
- **C**
  auxiliary functions
- **D**
  manager functions

Correct Answer :A

Explanation

An object's state is returned without modifying the object's abstract state by using a function called inspector. Invoking an inspector does not cause any noticeable change in the object's behavior of any of the functions of that object.

#513 Explained Report Bookmark

To create a template class, you begin with ____

- **A**
  the template definition
- **B**
  the keyword class
- **C**
  the function definitions
- **D**
  the keyword definition

Correct Answer :A

Explanation

The syntax must *start* with keyword *template*, case sensitive. Then it should include the typename and a variable to denote it. Then whenever that variable is used, it replaces it with the data type needed.

#514 Explained Report Bookmark

An exception specification begins with the keyword _____

- **A**
  exception

- B
  try
- C
  throw
- D
  catch

Correct Answer :C

Explanation

an exception specification begins the keyword throw. the throw keyword in java is used to explicitly throw an exception from a method or any block of code.

#515 Explained Report Bookmark

A measure of the strength of the connection between two functions is

- A
  cohesion
- B
  coupling
- C
  dependence
- D
  subjection

Correct Answer :B

Explanation

coupling is the degree of interdependence between software modules; a measure of how closely connected two routines or modules are; the strength of the relationships between modules. Coupling is usually contrasted with cohesion. Low coupling often correlates with high cohesion, and vice versa.

#516 Explained Report Bookmark

Which of the following statements opens a file named temp.dat for output?

- **A**
  outFile.open("temp .dat");
- **B**
  fileOut.output("temp .dat");
- **C**
  openFile.out("temp .dat");
- **D**
  ileOpen.out("temp .dat");

Correct Answer :A

Explanation

The OUTFILE command lets you redirect the text output of statements to a file. The process of writing to a file is very similar to the process of reading from a file in that both require opening a file object for access. The difference is in the second argument to open(), in which the string "w" i.e. write is passed.

#517 Explained Report Bookmark

The delete operator returns _____ to the operating system

- **A**
  memory that is no longer needed

- **B**
  void
- **C**
  recycle bin
- **D**
  None of the above

Correct Answer :A

Explanation

The delete operator doesn't return any value. Its function is to delete the memory allocated for an object.

#518 Explained Report Bookmark

Which is a good guideline for creating function names?

- **A**
  Use all lowercase letters to identify the functions as C++ functions
- **B**
  Use long names to reduce the likelihood of creating a duplicate function name
- **C**
  Use abbreviations as much as possible to save both keystrokes and memory
- **D**
  Avoid the use of digits because they are easily confused with letters

Correct Answer :D

Explanation

There's nothing wrong with having numbers in your function name, it's just a little unconventional.

#519 Explained Report Bookmark

The null character is represented by

- A
  \n
- B
  \0
- C
  \o
- D
  \r

Correct Answer :B

Explanation

The null character is often represented as the escape sequence \0 in source code string literals or character constants.

#520 Explained Report Bookmark

The string HELLO WORLD needs

- A
  11 bytes
- B
  12 bytes

- C
  10 Bytes
- D
  8 bytes

Correct Answer :A

Explanation

Total size (in Bytes) = ((Number of bits used to encode a single character) * (Number of characters))/8

Lets say we want to store “Hello World”, so we have 10 characters and a space, so that makes Number of characters = 11 .

If we are using ASCII encoding, we need 8 bits to encode each character, so Number of bits used to encode a single character = 8

Total size = (11 * 8)/8 = 11 Bytes.

#521 Explained Report Bookmark

The keyword 'this' is used

- AAs reference to current object
- BIn open recursion
- CExplicit constructor invocation
- D
  All the above

Correct Answer :D

Explanation

In object oriented programming the keyword 'this' is used to refer to objects, classes or any other entity that is part of currently running code. Different languages use it in different ways. In some languages, 'this' is the only way to access data and methods in current object. The concept, however, is similar in all languages using 'this'.It is usually a fixed reference or pointer which refers to current object. In some languages it is used explicitly while others use lexical scoping(range of functionality) to use it implicitly to make symbols within their class visible. The dispatch semantics of 'this' that method calls on 'this'are dynamically dispatched which means that these methods can be overriden by derived classes or objects.

#522 Explained Report Bookmark

Pure virtual function is used

- A to give meaning to derived class function
- B to give meaning to base class function
- C to initialize all functions
- D
  none

Correct Answer :B

Explanation

When we use virtual keyword our base class function becomes meaningless as it has no use. It only helps in calling all derived class functions of same name. If base class function is initialized with =0, it is known that function has no body and thus the virtual function of base class becomes pure virtual function. For example: class geom { public: virtual void show( )=0; }; class rect: public geom { void show( ) { cout << " the area is : y" << endl; } }

#523 Explained Report Bookmark

An Abstract class

- A allows normal method declaration within
- B can be instantiated
- C Must have abstract methods with implemetation within
- D
  none

Correct Answer :A

Explanation

Abstraction is the process of hiding implementation details and showing only functionality to the user. Classes that are declared with keyword abstract are called abstract classes. It cannot have any instances. It can carry normal functions declaration and definition along with abstract functions though the abstract methods(functions) cannot be defined inside the abstract class.For defining abstract methods subclasses are needed. Any class that extends the abstract class must implement all abstract methods declared by the super class.

#524 Explained Report Bookmark

Why is user defined copy constructor required?

- A there is no implicit copy constructor in C++
- B when pointers are involved implicit copy constructor does not give correct result
- C both a and b
- D
  none

Correct Answer :B

Explanation

C++ has an implicit copy constructor which is called by compiler to keep a copy of the object. In case where pointers are present in the code and programmer is not using user defined copy constructor then a problem occurs. The problem is that whenever a copy constructor is called(implicit or user defined) the copy destructor is also called to delete the copy at the end of scope. Implicit copy constructor copies an object bit by bit so pointer address will be copied causing in two different objects sharing same memory location. When the first object calls the destructor to deallocate the pointer no problem occurs but when second object does so it tries to deallocate a pointer that does not exist anymore and the application stops working. In order to prevent situations like this user defined copy constructor is called.

#525 Explained Report Bookmark

Dynamic dispatch is a feature that

- A selects which polymorphic operation to call at run time
- B selects which polymorphic operation to call at compile time
- C Both a and b
- D none

Correct Answer :A

Explanation

Dynamic dispatch also known as message passing is a process of selecting a procedure to run in response to a method call by looking for the method (function) in the table associated with the object, at run time. It distinguishes an object from a module which has fixed implementations for

all instances i.e. static dispatch. Dynamic dispatch has usage in Object oriented programming languages when different classes have different implementations of same function call due to inheritance. For example there are three classes. Class X- the base class, Class Y and Class Z- the derived class and all of them have a function call- show( ), then dynamic dispatch selects which implementation of function to call at run time.

#526 Explained Report Bookmark

Function Templates can have

- A Explicit instantiation definition with template argument for all parameters
- B explicit instantiation of declaration with template argument for all parameters
- C Both a and b
- D null

Correct Answer :C

Explanation

A function template cannot be defined as a type or a function in itself. There is no such coding available that contains only template definitions generated from a source file. A template must have real instance i.e. template arguments must be determined such that the compiler generates an actual function. Here both first and second options are correct. An explicit formation of instance definition forces instantiation of member functions. For instantiating a function template all template arguments must be known.

#527 Explained Report Bookmark

Use of preprocessor directive in OOP

- A for conditional compilation
- B for macro expansion
- C error and warning reporting
- D
  all the above

Correct Answer :d

Explanation

Preprocessor is a program that processes an input to produce an output which is used as another programs input. The preprocessor directives are generally invoked by the compiler to process before compilation. The preprocessor directive are capable of performing simple textual substitutions, macro expansion, conditional compilation, warning and error reporting. It begins with special character # followed by directive name. Preprocessor directive statement does not end with a semi colon. Few examples of preprocessor directives are: #include, #define, #undef, #if, #elif, #else, #endif, #line etc. It is used in various OOP languages like C, C++, C#. It has no use in Java.

#528 Explained Report Bookmark

Encapsulation helps

- A information hiding
- B in providing low coupling
- C in providing high cohesion
- D
  All the above

Correct Answer :D

Explanation

Encapsulation works on providing interactions through function calling only. Using keyword private or protected stops the use of data members outside class. The data thus remains accessible to functions inside class. Encapsulation is also called information hiding as it restricts the use of data inside class. Coupling is the degree of interdependence between different blocks(modules) of program code. Low coupling helps in keeping data safe inside the class. Coupling contrasts cohesion as with high cohesion comes reliability and reusability in a code.

#529 Explained Report Bookmark

Which property of Object Oriented Programming is exhibited by the code:

```
#include< iostream>

class quest

            {

               public:

           float add( float a, float b)

           {

             cout << “The sum is:”;

              reurn(a+b); }   }

          int add( int a, int b, int c)

          {

 cout << ”The sum is:”;

              return(a+b+c);
```

```
    }
int add( int a, int b=0)
{
   cout << ”The sum is:”;
   return(a+b);
    }
};
int main( )
{
quest q
   q. add( 50.5, 48.2)
   q. add( 42, 56, 82)
   q. add( 23)
    return 0;
   }
```

- **A** Compile time polymorphism
- **B** Run time polymorphism
- **C** Both
- **D** none

Correct Answer :A

Explanation

The property of function overloading is also known as Compile time polymorphism. Here in this code we see how a function is overloaded using different datatype, different number of arguments and default argument. Function overloading is a concept which allows the use of same function name to different functions for performing same or different task in the same class.

#530 Explained Report Bookmark

Give output of the following:

```
#include <iostream>

using namespace std;

class exam

{ private:

int x, y, z;

public:

int testcase( )

{

x=50; y=20; z=30;

}

friend int add( exam e);
```

```
};

int add( exam e)

{

return int( e.x+ e.y+ e.z);

}

int main( )

{

exam a;

a.testcase( );

cout << add(a);

return 0;

}
```

- **A**
  0
- **B**
  100
- **C**
  compile time error
- **D**
  none

Correct Answer :B

Explanation

This code shows usage of a friend function and how a friend function has access to private members of a class. Generally a function defined outside the class cannot access the private and protected members of the class but by declaring a function as friend function the class gives it permission to access its private and protected data.

#531 Explained Report Bookmark

Constructor chaining is

- A subclass constructor calling super class constructor
- B super class constructor calling subclass constructor
- C both
- D none

Correct Answer :A

Explanation

In Inheritance subclass or derived class inherits properties of super class or parent class. Constructor chaining is the property by which a constructor in derived class(s) can call constructor of parent class(s). In constructor chaining, when an object of derived class is created its constructor is called which further calls the constructor method of parent class. The keyword super in java helps in passing arguments to super class constructor. The creation of subclass object starts with initialization of class(s) above it in inheritance chain. There could be any number of classes in the inheritance chain. Every constructor function will call up the chain till the top most class is reached. For Example: public class animal { private string str; public animal(string str) { this.name= name; system.out.println(“I am an Animal”); } } public class lion extends animal { public lion(string str) { super(str); system.out.println(“I am a lion”); } }

#532 Explained Report Bookmark

One way pointers are useful is to refer to a memory address that has no _____

- **A**
  name
- **B**
  constant
- **C**
  location
- **D**
  field

Correct Answer :A

Explanation

Pointer arithmetic provides the programmer with a single way of dealing with different types: adding and subtracting the number of elements required instead of the actual offset in bytes.

#533 Explained Report Bookmark

The while loop is referred to as a(n) _____ loop because the loop condition is tested at the beginning of the loop

- **A**
  beginning
- **B**
  initial
- **C**
  pretest
- **D**
  priming

Correct Answer :C

Explanation

- In a pretest loop, the condition is evaluated before the instructions in the loop are processed.
- In a posttest loop, the condition is evaluated after the instructions in the loop are processed.

#534 Explained Report Bookmark

In a simple 'if' statement with no 'else'. What happens if the condition following the 'if is false?

- **A**
  the program searches for the last else in the program
- **B**
  nothing
- **C**
  control 'falls through' to the statement following 'if
- **D**
  the body of the statement is executed

Correct Answer :C

Explanation

It's not required to write the else part for the if statement.

if we have only *if statement* then tells the program to execute a block of code, *if* a *condition* is true.

#535 Explained Report Bookmark

A normal C++ operator that acts in special ways on newly defined data types is said to be

- A
  glorified
- B
  encapsulated
- C
  classified
- D
  overloaded

Correct Answer :D

Explanation

C++ has the ability to provide the operators with a special meaning for a data type, this ability is known as operator overloading.

#536 Explained Report Bookmark

A fundamental type such as int or double is a _____

- A
  programmer-defined type
- B
  complex type
- C
  nonscalar type
- D
  scalar type

Correct Answer :D

Explanation

A scalar is a simple single numeric value (as in 1, 2/3, 3.14, etc.), usually integer, fixed point, or float (single or double), as opposed to an array, structure, object, complex vector (real plus imaginary or magnitude plus angle components), higher dimensional vector or matrix (etc.)

#537 Explained Report Bookmark

Simple routines that programmers use as place holders while a system is being tested are called _____

- **A**
  stubs
- **B**
  stumps
- **C**
  holders
- **D**
  templates

Correct Answer :A

Explanation

A stub is a placeholder for a method that hasn't been completed, it generally contains comments as to how the method will be implemented and contains a dummy return value so the source file will compile

#538 Explained Report Bookmark

A function that changes the state of the cout object is called a(n) _____

- **A**
  member
- **B**
  adjuster
- **C**
  manipulator
- **D**
  operator

Correct Answer :C

Explanation

Manipulators are helper function which can be used for formatting input and output data. Manipulators specifically designed to be used in conjunction with the insertion (<<) and extraction (>>) operators on stream objects, for example: The iostream.

#539 Explained Report Bookmark

A group of related fields that contain all of the data about a specific person, place, or thing is called a

- **A**
  data file
- **B**
  field file
- **C**
  program file
- **D**
  record

Correct Answer :D

Explanation

A record is a collection of related fields. An Employee record may contain a name field(s), address fields, birthdate field and so on. A file is a collection of related records.

Tables are also called datasheets. Each table in a database holds data about a different, but related, subject. Data is stored in records. A record is composed of fields and contains all the data about one particular person, company, or item in a database.

#540 Explained Report Bookmark

Code that has already been tested is said to be _____

- **A**
  inherited
- **B**
  reusable
- **C**
  reliable
- **D**
  polymorphic

Correct Answer :C

Explanation

Already been tested code is called reliable.

#541 Explained Report Bookmark

Which of the following are valid characters for a numeric literal constant?

- A
  a decimal point
- B
  the letter e
- C
  a minus sign
- D
  All of the above

Correct Answer :D

Explanation

The following rules govern the formation of numeric literals:

- A literal must contain at least one digit
- A literal must contain no more than one sign character and, if one is used, it must be the leftmost character of the string
- A literal must not contain more than one decimal point. The decimal point is treated as an assumed decimal point and may appear anywhere within the literal except as the rightmost character

If a literal conforms to the rules for formation of a numeric literal, but is enclosed in quotation marks, it is a nonnumeric literal.

#542 Explained Report Bookmark

Which is true?

- A
  Sequential cohesion is slightly weaker than functional cohesion
- B
  Sequential cohesion is slightly stronger than functional cohesion
- C
  Sequential cohesion is much stronger than functional cohesion

- **D**
  Neither sequential cohesion nor functional cohesion is stronger than the other

Correct Answer :A

Explanation

Functional Cohesion: Every essential element for a single computation is contained in the component. A functional cohesion performs the task and functions. It is an ideal situation. Sequential Cohesion: An element outputs some data that becomes the input for other element, i.e., data flow between the parts. It occurs naturally in functional programming languages.

#543 Explained Report Bookmark

A function argument is

- **A**
  a variable in the function that receives a value from the calling program
- **B**
  a way that functions resist accepting the calling program's values
- **C**
  a value sent to the function by the calling program
- **D**
  a value returned by the function to the calling program

Correct Answer :C

Explanation

Arguments are the values passed from a function call (i.e., they are the values appearing inside the parentheses of the call) and are sent into the function).

#544 Explained Report Bookmark

n C++, class definitions are most often

- **A**
  stored with each program that uses them
- **B**
  stored in a header file that is included in the programs that use them
- **C**
  stored in a folder that you paste into every new project
- **D**
  retyped for every new project

Correct Answer :B

Explanation

C++ classes (and often function prototypes) are normally split up into two files. The header file has the extension of . h and contains class definitions and functions. The implementation of the class goes into the . cpp file.

#545 Explained Report Bookmark

Hiding individual components of an entry is ___

- **A**
  polymorphism
- **B**
  encapsulation
- **C**
  scaling

- **D**
  not recommended in C++

Correct Answer :B

Explanation

Encapsulation is defined as wrapping up of data and information under a single unit. In Object Oriented Programming, Encapsulation is defined as binding together the data and the functions that manipulates them.

#546 Explained Report Bookmark

Two access specifers in C++ are

- **A**
  public and private
- **B**
  int and double
- **C**
  formal and informal
- **D**
  void and free

Correct Answer :A

Explanation

In C++, there are three access specifiers: public - members are accessible from outside the class. private - members cannot be accessed (or viewed) from outside the class.

#547 Explained Report Bookmark

A measure of the strength of the connection between two functions is

- **A**
  cohesion
- **B**
  coupling
- **C**
  dependence
- **D**
  subjection

Correct Answer :B

Explanation

coupling is the degree of interdependence between software modules; a measure of how closely connected two routines or modules are; the strength of the relationships between modules.

#548 Explained Report Bookmark

Which of the following statements opens a file named temp.dat for output?

- **A**
  outFile.open("temp .dat");
- **B**
  fileOut.output("temp .dat");
- **C**
  openFile.out("temp .dat");
- **D**
  fileOpen.out("temp .dat");

Correct Answer :A

Explanation

The OUTFILE command lets you redirect the text output of statements to a file. The process of writing to a file is very similar to the process of reading from a file in that both require opening a file object for access. The difference is in the second argument to open(), in which the string "w" i.e. write is passed.

#549 Explained Report Bookmark

At what point of time a variable comes into existence in memory is determined by its

- **A**
  Storage class
- **B**
  Data type
- **C**
  Scope
- **D**
  All of the above

Correct Answer :A

Explanation

The answer is storage class. At what point of time a variable comes into existence in memory is determined by its storage class. ... The storage class for a variable determines its visibility and lifetime. The visibility of a variable determines the module of the program that accesses the variable

#550 Explained Report Bookmark

The standard output stream, which refers to the computer screen, is called

- **A**
  cin
- **B**
  cout
- **C**
  stin
- **D**
  stout

Correct Answer :B

Explanation

Usually the standard output device is the display screen. The C++ cout statement is the instance of the ostream class. It is used to produce output on the standard output device which is usually the display screen.

#551 Explained Report Bookmark

If you want only one memory location to be reserved for a class variable, no matter how many objects are instantiated, you should declare the variable as

- **A**
  static
- **B**
  unary
- **C**
  dynamic
- **D**
  volatile

Correct Answer :A

Explanation

Static memebers of a class share same memory by all the object. if u want to check just make a class and define a variable static in it

#552 Explained Report Bookmark

Which of the following is a C++ class?

- **A**
  >>
- **B**
  read()
- **C**
  cin
- **D**
  iostream

Correct Answer :D

Explanation

In C++ there are number of stream classes for defining various streams related with files and for doing input-output operations. All these classes are defined in the file iostream.h.

#553 Explained Report Bookmark

Every non const member function of a class is a_____

- **A**
  constructor

- **B**
  destructor
- **C**
  mutator
- **D**
  friend

Correct Answer :C

Explanation

Mutators are member functions that can change the state (i.e. the member variables inside) of an object

Accessors are member functions that will *not* change the state of an object. Such functions should always be declared with the keyword const at the end. These are known as "const member functions" and will not be permitted to change the object

#554 Explained Report Bookmark

Number of keywords are available in c++ are _____

- **A**
  32
- **B**
  27
- **C**
  21
- **D**
  63

Correct Answer :D

Explanation

There are 63 keywords currently defined for Standard C++. Early versions of C++ defined the overload keyword, but it is obsolete. Keep in mind that C++ is a case-sensitive language, and it requires that all keywords be in lowercase.

#555 Explained Report Bookmark

Storing address of derived class object into base class pointer. Such concept is called as __.

- **A**
  up casting.
- **B**
  down casting
- **C**
  object slicing
- **D**
  none of above

Correct Answer :A

Explanation

using upcasting allows us to treat the child class object as if it were the parent class object.

#556 Explained Report Bookmark

Size of object of empty class is always _____

- **A**
  1 byte

- **B**
  8 bits
- **C**
  Both of above
- **D**
  8 bytes

Correct Answer :C

Explanation

Size of an empty class is not zero. It is 1 byte generally. It is nonzero to ensure that the two different objects will have different addresses

8bit = 1 byte

#557 Explained Report Bookmark

What will be output of the following program on 64 bit compilation ?

```
#include<iostream.h>

using namespace std;

#pragma pack(1)



class A

    {
```

```
int a;
public:
        A()
        {
        }
};
class B: virtual public A
{

 int b;
    public:
        B()
        {
        }
};
class C: virtual public A
{
```

```
        int c;
        public:
            C()
            {
            }
    };
    class D: public C, public B
    {

        int d;
        public:
            D()
            {
            }
    };
int main()
```

```
{

   D objInstance;

   cout<<"sizeof(objInstance)::"<< sizeof(objInstance);



    return 0;

}
```

- A
  sizeof(objInstance)::32 bytes
- B
  sizeof(objInstance)::16bytes
- C
  sizeof(objInstance)::20bytes
- D
  sizeof(objInstance)::24bytes

Correct Answer :A

Explanation

The answer to this question clearly depends on your compiler. If you are using compiler that access and allocate 32 byte of space, then the 32 will be printed on screen

#558 Explained Report Bookmark

if you enter data 10 20 what will be output of the following program ?

```
if you enter data 10 20 what will be output of the following
program ?
```

```
#include <stdio.h>

using namespace std;

void AcceptInput(int &data)

{

    cout<<"Enter Data:: ";

    cin>>data;

}

void PrintOutput(const int &data)

{

    cout<<"Data ::";

}

void AcceptInput(int *data)

{

    cout<<"Enter Data:: ";

    cin>>*data;

}

void PrintOutput(const int *data)
```

```
{
    cout<<"Data ::"<<*data;
}
int main()
{
    int no1;
    AcceptInput(no1);
    AcceptInput(&no1);


    PrintOutput(no1);
    PrintOutput(&no1);
    return 0;
}
```

- **A**
  compile time error
- **B**
  garbage values
- **C**
  Data ::20 Data ::20
- **D**
  run time error

Correct Answer :C

Explanation

Here we are accessing and changing the variable values using pointer and address. & is accessing the address of particular variable and * is used to access value of certain variable pointed by pointer variable.

#559 Explained Report Bookmark

(>>) operator which is used with cin is called as _____ operator.

- **A**
  insertion
- **B**
  in
- **C**
  out
- **D**
  extraction

Correct Answer :D

Explanation

The extraction operator (>>), which is preprogrammed for all standard C++ data types, is the easiest way to get bytes from an input stream object. ... Another method is to derive an input stream class with a member function such as GetNextToken , which can call istream members to extract and format character data

#560 Explained Report Bookmark

Which of the following is an output of linker?

- A
  Object Code
- B
  Assembly Code
- C
  Executable Code
- D
  Intermediate Code

Correct Answer :C

Explanation

The output from the linker is the executable or program. There is no default extension, although some people use .exe which is a standard extension on some operating systems.

#561 Explained Report Bookmark

What will be the output of the program ?

```
#include<iostream>

using namespace std;

int main()

{

    int const a1=10;

    int &r1=a1;
```

```
    cout<<"a1="<<a1   ;   cout<<"r1="<<r1;

    r1=1000;



    cout<<"a1="<<a1   ;   cout<<"r1="<<r1;

    return 0;

}
```

- **A**
  a1=10 r1=10 a1=1000 r1=1000
- **B**
  compile time error
- **C**
  runtime error
- **D**
  a1=10 r1=10 a1=10 r1=1000

Correct Answer :B

Explanation

Binding reference will occur on line “int &r1=a1;” 'int&' to 'const int' discards qualifiers.

#562 Explained Report Bookmark

What will be the output of the program ?

```
#include<iostream>
```

```
using namespace std;

int fun(int a=0, int b, int c=0,int d=0);

int main()

{

    int ans= fun(10,20,30);

    cout<<"ans= "<<ans;

    ans= fun(10,20);

    cout<<"ans= "<<ans;

    ans= fun(10);

    cout<<"ans= "<<ans;

    return 0;

}

int fun(int a, int b, int c,int d)

{

    return a+b+c+d;

}
```

- **A**
  Compile time error

- B
  run time error
- C
  ans= 60 ans= 30 Ans= 10
- D
  garbage values

Correct Answer :A

Explanation

default argument missing for b variable of 'int fun(int, int, int, int)'|

#563 Explained Report Bookmark

Default arguments should be given in _____

- A
  right to left order
- B
  left to right order
- C
  depends of compiler
- D
  none of above

Correct Answer :A

Explanation

When you defines a function with default arguments the compiler checks for arguments from right to left and assigns the values passed from calling function from right to left. Suppose you defined a function,

- void myFunc(int a=0, int b=0, int c);

When you call this function with arguments like,

- myFunc(3,4);

the compiler assigns 4 to c. Now in which parameter the argument 3 will be passed? a or b?? The compiler here generates an error.

error: default argument missing for parameter 3 of 'void myFunc(int, int, int)

This is why we assigns default arguments from right to left.

#564 Explained Report Bookmark

C++ supports all data types provided by C language and C++ adds data types

- **A**
  bool
- **B**
  wchar_t
- **C**
  both 1 and 2
- **D**
  none of above

Correct Answer :C

Explanation

C++ defines two more: bool and wchar_t.

TYPE	DESCRIPTION
bool	Boolean value, true or false
wchar_t	wide character

#565 Explained Report Bookmark

The values stored in data members of the object called as _____ of object.

- **A**
  state
- **B**
  behavior
- **C**
  identity
- **D**
  none of above

Correct Answer :A

Explanation

In C++, Object is a real world entity, for example, chair, car, pen, mobile, laptop etc. In other words, object is an entity that has state and behavior. Here, state means data and behavior means functionality.

#566 Explained Report Bookmark

which of the following statments true about destructor

- **A**
  it is a member function
- **B**
  it is used to finalize object
- **C**
  it does not have any return value
- **D**
  all of above

Correct Answer :D

Explanation

A destructor is a member function that is invoked automatically when the object goes out of scope or is explicitly destroyed by a call to delete . A destructor has the same name as the class, preceded by a tilde (~). For example, the destructor for class String is declared: ~String() .

#567 Explained Report Bookmark

What is the output of this program?

```
/******************************************************************
****************


                                Online C++ Compiler.


                Code, Compile, Run and Debug C++ program
online.


Write your code in this editor and press "Run" button to
compile and execute it.
```

```
**********************************************************************
*****************/

 #include <iostream>

 using namespace std;

 class A

 {

 public:

     A(int n )

     {

     cout << n;

     }

 };

 class B: public A

 {

 public:

 B(int n, double d)
```

```
    : A(n)
    {
    cout << d;
    }
};
class C: public B
{
public:
    C(int n, double d, char ch)
    : B(n, d)
    {
    cout <<ch;
    }
};
int main()
{
C c(5, 4.3, 'R');
```

```
return 0;

}
```

- **A**
  54.3R
- **B**
  R4.35
- **C**
  4.3R5
- **D**
  None of the mentioned

Correct Answer :A

Explanation

We are passing the value and manipulated it by using the derived class. and first base constructor will called so A->B->C

#568 Explained Report Bookmark

Like constructors, can there be more than one destructors in a class?

null

- **A**
  Yes
- **B**
  No
- **C**
  May Be
- **D**
  Can't Say

Correct Answer :B

Explanation

There can be only one destructor in a class. Destructor's signature is always ~ClassNam()and they can not be passed arguments.

#569 Explained Report Bookmark

State whether the following statements about the constructor are True or False

i) constructors should be declared in the private section.

i) constructors are invoked automatically when the objects are created.

- **A**
  True,True
- **B**
  True,False
- **C**
  False,True
- **D**
  False,False

Correct Answer :C

Explanation

Statement 1 is false and statement 2 is true.

#570 Explained Report Bookmark

Which of the following is true about constructors

i) They cannot be virtual

ii) They cannot be private.

iii) They are automatically called by new operator

- **A**
  i,ii,iii
- **B**
  i & iii
- **C**
  ii & iii
- **D**
  i & ii

Correct Answer :B

Explanation

i) True: Virtual constructors don't make sense, it is meaningless to the C++ compiler to create an object polymorphically.

ii) False: Constructors can be private

#571 Explained Report Bookmark

What will be the output of the following program?

```
#include <iostream>

using namespace std;



class LFC
```

```
{
    int id;
    static int count;
public:
    LFC() {
        count++;
        id = count;
        cout << "constructor for id " << id << endl;
    }
    ~LFC() {
        cout << "destructor for id " << id << endl;
    }
};

int LFC::count = 0;

int main() {
```

```
    LFC a[3];

    return 0;

}
```

- **A**
  constructor for id 1 constructor for id 2 constructor for id 3 destructor for id 1 destructor for id 2 destructor for id 3
- **B**
  constructor for id 1 constructor for id 2 constructor for id 3 destructor for id 3 destructor for id 2 destructor for id 1
- **C**
  Compiler Dependent
- **D**
  constructor for id 1
  destructor for id 1

Correct Answer :D

Explanation

In the above program, id is a static variable and it is incremented with every object creation. Object a[0] is created first, but the object a[2] is destroyed first. Objects are always destroyed in reverse order of their creation. The reason for reverse order is, an object created later may use the previously created object.

#572 Explained Report Bookmark

What will be the output of the following program?

```
#include <iostream>
```

```
using namespace std;

class LFC {

    LFC() { cout << "Constructor called"; }

};



int main()

{

   LFC t1;

   return 0;

}
```

- **A**
  Compiler Error
- **B**
  Runtime Error
- **C**
  Constructor called
- **D**
  No Output

Correct Answer :D

Explanation

By default all members of a class are private. Since no access specifier is there for find(), it becomes private and it is called outside the class when t1 is constructed in main.

#573 Explained Report Bookmark

If the copy constructor receives its arguments by value, the copy constructor would

- **A**
  Call one-argument constructor of the class
- **B**
  Work without any problem
- **C**
  Call itself recursively
- **D**
  Call zero-argument constructor

Correct Answer :C

Explanation

If the copy constructor receives its arguments by value, the copy constructor would Call itself recursively.

#574 Explained Report Bookmark

which of this can not be declared as virtual

- **A**
  Both Constructor & Destructor
- **B**
  Destructor
- **C**
  Constructor

- **D**
  None of the above

Correct Answer :C

Explanation

In C++, the constructor cannot be virtual, because when a constructor of a class is executed there is no virtual table in the memory, means no virtual pointer defined yet. So, the constructor should always be non-virtual. But virtual destructor is possible

#575 Explained Report Bookmark

We must use initializer list in a constructor when

null

- **A**
  There is a reference variable in class
- **B**
  There is a constant variable in class
- **C**
  There is an object of another class. And the other class doesn't have default constructor
- **D**
  All of the above

Correct Answer :D

Explanation

Initializer List is used to initialize data members of a class. The list of members to be initialized is indicated with constructor as a comma separated list followed by a colon.

#576 Explained Report Bookmark

Which of the following implicitly creates a default constructor when the programmer does not explicitly define at least one constructor for a class?

null

- **A**
  Preprocessor
- **B**
  Linker
- **C**
  Loader
- **D**
  compiler

Correct Answer :D

Explanation

compiler implicitly creates a default constructor when the programmer does not explicitly define at least one constructor for a class.

#577 Explained Report Bookmark

Which of the following is the correct syntax to add the header file in the C++ program?

null

- **A**
  #include<userdefined>
- **B**
  #include "userdefined.h"
- **C**
  <include> "userdefined.h"
- **D**
  Both A and B

Correct Answer :D

Explanation

To include the herder files in the C++ program user can use any of the following given syntax.

- #include <Filename.h>

Or

- #include "filename.h"

#578 Explained Report Bookmark

Name the function whose definition can be substituted at a place where its function call is made _____

null

- A
  friends function

- **B**
  inline function
- **C**
  volatile function
- **D**
  external function

Correct Answer :B

Explanation

The inline function, whose definitions being small can be substituted at a place where its function call is made. They are inlined with their function calls.

#579 Explained Report Bookmark

What will be the output of the following C code?

```
#include<stdio.h>

#define inline

inline f(char a)

    {

        #ifdef inline

        printf("%c",a);

        #endif
```

```
    }

main()

{

    f('a');

}
```

- **A**
  Error
- **B**
  a
- **C**
  No error but nothing will be printed as output
- **D**
  97

Correct Answer :B

Explanation

In the code shown above, we have defined identifier names inline using the macro #define. The output will be printed only if inline is defined. Since it is defined, ‘a’ is printed as output.

#580 Explained Report Bookmark

What will be the error (if any) in the following C code?

```
#include <stdio.h>
```

```
void inline func1(float b)

{

    printf ("%lf\n",b*2);

}

int main()

{

     inline func1(2.2);

     return 0;

}
```

- **A**
  No error
- **B**
  Error in statement: void inline func1(float b)
- **C**
  Error in statement: printf(“%lf\n”,b*2);
- **D**
  Error in statement: inline func1(2.2);

Correct Answer :D

Explanation

The keyword inline is used only with the function definition. In the code shown above it has been used with the function call. Hence there is an error in the statement: inline func1(2.2);

#581 Explained Report Bookmark

Why would you want to use inline functions?

- **A**
  To decrease the size of the resulting program
- **B**
  To increase the speed of the resulting program
- **C**
  To simplify the source code file
- **D**
  To remove unnecessary functions

Correct Answer :B

Explanation

The inline functions are a C++ enhancement feature to increase the execution time of a program. Functions can be instructed to compiler to make them inline so that compiler can replace those function definition wherever those are being called.

#582 Explained Report Bookmark

Which of the following refers to characteristics of an array?

null

- **A**
  An array is a set of similar data items

- **B**
  An array is a set of distinct data items
- **C**
  An array can hold different types of datatypes
- **D**
  None of the above

Correct Answer :A

Explanation

Basically, an array is a set of similar data items that are stored in the contiguous memory blocks. You can access any data item randomly using the index address of the element s.

#583 Explained Report Bookmark

If we stored five elements or data items in an array, what will be the index address or the index number of the array's last data item?

null

- **A**
  3
- **B**
  5
- **C**
  4
- **D**
  88

Correct Answer :C

Explanation

The array uses the contiguous block of memory for storing the data elements. Hence the data items pushed into an array are stored at the continuous address space. The index number of the array begins with zero so, the first data item pushed into the array will be stored at zero and so on. Hence index number of the last (fifth) element will be 4.

#584 Explained Report Bookmark

Which of the following is the correct syntax for printing the address of the first element?

null

- **A**
  array[0];
- **B**
  array[1];
- **C**
  array[2];
- **D**
  None of the above

Correct Answer :A

Explanation

To print or access the first data item of the array, the correct syntax is "array[0];" because the array's index begins with zero instead of one. So the correct answer will be A.

#585 Explained Report Bookmark

Which type of program is recommended to include in try block?

null

- A
  Static memory allocation
- B
  Dynamic memory allocation
- C
  Const reference
- D
  Pointer

Correct Answer :B

Explanation

While during dynamic memory allocation, Your system may not have sufficient resources to handle it, So it is better to use it inside the try block.

#586 Explained Report Bookmark

What is the output of this program?

```
#include <iostream>

using namespace std;



int main()

{

    try

    {
```

```
        throw 10;

    }

    catch (...)

    {

        cout << "Default Exceptionn";

    }

    catch (int param)

    {

        cout << "Int Exceptionn";

    }

    return 0;

}
```

- **A**
  Default Exception
- **B**
  Int Exception
- **C**
  Compiler Error
- **D**
  None of the above

Correct Answer :C

Explanation

It is compiler error to put catch all block before any other catch. The catch(...) must be the last catch block.

#587 Explained Report Bookmark

Assigning one or more function body to the same name is called

- A
  Function Overloading
- B
  Function Overriding
- C
  Both a and b
- D
  None of the above

Correct Answer :A

Explanation

assigning one or more function body to the same name is called as method or function overloading.

#588 Explained Report Bookmark

Default values for a function are specified when ____ .

null

- A
  function is defined
- B
  function is declared
- C
  Both a and b
- D
  None of these

Correct Answer :B

Explanation

Default values for a function are specified when function is declared.

#589 Explained Report Bookmark

if an argument to a function is declared as const, then _____ .

null

- A
  function can modify the argument.
- B
  Function can’t modify the argument.
- C
  const argument to a function is not possible.
- D
  None of these

Correct Answer :B

Explanation

Function can't modify the argument

#590 Explained Report Bookmark

While overloading binary operators using member function, it requires ___ argument?

- **A**
  2
- **B**
  1
- **C**
  0
- **D**
  3

Correct Answer :B

Explanation

While overloading binary operators using member function, it requires 1 argument.

#591 Explained Report Bookmark

Which of the following operators should be preferred to overload as a global function rather than a member method?

null

- **A**
  Postfix ++
- **B**
  Comparison Operator
- **C**
  Insertion Operator <<

- **D**
  prefix ++

Correct Answer :C

Explanation

Insertion Operator should be preferred to overload as a global function rather than a member method.

#592 Explained Report Bookmark

Which of the following operator functions cannot be global, i.e., must be a member function.

null

- **A**
  new
- **B**
  delete
- **C**
  Converstion Operator
- **D**
  All of the above

Correct Answer :C

Explanation

Converstion Operator functions cannot be global, i.e., must be a member function.

#593 Explained Report Bookmark

When an object is created and initialized at the same time, a ……. gets called.

null

- **A**
  default constructor
- **B**
  parameterized constructor
- **C**
  implicit constructor
- **D**
  copy constructor

Correct Answer :D

Explanation

A copy constructor takes reference to an object of the same class as itself as an argument.

#594 Explained Report Bookmark

If you created a parameterized and a copy constructor in the class and you create an object with no arguments (0 arguments), what will be the output?

null

- **A**
  Program will execute successfully
- **B**
  A compile-time will occur
- **C**
  A run-time error will occur
- **D**
  A syntax error will occur

Correct Answer :B

Explanation

A compile time will occur

If we create any constructor in the class, no other constructor will be added. Thus, a compile-time error will occur.

#595 Explained Report Bookmark

If the object is passed by value to a copy constructor?

null

- **A**
  Only public members will be accessible to be copied
- **B**
  That will work normally
- **C**
  Compiler will give out of memory error
- **D**
  Data stored in data members won't be accessible

Correct Answer :C

Explanation

Compiler runs out of memory. This is because while passing the argument by value, a constructor of the object will be called. That in turn called another object constructor for values, and this goes on. This is like a constructor call to itself, and this goes on infinite times, hence it must be passed by reference, so that the constructor is not called.

#596 Explained Report Bookmark

The assignment operator can be used to assign an object to another object of the same type, this assignment by default performed by

null

- **A** Functions
- **B** Constructors
- **C** Memberwise copy
- **D** Bitwise operator

Correct Answer :C

Explanation

The assignment operator (=) can be used to assign an object to another object of the same type. uses default memberwise assignment to assign the data members of Date object

#597 Explained Report Bookmark

Which operator accesses a structure member via the object's variable name?

null

- **A** Dot member selection operator
- **B** Scope resolution operator
- **C** Arrow member selection operator

- **D**
  All of them

Correct Answer :A

Explanation

The dot operator accesses a structure member via the object's variable name or a reference to the object.

#598 Explained Report Bookmark

When the inheritance is private, the private methods in base class are _____ in the derived class (in C++).

null

- **A**
  Inaccessible
- **B**
  Accessible
- **C**
  Protected
- **D**
  Public

Correct Answer :A

Explanation

When the inheritance is private, the private methods in base class are inaccessible in the derived class (in C++).

#599 Explained Report Bookmark

What is meant by multiple inheritance?

null

- **A**
  Deriving a base class from derived class
- **B**
  Deriving a derived class from base class
- **C**
  Deriving a derived class from more than one base class
- **D**
  None of the mentioned

Correct Answer :C

Explanation

Multiple inheritance enables a derived class to inherit members from more than one parent.

#600 Explained Report Bookmark

What will be the order of execution of base class constructors in the following method of inheritance? class a: public b, public c {...};

null

- **A**
  b(); c(); a();
- **B**
  c(); b(); a();
- **C**
  a(); b(); c();
- **D**
  b(); a(); c();

Correct Answer :A

Explanation

b(); c(); a(); the order of execution of base class constructors in the following method of inheritance.class a: public b, public c {...};

#601 Explained Report Bookmark

Which of the following advantages we lose by using multiple inheritance?

null

- **A**
  Dynamic binding
- **B**
  Polymorphism
- **C**
  Both Dynamic binding & Polymorphism
- **D**
  None of the mentioned

Correct Answer :C

Explanation

The benefit of dynamic binding and polymorphism is that they help making the code easier to extend but by multiple inheritance it makes harder to track.

#602 Explained Report Bookmark

What will be the output of this program?

```
#include <iostream>



using namespace std;

class Base {};

class Derived: public Base {};



int main()

{

    Base *p = new Derived;

    Derived *q = new Base;

}
```

- **A**
  error: invalid conversion from "Derived*" to "Base*"
- **B**
  No Compiler Error
- **C**
  error: invalid conversion from "Base*" to "Derived*"
- **D**
  Runtime Error

Correct Answer :C

Explanation

A Base class pointer/reference can point/refer to a derived class object, but the other way is not possible.

#603 Explained Report Bookmark

Which concept is not available in C++?

- **A**
  Virtual constructor
- **B**
  Virtual destructor
- **C**
  Virtual function
- **D**
  Virtual Table

Correct Answer :A

Explanation

There is no concept of virtual constructor in C++ programming.

#604 Explained Report Bookmark

What is the output of following C++ program?

```
class Base{

public:

void f(){
```

```
    cout<<"Base::f()"<<endl;

    }

};

class Derived:public Base{

public:

void f(){

    cout<<"Derived::f()"<<endl; } }; int main(){ Base *d =
new Derived(); d->f();

    return 0;

}
```

- **A**
  Base::f()
- **B**
  Derived::f()
- **C**
  Base::f() Derived::f()
- **D**
  Compiler error

Correct Answer :A

Explanation

If we create an object of derived class, keep its address of it in base class pointer and make a call of the function f() which is in both class, then base class version will be called. If in the base class, we make the f() virtual as “virtual void f()”, then derived class version will be called.

#605 Explained Report Bookmark

Which of the following is an invalid template declaration:

null

- **A**
  template <int x> int func() {return x;}
- **B**
  template <double x> double func() {return x;}
- **C**
  template <typename x> void func(x t) {}
- **D**
  template <int x> float func() {return x;}

Correct Answer :B

Explanation

Valid template declaration is : template < *parameter-list* >

#606 Explained Report Bookmark

What can be passed by non-type template parameters during compile time?

null

- A
  Int
- B
  Float
- C
  Constant expression
- D
  None of the above

Correct Answer :C

Explanation

Non-type template parameters provide the ability to pass a constant expression at compile time. The constant expression may also be an address of a function, object or static class member.

#607 Explained Report Bookmark

A friend class can access _____ members of other class in which it is declared as friend

- A
  private
- B
  protected
- C
  public

- **D**
  Both A and B

Correct Answer :D

Explanation

A friend class can access private and protected members of other class in which it is declared as friend.

#608 Explained Report Bookmark

If class A is a friend of B, then B doesn’t become a friend of A automatically.

null

- **A**
  TRUE
- **B**
  FALSE
- **C**
  Can be true and false
- **D**
  Can not say

Correct Answer :A

Explanation

Friendship is not mutual. If class A is a friend of B, then B doesn’t become a friend of A automatically.

#609 Explained Report Bookmark

Which of the following is correct about friend functions?

null

- **A**
  Friend functions use the dot operator to access members of a class using class objects
- **B**
  Friend functions can be private or public
- **C**
  Friend cannot access the members of the class directly
- **D**
  All of the above

Correct Answer :D

Explanation

Friend function can be declared either in private or public part of the class. A friend function cannot access the members of the class directly. They use the dot membership operator with a member name.

#610 Explained Report Bookmark

Pick the correct statement.

null

- A
  Friend functions are in the scope of a class
- B
  Friend functions can be called using class objects
- C
  Friend functions can be invoked as a normal function
- D
  Friend functions can access only protected members not the private members

Correct Answer :C

Explanation

Friend functions are not in the scope of a class and hence cannot be called through a class object. A friend function can access all types of members of the class. They can be invoked as a normal function.

#611 Explained Report Bookmark

Where does keyword ‘friend’ should be placed?

- A
  function declaration
- B
  function definition
- C
  main function

- **D**
  block function

Correct Answer :A

Explanation

The keyword friend is placed only in the function declaration of the friend function and not in the function definition because it is used toaccess the member of a class.

#612 Explained Report Bookmark

Mutators (setters) are used to set values of _____data members.

- **A**
  Public
- **B**
  Private
- **C**
  Protected
- **D**
  None of the above

Correct Answer :B

Explanation

Mutators (setters) are used to set values of private data members. One of the main goals of a mutator is to check correctness of the value to be set to data member.

#613 Explained Report Bookmark

Which two features of object-oriented programming are the same?

null

- **A**
  Abstraction and Polymorphism features are the same
- **B**
  Inheritance and Encapsulation features are the same
- **C**
  Encapsulation and Polymorphism features are the same
- **D**
  Encapsulation and Abstraction

Correct Answer :D

Explanation

Encapsulation and Abstraction are the same OOPS concepts. Encapsulation hides the features of the object and binds all the properties inside a single class. And abstraction is a feature that shows the required data to the user.

#614 Explained Report Bookmark

Which of the following OOP concept binds the code and data together and keeps them secure from the outside world?

null

- **A**
  Polymorphism
- **B**
  Inheritance
- **C**
  Abstraction

- **D**
  Encapsulation

Correct Answer :D

Explanation

Encapsulation is an important concept of Object-oriented programming. This concept binds the data and methods in a single unit. It binds the methods which manipulate the data.

#615 Explained Report Bookmark

How can the concept of encapsulation be achieved in the program?

null

- **A**
  By using the Access specifiers
- **B**
  By using the concept of Abstraction
- **C**
  By using only private members
- **D**
  By using the concept of Inheritance

Correct Answer :A

Explanation

Users can achieve the concept of encapsulation by implementing the access specifiers in the code. It is not compulsory that the user use only private members.

#616 Explained Report Bookmark

Which operator from the following can be used to illustrate the feature of polymorphism?

null

- **A**
  Overloading <<
- **B**
  Overloading &&
- **C**
  Overloading | |
- **D**
  Overloading +=

Correct Answer :A

Explanation

<< is an insertion operator which is used for overloading (polymorphism).

#617 Explained Report Bookmark

Which among the following cannot be used for the concept of polymorphism?

null

- **A**
  Static member function
- **B**
  Constructor Overloading
- **C**
  Member function overloading
- **D**
  Global member function

Correct Answer :A

Explanation

These functions are not an object property. That's why they cannot be acceptable for overriding or overloading.

#618 Explained Report Bookmark

Which of the following is a mechanism of static polymorphism?

null

- **A**
  Templates
- **B**
  Function overloading
- **C**
  Operator overloading
- **D**
  All of the above

Correct Answer :D

Explanation

Function overloading is a mechanism of static polymorphism.

#619 Explained Report Bookmark

Which operator returns the address of unallocated blocks in memory?

null

- A
  The delete operator
- B
  The empty operator
- C
  The new operator
- D
  All of them

Correct Answer :C

Explanation

The new operator denotes a request for memory allocation on the Free Store.

#620 Explained Report Bookmark

What will be the output of the following program?

```
#include <iostream>

using namespace std;



int main()

{

     int *ival = new int (10);
```

```
    cout<<ival<<endl;



    return 0;

}
```

- **A**
  10
- **B**
  0
- **C**
  Memory address
- **D**
  None of these

Correct Answer :C

Explanation

ival is a pointer variable, and to print the value through a pointer variable, we use asterisk character (*) with the pointer variable name.

Therefore, statement cout<<ival<<endl; will print the memory address.

To print its value, cout<<*ival<<endl; will be used.

#621 Explained Report Bookmark

Which statement is trust about new operator in C++?

1. new is an operator which calls the constructor also
2. new allocates memory at run time and assigns newly allocated memory block's memory too the pointer
3. After allocating the memory new operators returns the pointer of the same type

- **A**
  (1) and (2)
- **B**
  (1) and (3)
- **C**
  (2) and (3)
- **D**
  All - (1), (2) and (3)

Correct Answer :D

Explanation

Alll these statements are true about "new operator in C++".

#622 Explained Report Bookmark

What is the output of the following program?

```
#include <iostream>

using namespace std;



class sample

{
```

```
    public:
     sample()
     {
          cout<<"Hi ";
     }
     ~sample()
     {
          cout<<"Bye ";
     }
};

int main()
{
     sample *obj = new sample();
     delete(obj);
     return 0;
}
```

- **A**
  Hi Bye
- **B**
  Hi
- **C**
  Bye
- **D**
  No output

Correct Answer :A

Explanation

In C++ programming language, operator new is used to create memory at run time (dynamically memory allocation) and delete is used to free/delete dynamically allocated memory.

new calls the constructor and delete calls the destructor.

In this program, sample() is a constrictor which is printing "Hi" and ~sample() is a destructor which is printing "Bye".

Since, we have used both of the statement in the program 1) new operator and 2) delete operator. Therefore, constructor and destructor both will be called and they will print "Hi" and "Bye". Thus, the output will be "Hi Bye".

#623 Explained Report Bookmark

. How to create a dynamic array of pointers (to integers) of size 10 using new in C++?

- **A**
  int *arr = new int *[10];

- **B**
  int **arr = new int *[10];
- **C**
  int *arr = new int [10];
- **D**
  Not Possible

Correct Answer :B

Explanation

Dynamic array of pointer having size 10 using new is created as,

int **arr = new int *[10];

#624 Explained Report Bookmark

What happens when delete is used for a NULL pointer?

int *ptr = NULL;

delete ptr;

- **A**
  Compiler Error
- **B**
  Run-time Crash
- **C**
  No Effect
- **D**
  None of the above

Correct Answer :C

Explanation

Deleting a null pointer has no effect, so it is not necessary to check for a null pointer before calling delete.

#625 Explained Report Bookmark

What is output of below program?

```
int main()

{

  int a=10;

  int b,c;

  b = a++;

  c = a;

  cout<<a<<b<<c;

  return 0;

}
```

- **A**
  111011
- **B**
  111111
- **C**
  101011
- **D**
  101010

Correct Answer :A

Explanation

a = 10

b = a++ => b = 10 a = 11

C = a => c = 11

Therefore, a b c => 111011

#626 Explained Report Bookmark

Static variables are like …… as they are declared in a class declaration and defined in the source file.

- **A**
  inline member function
- **B**
  non-inline member function
- **C**
  static member function
- **D**
  dynamic member function

Correct Answer :B

Explanation

The member variables declared in a class declaration are normally instance variables – a separate copy of them is created for each class object. ... Static variables are like non-inline member functions in that they

are declared in a class declaration and defined in the corresponding source file.

#627 Explained Report Bookmark

…….. member variable is initialized to zero when the first object of its class is created where no other initialization is permitted.

null

- **A**
  friend
- **B**
  static
- **C**
  public
- **D**
  private

Correct Answer :B

Explanation

Static member are by default zero.

#628 Explained Report Bookmark

Which of the following is true?

null

- **A**
  Static methods cannot be overloaded.
- **B**
  Static data members can only be accessed by static methods.

- C
  Non-static data members can be accessed by static methods.
- D
  Static methods can only access static members (data and methods)

Correct Answer :D

Explanation

A static function is a special type of function which is used to access only static data, any other normal data cannot be accessed through static function.

Just like static data, static function is also a class function, it is not associated with any class object.

Static method overloaded and static method can access only static members.

#629 Explained Report Bookmark

Can a Structure contain pointer to itself?

null

- A
  YES
- B
  NO
- C
  Compilation Error
- D
  Runtime Error

Correct Answer :A

Explanation

You cannot declare a structure type that contains itself as a member, but you can declare a structure type that contains a pointer to itself as a member. A structure variable definition contains an optional storage class keyword, the struct keyword, a structure tag, a declarator, and an optional identifier.

#630 Explained Report Bookmark

What is the output of following code?

```
#include<iostream>

using namespace std;



struct student{

 char a; int c;     char b;

};



int main()

{

    cout<<sizeof(student);
```

```
    return 0;

}
```

- **A**
  12
- **B**
  8
- **C**
  10
- **D**
  4

Correct Answer :A

Explanation

Char => 4byte,

int => 4byte,

Char => 4byte

Assuming `sizeof(int) == 4`,

the size of the structure `bar` will be equal to the sum of the sizes of all members together, `sizeof(student) == 12`.