C INTERVIEW QUESTIONS (WITH ANSWERS)

1. What is C language?

C is a procedural programming language used for developing system and application software. It provides direct access to memory and supports structured programming. It is efficient, portable, and widely used in real-world systems. Many modern languages are derived from C.

2. What are the features of C?

C is simple, efficient, and portable across platforms. It supports structured programming and modular design. It provides low-level memory access using pointers. It also has a rich set of built-in operators and functions.

3. What is a variable?

A variable is a named memory location used to store data. Its value can change during program execution. Each variable has a data type that defines its size and usage. It helps in storing and manipulating data in programs.

4. What are data types in C?

Data types define the type of data a variable can hold. Basic types include int, char, float, and double. Derived types include arrays and pointers. User-defined types include structures, unions, and enums.

5. What is type casting?

Type casting is the process of converting one data type into another. It can be implicit (automatic) or explicit (manual). It is used to ensure compatibility between different data types. It helps in controlling data conversion.

6. What is a constant?

A constant is a fixed value that cannot be changed during program execution. It improves code readability and reliability. Constants can be defined using keywords like const or using #define. They prevent accidental modification.

7. What are tokens in C?

Tokens are the smallest units in a C program. They include keywords, identifiers, constants, operators, and symbols. They form the basic building blocks of a program. The compiler processes tokens during compilation.

8. What is an operator?

An operator is a symbol used to perform operations on variables and values. Examples include arithmetic, relational, logical, and assignment operators. They help in performing calculations and decision-making. Operators are essential for program logic.

9. What is an expression?

An expression is a combination of variables, constants, and operators. It produces a value when evaluated. Expressions are used in assignments, conditions, and loops. They form the core of computations in C.

10. What is control structure?

Control structures determine the flow of program execution. They include selection (if, switch) and looping (for, while). They allow decision-making and repetition of tasks. They are essential for logical program design.

11. What is a pointer?

A pointer is a variable that stores the address of another variable. It allows direct manipulation of memory. Pointers improve efficiency in handling arrays and functions. They are widely used in dynamic memory allocation.

12. What is NULL pointer?

A NULL pointer is a pointer that does not point to any valid memory location. It is used to indicate that a pointer is not initialized. It helps prevent accidental access to random memory. It improves program safety.

13. What is dangling pointer?

A dangling pointer refers to memory that has already been freed. Accessing it leads to undefined behavior. It can cause crashes or unexpected results. It should be avoided by setting pointers to NULL after freeing.

14. What is memory leak?

A memory leak occurs when allocated memory is not released. It wastes system resources and affects performance. It commonly happens in dynamic memory allocation. Proper use of free() prevents memory leaks.

15. What is dynamic memory allocation?

Dynamic memory allocation allows memory to be allocated at runtime. It is done using functions like malloc and calloc. It provides flexibility in managing memory. It is useful when data size is not fixed.

16. Difference between stack and heap?

Stack memory is automatically managed and used for function calls. It is fast but limited in size. Heap memory is manually managed using allocation functions. It is flexible but slower compared to stack.

17. What is function?

A function is a reusable block of code that performs a specific task. It improves modularity and reduces code duplication. Functions make programs easier to read and maintain. They can take inputs and return outputs.

18. What is recursion?

Recursion is a process where a function calls itself. It is used to solve problems in smaller steps. A base condition is required to stop the recursion. It simplifies complex problems like factorial and searching.

19. What is call by value?

In call by value, a copy of the variable is passed to a function. Changes made inside the function do not affect the original variable. It is simple and safe to use. It is the default method in C.

20. What is call by reference?

In call by reference, the address of the variable is passed to a function. Changes made inside the function affect the original variable. It is more efficient for large data. It is implemented using pointers.

21. What is an array?

An array is a collection of elements of the same data type. It stores data in contiguous memory locations. Elements are accessed using an index. Arrays are useful for storing multiple values efficiently.

22. What is a string?

A string is a sequence of characters ending with a null character '\0'. It is stored as a character array.

Strings are used for handling text data. Many library functions are available for string operations.

23. What is structure?

A structure is a user-defined data type that groups different data types. It helps in organizing complex data. Each member has separate memory. It is widely used in real-world applications.

24. What is union?

A union is a user-defined data type where all members share the same memory. Only one member can hold a value at a time. It saves memory space. It is useful in memory-critical applications.

25. Difference between structure and union?

Structures allocate separate memory for each member. Unions share the same memory among members. Structures allow simultaneous access to all members. Unions allow only one active member at a time.

26. What is enum?

An enum is a user-defined data type consisting of named constants. It improves code readability. It assigns integer values automatically. It is useful for representing fixed sets of values.

27. What is typedef?

typedef is used to create a new name for an existing data type. It improves readability and simplifies complex declarations. It is often used with structures and pointers. It helps in writing cleaner code.

28. What is preprocessing?

Preprocessing is the initial stage of compilation. It processes directives like #include and #define. It prepares the source code before compilation. It helps in code reuse and macro expansion

29. What is macro?

A macro is a piece of code defined using #define. It is replaced by the preprocessor before compilation.

It improves performance by avoiding function calls. However, it does not provide type checking.

30. What is file handling?

File handling allows storing and retrieving data from files. It provides permanent storage of data. Functions like fopen, fread, and fwrite are used. It is useful for managing large data.

31. What is segmentation fault?

A segmentation fault occurs when a program accesses invalid memory. It leads to program crashes. It is caused by null pointers or illegal memory access. Proper handling prevents such errors.

32. What is undefined behavior?

Undefined behavior occurs when the program produces unpredictable results. It happens due to incorrect coding practices. The compiler does not guarantee consistent output. It should be avoided carefully.

33. What is const keyword?

The const keyword makes a variable read-only. Its value cannot be changed after initialization. It improves program safety. It also helps in preventing accidental modifications.

34. What is volatile keyword?

The volatile keyword tells the compiler not to optimize a variable. It is used when the value may change unexpectedly. It is common in embedded systems. It ensures correct program behavior.

35. What is function pointer?

A function pointer stores the address of a function. It allows functions to be passed as arguments. It is useful in callbacks and dynamic function calls. It increases flexibility in programs.

36. What is inline function?

An inline function is expanded at compile time. It reduces function call overhead. It improves performance for small functions. It is defined using the inline keyword.

37. What is register variable?

A register variable is stored in CPU registers instead of memory. It provides faster access. It is used for frequently accessed variables. However, the compiler decides its usage.

38. What is extern keyword?

The extern keyword is used to declare global variables defined in another file. It helps in sharing variables across multiple files. It improves modular programming. It avoids multiple definitions.

39. Why is C used in system programming?

C provides direct hardware access and efficient memory management. It is fast and reliable. It allows low-level programming. It is widely used in operating systems and embedded systems.

40. What are advantages of C?

C is fast, portable, and flexible. It allows low-level memory manipulation. It is widely supported and efficient. It is suitable for both system and application development.

C++ INTERVIEW QUESTIONS (WITH ANSWERS)

1. What is C++?

C++ is an extension of the C programming language that supports object-oriented programming. It combines procedural and object-oriented features. It is widely used for system software, game development, and real-time applications. It provides high performance and flexibility.

2. What are the main features of C++?

C++ supports object-oriented concepts like classes, inheritance, and polymorphism. It also provides low-level memory manipulation. It includes features like templates, exception handling, and STL. These make it powerful and versatile.

3. What is Object-Oriented Programming (OOP)?

OOP is a programming paradigm based on objects and classes. It focuses on organizing code using real-world entities. It improves code reusability, scalability, and maintainability. C++ strongly supports OOP concepts.

4. What is a class?

A class is a user-defined data type that acts as a blueprint for objects. It contains variables and functions. It helps in organizing data and behavior together. It supports encapsulation and abstraction.

5. What is an object?

An object is an instance of a class. It represents a real-world entity. It contains data and functions defined by the class. Objects interact with each other in a program.

6. What is encapsulation?

Encapsulation is the process of hiding data and allowing access through methods. It protects data from unauthorized access. It improves data security and modularity. It is implemented using access specifiers.

7. What is abstraction?

Abstraction means hiding implementation details and showing only essential features. It simplifies complex systems. It allows focusing on what an object does rather than how. It is achieved using classes and interfaces.

8. What is inheritance?

Inheritance allows one class to acquire properties of another class. It promotes code reuse and reduces redundancy. It supports hierarchical classification. It is a key feature of OOP.

9. What is polymorphism?

Polymorphism allows functions to behave differently based on context. It enables one interface with multiple implementations. It improves flexibility and scalability. It is achieved through overloading and overriding.

10. What is function overloading?

Function overloading allows multiple functions with the same name but different parameters. It improves readability and code reuse. The compiler decides which function to call. It is a form of compile-time polymorphism.

11. What is operator overloading?

Operator overloading allows operators to work with user-defined types. It enhances code readability. It enables custom behavior for operators. It is implemented using special functions.

12. What is constructor?

A constructor is a special function used to initialize objects. It is called automatically when an object is created. It has the same name as the class. It helps in initializing data members.

13. What is destructor?

A destructor is a special function used to destroy objects. It is called automatically when an object goes out of scope. It releases resources and memory. It has the same name as the class with a tilde (~).

14. What is this pointer?

The this pointer refers to the current object of a class. It is used to access members of the current object. It helps resolve naming conflicts. It is implicitly passed to member functions.

15. What is dynamic memory allocation in C++?

Dynamic memory allocation is done using new and deletes operators. It allows memory allocation at runtime. It provides flexibility in handling data. It must be managed carefully to avoid memory leaks.

16. What is difference between C and C++?

C is procedural, while C++ supports object-oriented programming. C++ includes features like classes and inheritance. C++ provides better abstraction and code reuse. It is more powerful than C.

17. What is namespace?

A namespace is used to organize code and avoid naming conflicts. It allows grouping of related functions and variables. It improves code readability. The std namespace is commonly used.

18. What is STL (Standard Template Library)?

STL is a collection of predefined classes and functions. It includes containers, algorithms, and iterators. It helps in efficient programming. It reduces development time.

19. What are access specifiers?

Access specifiers define the accessibility of class members. They include public, private, and protected. They control data hiding and security. They are essential for encapsulation.

20. What is virtual function?

A virtual function allows runtime polymorphism. It ensures that the correct function is called for an object. It is used with inheritance. It is defined using the virtual keyword.

21. What is pure virtual function?

A pure virtual function is declared but not defined in the base class. It forces derived classes to implement it. It is used to create abstract classes. It is defined using = 0.

22. What is abstract class?

An abstract class contains at least one pure virtual function. It cannot be instantiated. It is used as a base class. It provides a blueprint for derived classes.

23. What is friend function?

A friend function can access private and protected members of a class. It is declared using the friend keyword. It is not a member of the class. It improves flexibility but should be used carefully.

24. What is inline function?

An inline function is expanded at compile time. It reduces function call overhead. It improves performance for small functions. It is defined using the inline keyword.

25. What is reference variable?

A reference variable is an alias for another variable. It provides an alternative name. It must be initialized at declaration. It is used for efficient parameter passing.

26. What is exception handling?

Exception handling manages runtime errors. It uses try, catch, and throw blocks. It prevents program crashes. It improves program reliability.

27. What is file handling in C++?

File handling allows reading and writing data to files. It uses fstream classes. It supports input and output operations. It provides permanent data storage.

28. What is template in C++?

Templates allow writing generic code. They work with any data type. They improve code reusability. They are widely used in STL.

29. What is function overriding?

Function overriding occurs when a derived class provides its own implementation. It replaces the base class function. It requires inheritance and virtual functions. It supports runtime polymorphism.

30. What is multiple inheritance?

Multiple inheritance allows a class to inherit from more than one class. It increases flexibility. It can cause ambiguity issues. It is handled using scope resolution.

31. What is ambiguity in C++?

Ambiguity occurs when the compiler cannot decide which function to use. It happens in multiple inheritance. It can be resolved using scope resolution operator. It should be handled carefully.

32. What is copy constructor?

A copy constructor creates a new object from an existing object. It copies data members. It is used in object initialization. It ensures proper copying of objects.

33. What is shallow copy?

Shallow copy copies only the address of data. It does not create a new memory location. It may cause issues like double deletion. It is not safe for dynamic memory.

34. What is deep copy?

Deep copy creates a new memory location for data. It copies actual values instead of addresses. It prevents memory issues. It is safer than shallow copy.

35. What is operator new and delete?

new is used to allocate memory dynamically. delete is used to free allocated memory. They replace malloc and free in C++. They provide better control and type safety.

36. What is virtual destructor?

A virtual destructor ensures proper destruction of derived class objects. It is important in inheritance. It prevents memory leaks. It is defined using the virtual keyword.

37. What is static member?

A static member belongs to the class, not objects. It is shared among all objects. It is accessed using class name. It is useful for common data.

38. What is const keyword in C++?

const is used to define constant variables. It prevents modification of values. It improves program safety. It is also used with functions and pointers.

39. What is lambda function?

A lambda function is an anonymous function. It is defined inline. It is useful for short operations. It improves code readability.

40. Why is C++ used in real-time systems?

C++ provides high performance and low-level control. It supports object-oriented design. It is efficient for system-level programming. It is widely used in gaming, embedded systems, and applications.

DBMS INTERVIEW QUESTIONS (WITH ANSWERS)

1. What is DBMS?

A DBMS is software used to store, manage, and retrieve data efficiently. It acts as an interface between users and databases. It ensures data consistency, security, and integrity. Examples include MySQL and Oracle.

2. What is a database?

A database is an organized collection of related data. It is designed for easy access, management, and updating. Data is stored in structured formats like tables. It helps in efficient data handling.

3. What is the difference between DBMS and RDBMS?

DBMS stores data in a simple structure, while RDBMS uses tables with relationships. RDBMS supports constraints and normalization. It ensures better data integrity. Examples of RDBMS include MySQL and PostgreSQL.

4. What is a table?

A table is a collection of related data organized in rows and columns. Each row represents a record, and each column represents an attribute. Tables are the basic units of a database. They help in structured data storage.

5. What is a primary key?

A primary key uniquely identifies each record in a table. It cannot contain duplicate or NULL values. It ensures data integrity. It is essential for establishing relationships between tables.

6. What is a foreign key?

A foreign key is a field that links one table to another. It refers to the primary key in another table. It maintains referential integrity. It helps in building relationships between tables.

7. What is normalization?

Normalization is the process of organizing data to reduce redundancy. It divides large tables into smaller related tables. It improves data consistency. It also enhances database efficiency.

8. What are normal forms?

Normal forms are rules used in normalization. They include 1NF, 2NF, 3NF, and BCNF. Each level removes specific types of redundancy. They help in designing efficient databases.

9. What is denormalization?

Denormalization is the process of combining tables to improve performance. It reduces the number of joins. It may introduce redundancy. It is used when fast data retrieval is required.

10. What is SQL?

SQL (Structured Query Language) is used to interact with databases. It allows data retrieval, insertion, updating, and deletion. It is the standard language for relational

11. What are SQL commands?

SQL commands are used to perform operations on databases. They include DDL, DML, DCL, and TCL. Each category has specific functions. They help in managing database structure and data.

12. What is a constraint?

A constraint is a rule applied to a table column. It ensures data accuracy and integrity. Examples include NOT NULL, UNIQUE, and CHECK. Constraints prevent invalid data entry.

13. What is an index?

An index is used to speed up data retrieval. It works like a book index. It improves query performance. However, it may slow down insert and update operations.

14. What is a view?

A view is a virtual table based on a query. It does not store data physically. It provides a customized view of data. It enhances security and simplifies complex queries.

15. What is a join?

A join is used to combine data from multiple tables. It is based on related columns. Types include inner, left, right, and full joins. It helps in retrieving meaningful data.

16. What is transaction?

A transaction is a sequence of database operations. It is treated as a single unit of work. It ensures data consistency. It must follow ACID properties.

17. What are ACID properties?

ACID stands for Atomicity, Consistency, Isolation, and Durability. These properties ensure reliable transactions. They prevent data corruption. They are essential for database integrity.

18. What is concurrency control?

Concurrency control manages multiple users accessing the database. It prevents conflicts and ensures consistency. It uses locking and timestamp techniques. It improves performance in multi-user environments.

19. What is deadlock?

Deadlock occurs when two or more transactions wait for each other. It results in a standstill. It must be detected and resolved. It affects system performance.

20. What is schema?

A schema defines the structure of a database. It includes tables, columns, and relationships. It acts as a blueprint. It helps in database design.

21. What is instance?

An instance is the actual data stored in the database at a given time. It changes frequently. It reflects the current state of the database. It is different from schema.

22. What is data independence?

Data independence allows changes in structure without affecting applications. It improves flexibility. It has two types: logical and physical. It is an important feature of DBMS.

23. What is ER model?

The ER model represents data using entities and relationships. It is used for database design. It uses diagrams for visualization. It helps in understanding data structure.

24. What is entity?

An entity is a real-world object stored in a database. It can be a person, place, or thing. It has attributes. It is represented in tables.

25. What is attribute?

An attribute describes properties of an entity. It represents data fields in a table. Examples include name, age, and ID. It defines characteristics of data.

26. What is relationship?

A relationship defines how entities are connected. It shows associations between tables. Types include one-to-one, one-to-many, and many-to-many. It helps in database modeling.

27. What is cardinality?

Cardinality defines the number of relationships between entities. It specifies how many instances are related. It helps in designing database structure. It ensures proper relationships.

28. What is tuple?

A tuple is a single row in a table. It represents a record. It contains values for each column. It is a basic unit of data.

29. What is domain?

A domain defines the set of valid values for an attribute. It ensures data accuracy. It restricts invalid entries. It is important for data integrity.

30. What is trigger?

A trigger is a procedure that executes automatically. It runs when specific events occur. It is used for validation and auditing. It improves automation.

31. What is stored procedure?

A stored procedure is a precompiled SQL code. It is stored in the database. It improves performance and reusability. It reduces network traffic.

32. What is cursor?

A cursor is used to process rows individually. It allows row-by-row operations. It is useful in complex queries. It provides more control over data processing.

33. What is data redundancy?

Data redundancy means duplication of data. It increases storage and inconsistency. It is minimized using normalization. It affects database efficiency.

34. What is data integrity?

Data integrity ensures accuracy and consistency of data. It prevents invalid data entry. It is maintained using constraints. It is crucial for reliable databases.

35. What is backup and recovery?

Backup is the process of copying data for safety. Recovery restores data after failure. It protects against data loss. It ensures business continuity.

36. What is distributed database?

A distributed database is spread across multiple locations. It improves performance and availability. It allows data sharing across systems. It is used in large organizations.

37. What is NoSQL?

NoSQL databases store non-relational data. They handle large-scale and unstructured data. They are flexible and scalable. Examples include MongoDB.

38. What is OLTP?

OLTP systems handle real-time transactions. They are used in banking and e-commerce. They support fast query processing. They focus on day-to-day operations.

39. What is OLAP?

OLAP systems are used for data analysis. They handle large volumes of data. They support decision-making. They are used in business intelligence.

40. Why DBMS is important?

DBMS ensures efficient data management and security. It reduces redundancy and improves consistency. It supports multi-user access. It is essential for modern applications.