# C:-

## Q.1 What is the difference between the local variable and global variable in C?

Following are the differences between a local variable and global variable:

| Basis for comparison | Local variable | Global variable |
|---|---|---|
| Declaration | A variable which is declared inside function or block is known as a local variable. | A variable which is declared outside function or block is known as a global variable. |
| Scope | The scope of a variable is available within a function in which they are declared. | The scope of a variable is available throughout the program. |
| Access | Variables can be accessed only by those statements inside a function in which they are declared. | Any statement in the entire program can access variables. |
| Life | Life of a variable is created when the function block is entered and destroyed on its exit. | Life of a variable exists until the program is executing. |
| Storage | Variables are stored in a stack unless specified. | The compiler decides the storage location of a variable. |

## Q.2 What is a pointer in C?

A pointer is a variable that refers to the address of a value. It makes the code optimized and makes the performance fast. Whenever a variable is declared inside a program, then the system allocates some memory to a variable. The memory contains some address number. The variables that hold this address number is known as the pointer variable.

**For example:**

```
Data_type *p;
```

The above syntax tells that p is a pointer variable that holds the address number of a given data type value.

## Q.3 What is the usage of the pointer in C?

- **Accessing array elements**: Pointers are used in traversing through an array of integers and strings. The string is an array of characters which is terminated by a null character '\0'.
- **Dynamic memory allocation**: Pointers are used in allocation and deallocation of memory during the execution of a program.
- **Call by Reference**: The pointers are used to pass a reference of a variable to other function.
- **Data Structures like a tree, graph, linked list, etc.**: The pointers are used to construct different data structures like tree, graph, linked list, etc.

## Q.4 What is the difference between malloc() and calloc()?

| | **calloc()** | **malloc()** |
|---|---|---|
| Description | The malloc() function allocates a single block of requested memory. | The calloc() function allocates multiple blocks of requested memory. |
| Initialization | It initializes the content of the memory to zero. | It does not initialize the content of memory, so it carries the garbage value. |
| Number of arguments | It consists of two arguments. | It consists of only one argument. |
| Return value | It returns a pointer pointing to the allocated memory. | It returns a pointer pointing to the allocated memory. |

## Q.5 What is the difference between getch() and getche()?

The **getch()** function reads a single character from the keyboard. It doesn't use any buffer, so entered data will not be displayed on the output screen.

The **getche()** function reads a single character from the keyword, but data is displayed on the output screen. Press Alt+f5 to see the entered character.

# CPP:-

## Q.1 What is the difference between reference and pointer?

Following are the differences between reference and pointer:

| Reference | Pointer |
|---|---|
| Reference behaves like an alias for an existing variable, i.e., it is a temporary variable. | The pointer is a variable which stores the address of a variable. |
| Reference variable does not require any indirection operator to access the value. A reference variable can be used directly to access the value. | Pointer variable requires an indirection operator to access the value of a variable. |
| Once the reference variable is assigned, then it cannot be reassigned with different address values. | The pointer variable is an independent variable means that it can be reassigned to point to different objects. |
| A null value cannot be assigned to the reference variable. | A null value can be assigned to the reference variable. |
| It is necessary to initialize the variable at the time of declaration. | It is not necessary to initialize the variable at the time of declaration. |

## Q.2 Define namespace in C++.

- The namespace is a logical division of the code which is designed to stop the naming conflict.
- The namespace defines the scope where the identifiers such as variables, class, functions are declared.
- The main purpose of using namespace in C++ is to remove the ambiguity. Ambiquity occurs when the different task occurs with the same name.
- For example: if there are two functions exist with the same name such as add(). In order to prevent this ambiguity, the namespace is used. Functions are declared in different namespaces.
- C++ consists of a standard namespace, i.e., std which contains inbuilt classes and functions. So, by using the statement "using namespace std;" includes the namespace "std" in our program.

## Q.3 What is a virtual function?

- A virtual function is used to replace the implementation provided by the base class. The replacement is always called whenever the object in question is actually of the derived class, even if the object is accessed by a base pointer rather than a derived pointer.
- A virtual function is a member function which is present in the base class and redefined by the derived class.
- When we use the same function name in both base and derived class, the function in base class is declared with a keyword virtual.
- When the function is made virtual, then C++ determines at run-time which function is to be called based on the type of the object pointed by the base class pointer. Thus, by making the base class pointer to point different objects, we can execute different versions of the virtual functions.

**Rules of a virtual function:**

- The virtual functions should be a member of some class.

- The virtual function cannot be a static member.
- Virtual functions are called by using the object pointer.
- It can be a friend of another class.
- C++ does not contain virtual constructors but can have a virtual destructor.

## Q.4 What is a pure virtual function?

The pure virtual function is a virtual function which does not contain any definition. The normal function is preceded with a keyword virtual. The pure virtual function ends with 0.

**Syntax of a pure virtual function:**

```
virtual void abc()=0;   //pure virtual function.
```

## Q.5 Define inline function

If a function is inline, the compiler places a copy of the code of that function at each point where the function is called at compile time. One of the important advantages of using an inline function is that it eliminates the function calling overhead of a traditional function.

# Data Structure:-

## Q.1 What are the advantages of Linked List over an array?

- The size of a linked list can be incremented at runtime which is impossible in the case of the array.
- The List is not required to be contiguously present in the main memory, if the contiguous space is not available, the nodes can be stored anywhere in the memory connected through the links.

- The List is dynamically stored in the main memory and grows as per the program demand while the array is statically stored in the main memory, size of which must be declared at compile time.
- The number of elements in the linked list are limited to the available memory space while the number of elements in the array is limited to the size of an array.

## Q.2 **Define the queue data structure &** List some applications of queue data structure.

A queue can be defined as an ordered list which enables insert operations to be performed at one end called REAR and delete operations to be performed at another end called FRONT.

The Applications of the queue is given as follows:

- Queues are widely used as waiting lists for a single shared resource like a printer, disk, CPU.
- Queues are used in the asynchronous transfer of data (where data is not being transferred at the same rate between two processes) for eg. pipes, file IO, sockets.
- Queues are used as buffers in most of the applications like MP3 media player, CD player, etc.
- Queues are used to maintain the playlist in media players to add and remove the songs from the play-list.
- Queues are used in operating systems for handling interrupts.

## **Q.3** What is the difference between NULL and VOID?

- Null is actually a value, whereas Void is a data type identifier.
- A null variable simply indicates an empty value, whereas void is used to identify pointers as having no initial size.

## **Q.4** What are the advantages of Selecetion Sort?

- It is simple and easy to implement.
- It can be used for small data sets.
- It is 60 per cent more efficient than bubble sort.

## Q.5 What are the applications of Graph data structure?

The graph has the following applications:

- Graphs are used in circuit networks where points of connection are drawn as vertices and component wires become the edges of the graph.
- Graphs are used in transport networks where stations are drawn as vertices and routes become the edges of the graph.
- Graphs are used in maps that draw cities/states/regions as vertices and adjacency relations as edges.
- Graphs are used in program flow analysis where procedures or modules are treated as vertices and calls to these procedures are drawn as edges of the graph.

# DBT:-

## Q.1 What is normalization?

Database normalization is a data design & organizational process applied to data structures based on rules that help to build relational databases.

In relational database design, the process of organizing data to minimize redundancy is called normalization.

Normalization usually involves dividing database data into different tables and defining relationships between the tables.

The objective is to isolate data so that additions, deletions, and modifications of a field can be

made in just one table and then retrieved through the rest of the database via the defined relationships.

The key traits for Normalization are eliminating redundant data and ensuring data dependencies.

**Q.2 How is the ACID property related to databases?**

ACID (an acronym for Atomicity Consistency Isolation Durability)

For a reliable database, all four of these attributes should be achieved:

Atomicity is an all-or-none rule for database modifications.

Consistency guarantees that a transaction never leaves your database in a half-finished state.

Isolation keeps transactions separated from each other until they are finished.

Durability guarantees that the database will keep track of pending changes in such a way that

the server can recover from an abnormal termination and committed transactions will not be lost.

**Q.3 What is a Cursor & type of Cursor?**

A cursor is a temporary work area created in the system memory when a SQL statement is executed & we can give a name to it and manipulate multiple records. A cursor contains information on a select statement and the rows of data accessed by it. This temporary work

area is used to store the data retrieved from the database, and manipulate this data. A cursor can hold more than one row, but can process only one row at a time. By default record pointer

set to first record and when record processed then pointer automatically goes to second record. The set of rows the cursor holds is called the active set.

Types of cursors:-

1) Implicit cursors:-These are created by default when DML statements like, INSERT, UPDATE, and DELETE statements are executed. They are also created when a SELECT statement that returns just one row is executed.

2) Explicit cursors:-They must be created when you are executing a SELECT statement that returns more than one row. Even though the cursor stores multiple records, only one record

can be processed at a time, which is called as current row. When you fetch a row the current row position moves to next row.

**Q.4 What is an index?**

An index is a physical structure containing pointers to the data. Indices are created in an existing table to locate rows more quickly and efficiently. It is possible to create an index on one or more columns of a table, and each index is given a name. The users can see the index name but cannot see the indices themselves; they are just used to speed up queries. Effective indices are one of the best ways to improve performance of a database application.

An Index can give you improved query performance because a seek action occurs for retrieving records from your table in a query. A seek means you were able to locate record(s) without having to examine every row to locate those record(s).

A table scan occurs when there is no index available or when a poorly created index exists on the table for a query running against that table. In a table scan, Server examines every row in the table to satisfy the query results.

**Q.5 Difference between Stored Procedure and function?**

Basic Differences:

A Function must return a value but in Stored Procedures it is optional: a procedure can return 0 or n values.

Functions can have only input parameters for it, whereas procedures can have input/output

parameters.

For a Function it is mandatory to take one input parameter, but a Stored Procedure may take 0 to n input parameters.

Functions can be called from a Procedure whereas Procedures cannot be called from a Function.

Advanced Differences:

Exceptions can be handled by try-catch blocks in a Procedure, whereas a try-catch block cannot be used in a Function.

We can go for Transaction Management in a Procedure, whereas in a Function we can't.

In SQL:

A Procedure allows SELECT as well as DML (INSERT, UPDATE, and DELETE) statements in it, whereas Function allows only SELECT statement in it.

Procedures cannot be utilized in a SELECT statement, whereas Functions can be embedded in a

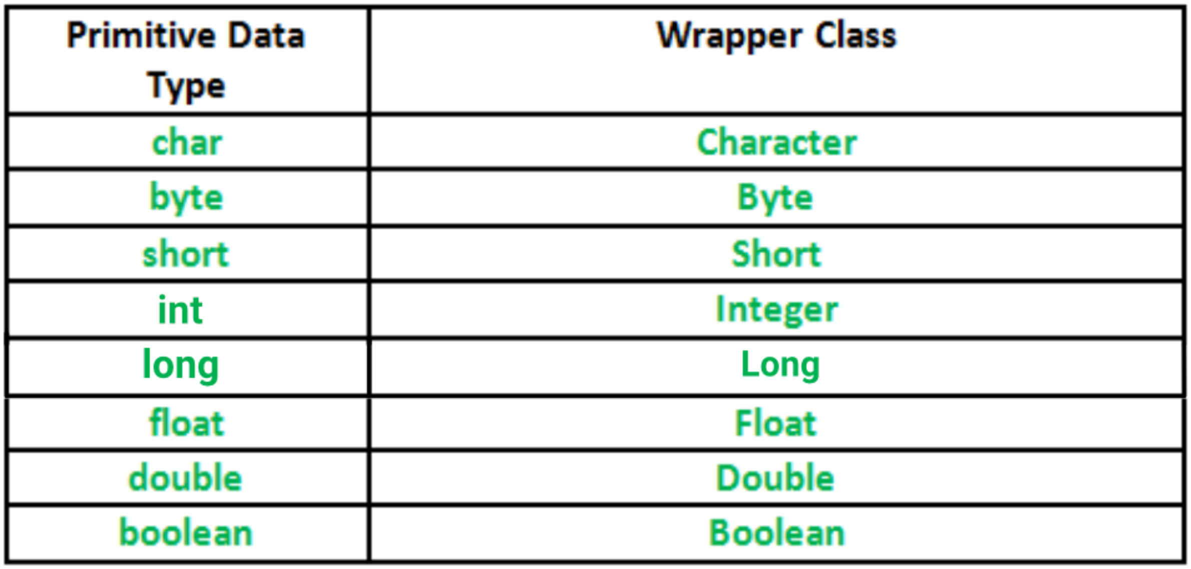
SELECT statement.

Stored Procedures cannot be used in SQL statements anywhere in a WHERE (or a HAVING or a SELECT) block, whereas Functions can.

## Core Java:-

**Q.1 What do you mean by wrapper classes?**
**Answer-** A Wrapper class is a class whose object wraps or contains primitive data types. When we create an object to a wrapper class, it contains a field and in this field, we can store primitive data types. In other words, we can wrap a primitive value into a wrapper class object.

**Primitive Data types and their Corresponding Wrapper class**



| Primitive Data Type | Wrapper Class |
|---|---|
| char | Character |
| byte | Byte |
| short | Short |
| int | Integer |
| long | Long |
| float | Float |
| double | Double |
| boolean | Boolean |

### Q.2 **Explain Static Variable , Static Method, Static Block in java**

- The static variable is a class level variable and it is common to all the class objects i.e. a single copy of the static variable is shared among all the class objects.
- A static method manipulates the static variables in a class. It belongs to the class instead of the class objects and can be invoked without using a class object.
- The static initialization blocks can only initialize the static instance variables. These blocks are only executed once when the class is loaded.

### Q.3 What are the differences between this and super keyword?

There are the following differences between this and super keyword.

- The super keyword always points to the parent class contexts whereas this keyword always points to the current class context.
- The super keyword is primarily used for initializing the base class variables within the derived class constructor whereas this keyword primarily used to differentiate between local and instance variables when passed in the class constructor.
- The super and this must be the first statement inside constructor otherwise the compiler will throw an error.

### Q.4 What are autoboxing and unboxing? When does it occur?

The **autoboxing** is the process of converting primitive data type to the corresponding wrapper class object, eg., int to Integer.

The **unboxing** is the process of converting wrapper class object to primitive data type. For eg., integer to int.

Unboxing and autoboxing occur automatically in Java. However, we can externally convert one into another by using the methods like valueOf() or xxxValue().

It can occur whenever a wrapper class object is expected, and primitive data type is provided or vice versa.

- Adding primitive types into Collection like ArrayList in Java.
- Creating an instance of parameterized classes ,e.g., ThreadLocal which expect Type.
- Java automatically converts primitive to object whenever one is required and another is provided in the method calling.
- When a primitive type is assigned to an object type.

**Q.5 What is the difference between Collection and Collections?**
The differences between the Collection and Collections are given below.

- The Collection is an interface whereas Collections is a class.
- The Collection interface provides the standard functionality of data structure to List, Set, and Queue. However, Collections class is to sort and synchronize the collection elements.
- The Collection interface provides the methods that can be used for data structure whereas Collections class provides the static methods which can be used for various operation on a collection.

## Adv.java:-

**Q.1 What is difference between Get and Post method?**

| Get | Post |
|---|---|
| 1) Limited amount of data can be sent because data is sent in header. | Large amount of data can be sent because data is sent in body. |
| 2) Not Secured because data is exposed in URL bar. | Secured because data is not exposed in URL bar. |
| 3) Can be bookmarked | Cannot be bookmarked |
| 4) Idempotent | Non-Idempotent |
| 5) It is more efficient and used than Post | It is less efficient and used |

Q.2 **What are Design Patterns? Explain the advantages of Java design pattern?**

Design patterns represent the best practices used by experienced object-oriented software developers. Design patterns are solutions to general problems that software developers faced during software development. These solutions were obtained by trial and error by numerous software developers over quite a substantial period of time.

Advantages:

- The Design Patterns are reusable in multiple projects.
- The Design Patterns provide a solution that helps to define the system architecture.
- The Design Patterns capture software engineering experiences.
- The Design Patterns provide transparency to the design of an application.
- They are testified and well-proved since they have been built upon the knowledge and experience of expert software developers.

## Q.3 What is Maven? What are the advantages of Maven?

Maven is a project management tool. It is based on POM (Project Object Model).
It is used for projects build, dependency and documentation.
It simplifies the build process like ANT. But it is too much advanced than ANT.

**Advantages:-**

- No need to add jar file in each project
- Creates right directory structure
- Builds and deploys the project

## Q.4 What is SessionFactory?

SessionFactory provides the instance of Session. It is a factory of Session. It holds the data of second level cache that is not enabled by default.
The SessionFactory is a factory of session and client of ConnectionProvider. It holds second level cache (optional) of data. The org.hibernate.SessionFactory interface provides factory method to get the object of Session.

The SessionFactory is a thread safe object and used by all the threads of an application.
The SessionFactory is heavyweight object so usually it is created during application start up and kept for later use. You would need one SessionFactory object per

database using a separate configuration file. So if you are using multiple databases then you would have to create multiple SessionFactory objects.

**Q.5 What is IOC and DI?**

IOC (Inversion of Control) and DI (Dependency Injection) is a design pattern to provide loose coupling. It removes the dependency from the program.

IOC makes the code loosely coupled. In such case, there is no need to modify the code if our logic is moved to new environment.

In Spring framework, IOC container is responsible to inject the dependency. We provide metadata to the IOC container either by XML file or annotation.

Advantage of Dependency Injection

- makes the code loosely coupled so easy to maintain
- makes the code easy to test

**IOC container is responsible to:**

- create the instance
- configure the instance, and
- assemble the dependencies

# HTML & CSS :-

## Q1. What is the difference between HTML elements and tags?

HTML elements communicate to the browser to render text. When the elements are enclosed by brackets <>, they form HTML tags. Most of the time, tags come in a pair and surround content.

## Q.2 When to use scripts in the head and when to use scripts in the body?

If the scripts contain some event-triggered functions or jquery library then we should use them in the head section. If the script writes the content on the page or is not inside a function then it should be placed inside the body section at the bottom. In short, follow below three points:

1. Place library scripts or event scripts in the head section.
2. Place normal scripts that do not write anything on the page, in the head section until there is any performance issue.
3. Place scripts that render something on the web page at the bottom of the body section.

## **Q.3** What is the difference between class selectors and id selectors?

An overall block is given to class selector while id selectors take only a single element differing from other elements.

### *CSS Class Selector*

```
<style>
.center {
   text-align: center;
   color: blue;
}
</style>

```

### *CSS Id Selector*

```
<style>
#para1 {
   text-align: center;
   color: blue;
}
</style>
```

## **Q.4** What is a CSS selector?

It is a string that identifies the elements to which a particular declaration apply. It is also referred as a link between the HTML document and the style sheet. It is equivalent of HTML elements. There are several different types of selectors in CSS: -

- CSS Element Selector
- CSS Id Selector
- CSS Class Selector
- CSS Universal Selector
- CSS Group Selector

## **Q.5** What are the limitations of CSS?

- Ascending by selectors is not possible
- Limitations of vertical control
- No expressions
- No column declaration
- Pseudo-class not controlled by dynamic behavior
- Rules, styles, targeting specific text not possible

# JavaScript & JQuery:-

## Q.1 What is the difference between == and ===?

The == operator checks equality only whereas === checks equality, and data type, i.e., a value must be of the same type.

## Q.2  What does the isNaN() function?

The isNan() function returns true if the variable value is not a number. For example:

```
function number(num) {
  if (isNaN(num)) {
    return "Not a Number";
  }
  return "Number";
}
console.log(number('1000F'));
// expected output: "Not a Number"

console.log(number('1000'));
// expected output: "Number"
```

## Q.3 What is the difference between View state and Session state?

"View state" is specific to a page in a session whereas "Session state" is specific to a user or browser that can be accessed across all pages in the web application.

## Q.4 What are the selectors in jQuery? How many types of selectors in jQuery?

If you want to work with an element on the web page, first you need to find it. Selectors find the HTML elements in jQuery. There are many types of selectors. Some basic selectors are:

- **Name:** It is used to select all elements which match with the given element Name.
- **#ID:** It is used to select a single element which matches with the given ID
- **.Class:** It is used to select all elements which match with the given Class.
- **Universal (*):** It is used to select all elements available in a DOM.
- **Multiple Elements E, F, G:** It is used to selects the combined results of all the specified selectors E, F or G.
- **Attribute Selector:** It is used to select elements based on its attribute value.

## Q.5 What is the use of css() method in JQuery?

The jQuery CSS() method is used to get (return)or set style properties or values for selected elements. It facilitates you to get one or more style properties. The jQuery CSS() provides two ways:

### *Return a CSS property*

It is used to get the value of a specified CSS property.

# .NET:-

Q.1 What is MSIL?

When the code is compiled, the compiler translates your code into Microsoft intermediate language (MSIL). MSIL is understood by any environment where .Net framework is installed i.e. write once run anywhere. It is platform independent. MSIL is a part of assembly.

The common language runtime includes a JIT compiler for converting this MSIL then to native code.

MSIL contains metadata that is the key to cross language interoperability. Since this metadata is standardized across all .NET languages, a program written in one language can understand the metadata and execute code, written in a different language. MSIL includes instructions for loading, storing, initializing, and calling methods on objects, as well as instructions for arithmetic and logical operations, control flow, direct memory access, exception handling, and other operations. It is also known as CIL(common Intermediate Language).

**Q.2 What is an assembly?**

An assembly is a collection of one or more .exe or dll's. An assembly is the fundamental unit for application development and deployment in the .NET Framework. An assembly contains a collection of types and resources that are built to work together and form a logical unit of functionality. An assembly provides the CLR with the information it needs to be aware of type implementations.

Assemblies can also be private or shared. A private assembly is installed in the installation directory of an application and is accessible to that application only. On the other hand, a shared assembly is shared by multiple applications. A shared assembly has a strong name and is installed in the GAC.

### Q.3 What is Serialization?

Serialization is converting object to stream of bytes. It is a process of bringing an object into a form that it can be written on stream. It's the process of converting the object into a form so that it can be stored on a file, database or memory. It is transferred across the network. Main purpose is to save the state of the object.

The benefits of XML serialization include the following:

Allows for complete and flexible control over the format and schema of the XML produced by serialization.

Serialized format is both human-readable and machine-readable.

Easy to implement. Does not require any custom serialization-related code in the object to be serialized.The XML Schema Definition tool (xsd.exe) can generate an XSD Schema from a set of serializable classes, and generate a set of serializable classes from an XSD Schema, making it easy to programmatically consume and manipulate nearly any XML data in an object-oriented (rather than XML-oriented) fashion.

Objects to be serialized do not need to be explicitly configured for serialization, either by the SerializableAttribute or by implementing the ISerializable interface.

### Q.4 WHY DO WE USE DELEGATE?

There are following reason because of we use delegate.

1. It is used for type safety.

2. For executing multiple methods through one execution.

3. It allows us to pass methods as parameter.

4. For asynchronous programming.

5. For Call back method implementation.

6. It is also used when working with event based programming.

7. When creating anonymous method.

8. When working with lambda expression.

Q.5 What is the difference between Dispose & Finalize?

| BASIS FOR COMPARISON | DISPOSE( ) | FINALIZE( ) |
|---|---|---|
| Defined | The method dispose( ) is defined in the interface IDisposable interface. | The method finalize( ) id defined in java.lang.object class. |
| Syntax | public void Dispose( ){<br>// Dispose code here<br>} | protected void finalize( ){<br>// finalization code here<br>} |

| BASIS FOR COMPARISON | DISPOSE( ) | FINALIZE( ) |
|---|---|---|
| Invoked | The method dispose( ) is invoked by the user. | The method finalize( ) is invoked by the garbage collector. |
| Purpose | Method dispose( ) is used to free unmanaged resources whenever it is invoked. | Method finalize( ) is used to free unmanaged resources before the object is destroyed. |
| Implementation | The method dispose( ) is to be implemented whenever there is a close( ) method. | The method finalize( ) is to be implemented for unmanaged resources. |
| Access specifier | The method dispose( ) is declared as public. | The method finalize( ) is declared as private. |

| BASIS FOR COMPARISON | DISPOSE( ) | FINALIZE( ) |
|---|---|---|
| Action | The method dispose( ) is faster and instantly disposes an object. | The method finalize is slower as compared to dispose |
| Performance | The method disposes( ) performs the instantaneous action hence, does not effect the performance of websites. | The method finalize( ) being slower affects the performance of the websites. |