Technical

1. What is the output of the following C code?

```
#include <stdio.h>
int main() {
    int arr[5] = {1, 2, 3, 4, 5};
    int *p = arr;
    printf("%d\n", *(p + 2));
    return 0;
}
```

A) 2

B) 3

C) 4

D) Garbage Value

Answer: B) 3

Explanation: The pointer `p` points to the first element of the array `arr`. The expression `*(p + 2)` dereferences the address of the third element of the array (index 2), which is 3.

2. What does the following C++ program print?

```
#include <iostream>
struct Node {
    int x;
    Node* next;
};
int main() {
    Node* n1 = new Node{10, nullptr};
    Node* n2 = n1;
    n2->x = 20;
    std::cout << n1->x << std::endl;
    return 0;
```

```
}
```

A) 10

B) 20

C) 0

D) Compiler Error

Answer: B) 20

Explanation: `n1` and `n2` are pointers that point to the same `Node` object in memory. Modifying the data through `n2` (i.e., `n2->x = 20`) changes the object's `x` member. Since `n1` points to that same object, accessing `n1->x` will reflect this change.

3. What is the output of the following Java code snippet?

```
import java.util.HashMap;
public class Test {
    public static void main(String[] args) {
        HashMap<String, Integer> map = new HashMap<>();
        map.put("A", 1);
        map.put("A", 2);
        System.out.println(map.get("A"));
    }
}
```

A) 1

B) 2

C) null

D) An exception is thrown

Answer: B) 2

Explanation: In a `HashMap`, if you `put` a key that already exists, the new value will overwrite the old value. In this case, the key "A" is first associated with the value 1, and then it is updated to be associated with the value 2.

4. Consider a table named `Products` with columns `ProductID`, `ProductName`, and `Price`. What does this SQL query do?

```
SELECT COUNT(DISTINCT Price) FROM Products;
```

A) Counts all products.

B) Counts the number of unique product prices.

C) Returns the highest price.

D) Counts the number of products with a price.

Answer: B) Counts the number of unique product prices.

Explanation: `COUNT(Price)` would count all non-NULL prices. The `DISTINCT` keyword causes the `COUNT` function to only consider unique values in the `Price` column.

5. What is the output of the following C code?

```
#include <stdio.h>
int main() {
    char str[] = "Hello";
    str[1] = 'a';
    printf("%s\n", str);
    return 0;
}
```

A) Hello

B) Hallo

C) Hbllo

D) Compilation Error

Answer: B) Hallo

Explanation: A string literal used to initialize a character array is modifiable. `str[1]` accesses the second character ('e') and changes it to 'a'. The `printf` then prints the modified string.

6. What is the output of this C++ code snippet?

```
#include <iostream>
class Parent {
public:
    void show() { std::cout << "Parent "; }
};
class Child : public Parent {
public:
    void show() { std::cout << "Child "; }
};
int main() {
    Parent *p = new Child();
    p->show();
    delete p;
    return 0;
}
```

A) Parent

B) Child

C) Parent Child

D) Undefined Behavior

Answer: A) Parent

Explanation: The `show()` function in the `Parent` class is not declared as `virtual`. Therefore, the call `p->show()` is resolved at compile time based on the type of the pointer (`Parent*`), not the type of the object it points to (`Child`). This is an example of static binding.

7. What is the output of the following Java program?

```
public class Test {
    public static void main(String[] args) {
```

```
        String s1 = "abc";
        String s2 = new String("abc");
        if (s1.equals(s2)) {
            System.out.print("1");
        }
        if (s1 == s2) {
            System.out.print("2");
        }
    }
}
```

A) 1

B) 2

C) 12

D) No output

Answer: A) 1

Explanation: `.equals()` compares the content of the strings, which is identical ("abc"), so it returns true. The `==` operator compares object references. `s1` refers to an object in the string pool, while `s2` refers to a new object created in the heap. Since they are different objects, `s1 == s2` is false.

8. Given a `Students` table with `StudentID` and `Score`, what will this SQL query produce?

SELECT * FROM Students ORDER BY Score DESC LIMIT 1;

A) The student with the lowest score.

B) All students sorted by score.

C) The student with the highest score.

D) The first student entered into the table.

Answer: C) The student with the highest score.

Explanation: `ORDER BY Score DESC` sorts the students in descending order of their scores. `LIMIT 1` then restricts the output to only the first row from the sorted result, which will be the student with the highest score.

9. Predict the output of this C code snippet:

```
#include <stdio.h>
int main() {
    int a = 5, b = 3;
    printf("%d\n", a & b);
    return 0;
}
```

A) 5

B) 3

C) 1

D) 7

Answer: C) 1

Explanation: The `&` operator performs a bitwise AND operation. The binary representation of 5 is `101` and the binary representation of 3 is `011`. Performing a bitwise AND (`101` & `011`) results in `001`, which is 1 in decimal.

10. What is the output of the following C++ code?

```
#include <iostream>
#include <vector>
int main() {
    std::vector<int> v;
    v.push_back(10);
    v.push_back(20);
    std::cout << v.size() << " " << v.capacity() << std::endl;
    return 0;
}
```

A) 2 0

B) 2 1

C) 2 2

D) Output is implementation-defined.

Answer: D) Output is implementation-defined.

Explanation: `size()` will correctly return 2. However, the `capacity()` of a vector is implementation-defined. A common growth strategy is to double the capacity, so it could be 2, but it could also be a different value depending on the standard library implementation.

11. What is printed by the following Java code?

```
public class Test {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder("Hello");
        sb.append(" World");
        sb.reverse();
        System.out.println(sb);
    }
}
```

A) Hello World

B) dlroW olleH

C) World olleH

D) olleH dlroW

Answer: B) dlroW olleH

Explanation: `sb` first becomes "Hello World". The `.reverse()` method reverses the entire character sequence in place, resulting in "dlroW olleH".

12. A table `Orders` has a column `CustomerID`. How do you find which customers have placed more than 10 orders?

```
SELECT CustomerID FROM Orders GROUP BY CustomerID HAVING COUNT(CustomerID) > 10;
```

A) This query is correct.

B) It should use `WHERE COUNT(CustomerID) > 10`.

C) It should not use `GROUP BY`.

D) It should use `WHERE CustomerID > 10`.

Answer: A) This query is correct.

Explanation: To filter groups based on an aggregate function (like `COUNT()`), you must use the `HAVING` clause. The `WHERE` clause is used to filter rows before they are grouped. This query correctly groups orders by `CustomerID` and then filters for those groups having a count greater than 10.

13. What is the output of the following C code?

```
#include <stdio.h>
void func(int *px) {
    *px = 30;
}
int main() {
    int x = 10;
    func(&x);
    printf("%d\n", x);
    return 0;
}
```

A) 10

B) 30

C) 0

D) Garbage Value

Answer: B) 30

Explanation: The address of `x` is passed to `func`. Inside `func`, the pointer `px` is dereferenced, and the value at that address (the value of `x` in `main`) is changed to 30. This is an example of pass-by-reference using pointers.

14. What is the output of this C++ program?

```
#include <iostream>
int main() {
    try {
        throw 'a';
```

```
    } catch (int x) {
        std::cout << "Caught int ";
    } catch (char x) {
        std::cout << "Caught char ";
    } catch (...) {
        std::cout << "Caught generic ";
    }
    return 0;
}
```

A) Caught int

B) Caught char

C) Caught generic

D) No output, program terminates

Answer: B) Caught char

Explanation: An exception of type `char` is thrown. The `catch` blocks are checked in order. The `catch (int x)` block does not match. The `catch (char x)` block is a perfect match for the type of exception thrown, so it is executed.

15. What is the output of the Java code?

```
import java.util.HashSet;
public class Test {
    public static void main(String[] args) {
        HashSet<String> set = new HashSet<>();
        set.add("A");
        set.add("B");
        boolean added = set.add("A");
        System.out.println(set.size() + " " + added);
    }
}
```

A) 2 true

B) 2 false

C) 3 true

D) 3 false

Answer: B) 2 false

Explanation: A `HashSet` only stores unique elements. The second attempt to add "A" will fail because "A" is already in the set. The `.add()` method returns `false` if the element is already present. The size of the set remains 2.

16. Which SQL statement is used to delete a table from a database?

DROP TABLE Customers;

A) `DELETE TABLE Customers;`

B) `TRUNCATE TABLE Customers;`

C) This query is correct.

D) `REMOVE TABLE Customers;`

Answer: C) This query is correct.

Explanation: The `DROP TABLE` statement is used to remove a table definition, its data, indexes, triggers, constraints, and permission specifications. `DELETE` and `TRUNCATE` remove rows from a table but do not remove the table itself.

17. What is the behavior of the following C code?

```
#include <stdio.h>
int main() {
    const int val = 10;
    int *ptr = (int*)&val;
    *ptr = 20;
    printf("%d\n", val);
    return 0;
}
```

A) 10

B) 20

C) Compiler Error

D) Undefined Behavior

Answer: D) Undefined Behavior

Explanation: Attempting to modify a `const`-qualified object through a pointer results in undefined behavior. The compiler might place `val` in read-only memory. While the code may compile with a cast, the runtime behavior is not guaranteed by the C standard. It might crash, print 10, or print 20.

18. What is printed by this C++ code?

```
#include <iostream>
struct Point {
    int x, y;
    Point() : x(0), y(0) {}
    Point(int a, int b) : x(a), y(b) {}
};
int main() {
    Point p1;
    Point p2 = Point(5, 10);
    std::cout << p1.x << " " << p2.y << std::endl;
    return 0;
}
```

A) 0 10

B) 0 0

C) 5 10

D) Compiler Error

Answer: A) 0 10

Explanation: `p1` is created using the default constructor `Point()`, which initializes `x` and `y` to 0. `p2` is created using the parameterized constructor, which initializes its members to 5 and 10. The code then prints `p1.x` (which is 0) and `p2.y` (which is 10).

19. What does this Java code output?

```
interface Animal {
    void sound();
}
class Dog implements Animal {
    public void sound() {
        System.out.println("Woof");
    }
}
public class Test {
    public static void main(String[] args) {
        Animal a = new Dog();
        a.sound();
    }
}
```

A) Woof

B) sound

C) null

D) Compilation Error

Answer: A) Woof

Explanation: This is an example of polymorphism. An object of the `Dog` class is created and assigned to a reference variable of the `Animal` interface type. When `a.sound()` is called, Java uses dynamic method dispatch to call the `sound()` method of the actual object, which is `Dog`, so "Woof" is printed.

20. What is the purpose of an index in a database?

A) To ensure data integrity.

B) To speed up data retrieval operations.

C) To store data in a sorted order.

D) To enforce uniqueness.

Answer: B) To speed up data retrieval operations.

Explanation: An index is a special lookup table that the database search engine can use to speed up data retrieval. Simply put, an index is a pointer to data in a table. It works like the index in the back of a book. While some indexes can enforce uniqueness, their primary purpose is performance.

21. What is the output of the following C code?

```
#include <stdio.h>
int main() {
    int x = 10;
    printf("%d %d %d\n", x, x++, ++x);
    return 0;
}
```

A) 12 11 12

B) 10 10 12

C) 12 10 11

D) Undefined Behavior

Answer: D) Undefined Behavior

Explanation: The order of evaluation of arguments to a function call is not specified by the C standard. The compiler is free to evaluate `x`, `x++`, and `++x` in any order. Because the variable `x` is modified and accessed multiple times without an intervening sequence point, the behavior is undefined.

22. What is the output of the C++ code below?

```
#include <iostream>
class MyClass {
public:
    int val;
    MyClass(int v) : val(v) { std::cout << "C "; }
    MyClass(const MyClass& other) : val(other.val) { std::cout << "CC "; }
};
void func(MyClass obj) {
    std::cout << "F ";
}
```

```
int main() {
    MyClass m(10);
    func(m);
    return 0;
}
```

A) C F

B) C CC F

C) CC F

D) C F CC

Answer: B) C CC F

Explanation: First, the object `m` is created using the constructor, printing "C ". Then, `m` is passed by value to `func`. This creates a copy of `m` for the function's parameter, which calls the copy constructor, printing "CC ". Finally, the body of `func` is executed, printing "F ".

23. What is the result of running this Java code?

```
public class Test {
    static {
        System.out.print("S ");
    }
    public static void main(String[] args) {
        System.out.print("M ");
    }
}
```

A) M S

B) S M

C) S

D) M

Answer: B) S M

Explanation: Static blocks are executed when the class is first loaded into memory by the JVM. This happens before the `main` method is called. Therefore, "S " is printed first, followed by "M " from the `main` method.

24. What does the `LEFT JOIN` keyword do in SQL?

SELECT * FROM Customers LEFT JOIN Orders ON Customers.ID = Orders.CustomerID;

A) Returns all records from the `Orders` table, and the matched records from the `Customers` table.

B) Returns records that have matching values in both tables.

C) Returns all records from the `Customers` table, and the matched records from the `Orders` table.

D) Returns all records from both tables.

Answer: C) Returns all records from the `Customers` table, and the matched records from the `Orders` table.

Explanation: A `LEFT JOIN` returns all rows from the left table (`Customers`) and the matched rows from the right table (`Orders`). If there is no match in the right table, the result is `NULL` on the right side.

25. Predict the output of this C code:

```
#include <stdio.h>
int main() {
    int i = 0;
    for (i = 0; i < 5; i++);
    printf("%d\n", i);
    return 0;
}
```

A) 4

B) 5

C) 0

D) Compiler Error

Answer: B) 5

Explanation: The semicolon at the end of the `for` loop `for (i = 0; i < 5; i++);` makes it an empty loop. The loop will execute 5 times (for i = 0, 1, 2, 3, 4). On the last iteration, `i` becomes 5, the condition `5 < 5` becomes false, and the loop terminates. The `printf` statement then prints the final value of `i`, which is 5.

26. What does this C++ code snippet output?

```
#include <iostream>
int main() {
    const int x = 5;
    // x = 10;
    int *p = (int*)&x;
    *p = 10;
    std::cout << x << " " << *p << std::endl;
    return 0;
}
```

A) 5 10

B) 10 10

C) 5 5

D) Undefined Behavior

Answer: D) Undefined Behavior

Explanation: Attempting to modify a `const` object results in undefined behavior. The compiler might perform optimizations based on the `const` declaration (e.g., replacing all uses of `x` with the literal `5`). The output is not guaranteed and can vary between compilers and optimization levels.

27. What is the output of the Java code?

```
public class Test {
    public static void main(String[] args) {
        final String message;
        message = "Hello";
        // message = "World";
        System.out.println(message);
    }
}
```

A) World

B) Hello

C) null

D) Compilation Error

Answer: B) Hello

Explanation: A `final` local variable is a blank final variable that can be assigned a value only once. Here `message` is assigned "Hello". The commented out line `message = "World";` would cause a compilation error if uncommented, but as it is, the code compiles and runs, printing "Hello".

28. What is the primary purpose of database normalization?

A) To make the database run faster.

B) To reduce data redundancy and improve data integrity.

C) To increase database security.

D) To create a denormalized schema for reporting.

Answer: B) To reduce data redundancy and improve data integrity.

Explanation: Normalization is the process of organizing columns and tables in a relational database to minimize data redundancy. Its main goals are to eliminate undesirable characteristics like Insertion, Update, and Deletion Anomalies, thus improving data integrity.

29. What is the output of the following C code?

```
#include <stdio.h>
int main() {
    printf("%ld\n", sizeof(void*));
    return 0;
}
```

A) 2

B) 4

C) 8

D) Depends on the system architecture (32-bit or 64-bit)

Answer: D) Depends on the system architecture (32-bit or 64-bit)

Explanation: The size of a pointer (`void*` is a generic pointer) depends on the memory architecture of the machine it is compiled for. On a 32-bit system, it will typically be 4 bytes. On a 64-bit system, it will be 8 bytes.

30. What is a smart pointer in C++?

A) A class that wraps a raw pointer to manage its lifetime.

B) A pointer that automatically converts between types.

C) A pointer that can point to member functions.

D) A raw pointer that is faster than normal pointers.

Answer: A) A class that wraps a raw pointer to manage its lifetime.

Explanation: Smart pointers (like `std::unique_ptr` and `std::shared_ptr`) are objects that act like pointers but provide automatic memory management. They automatically deallocate the memory they point to when they go out of scope, helping to prevent memory leaks.

31. What is the output of this Java code snippet?

```
public class Test {
    public static void main(String[] args) {
        try {
            System.out.print("1");
            return;
        } finally {
            System.out.print("2");
        }
    }
}
```

A) 1

B) 2

C) 12

D) 21

Answer: C) 12

Explanation: The `finally` block is always executed, even if a `return` statement is encountered in the `try` block. The `try` block executes, printing "1". Before the `main` method actually returns, the `finally` block is executed, printing "2". Then the method returns.

32. What is the default transaction isolation level in most relational databases like PostgreSQL and SQL Server?

A) READ UNCOMMITTED

B) READ COMMITTED

C) REPEATABLE READ

D) SERIALIZABLE

Answer: B) READ COMMITTED

Explanation: `READ COMMITTED` is a common default isolation level. It guarantees that any data read is committed at the moment it is read. It prevents "dirty reads," where a transaction might read data that has been modified but not yet committed by another transaction.

33. What will be printed by the following C code?

```
#include <stdio.h>
int main() {
    float f = 0.1;
    if (f == 0.1)
        printf("Equal\n");
    else
        printf("Not Equal\n");
    return 0;
}
```

A) Equal

B) Not Equal

C) It depends on the compiler

D) Compilation Error

Answer: C) It depends on the compiler

Explanation: Comparing floating-point numbers for exact equality is problematic due to representation errors. The literal `0.1` is of type `double` by default. The `float` variable `f` may not be able to represent 0.1 exactly. The comparison `f == 0.1` promotes `f` to a `double` and then compares. This comparison might succeed or fail depending on the specific floating-point representation and compiler behavior. The safe way is to check if the absolute difference is within a small tolerance.

34. What is the output of this C++ code?

```
#include <iostream>
struct A {
    A() { std::cout << "A"; }
    ~A() { std::cout << "~A"; }
};
struct B {
    A a;
    B() { std::cout << "B"; }
    ~B() { std::cout << "~B"; }
};
int main() {
    B b;
    return 0;
}
```

A) AB\~A\~B

B) BA\~B\~A

C) AB\~B\~A

D) BA\~A\~B

Answer: C) AB\~B\~A

Explanation: When object `b` of type `B` is constructed, its member variable `a` must be constructed first, printing "A". Then, the constructor of `B` runs, printing "B". Destructors are called in the reverse order of construction. First, the destructor of `B` runs, printing "\~B". Then the destructor of the member variable `a` runs, printing "\~A".

35. In Java, what is the `volatile` keyword used for?

A) To make a variable's value constant.

B) To indicate that a variable's value will be modified by different threads.

C) To prevent a method from being overridden.

D) To mark a variable for garbage collection.

Answer: B) To indicate that a variable's value will be modified by different threads.

Explanation: The `volatile` keyword ensures that reads and writes to a variable are atomic and that any write to a volatile variable is visible to other threads. It prevents the compiler from performing certain optimizations, like caching the variable's value in a register, ensuring that each read fetches the value from main memory.

36. Which SQL clause is used with aggregate functions to filter group results?

A) WHERE

B) LIKE

C) HAVING

D) JOIN

Answer: C) HAVING

Explanation: The `HAVING` clause was added to SQL because the `WHERE` keyword could not be used with aggregate functions like `COUNT()`, `SUM()`, etc. `HAVING` filters the results after the grouping has been done.

37. Predict the output of this C code snippet.

```
#include <stdio.h>
int main() {
    char s[] = "ABC";
    printf("%c\n", s[3]);
    return 0;
}
```

A) C

B) Garbage Value

C) The null character

D) Runtime Error

Answer: C) The null character

Explanation: A string literal in C is terminated by a null character (`\0`). The array `s` is initialized with the characters 'A', 'B', 'C', and `\0`. So its size is 4. `s[3]` accesses the fourth element, which is the null terminator. Printing it as a character may result in no visible output, but the character itself is the null character.

38. What is RAII in C++?

A) A design pattern for exception handling.

B) Resource Acquisition Is Initialization.

C) A memory allocation technique.

D) A way to achieve runtime polymorphism.

Answer: B) Resource Acquisition Is Initialization.

Explanation: RAII is a core C++ programming technique where you bind the life cycle of a resource (like allocated memory, a file handle, or a network socket) to the lifetime of an object. The resource is acquired in the object's constructor and released in its destructor. This helps prevent resource leaks. Smart pointers are a prime example of RAII.

39. What will the following Java code print?

```
import java.util.Optional;
public class Test {
    public static void main(String[] args) {
        Optional<String> name = Optional.empty();
        System.out.println(name.orElse("Default"));
    }
}
```

A) Default

B) null

C) An exception is thrown

D) No output

Answer: A) Default

Explanation: `Optional.empty()` creates an empty `Optional` instance. The `.orElse()` method is used to retrieve the value if one is present, or return a default value if the `Optional` is empty. In this case, since the optional is empty, it returns the string "Default".

40. What is a Foreign Key?

A) A key that uniquely identifies a record in a table.

B) A key used to link two tables together.

C) A key that can be null.

D) The main key of the database.

Answer: B) A key used to link two tables together.

Explanation: A Foreign Key is a field (or collection of fields) in one table that uniquely identifies a row of another table. It is used to establish and enforce a link between the data in two tables.

41. What is the output of the C code?

```
#include <stdio.h>
int main() {
    int x = 5;
    int y = (x++, ++x);
    printf("%d %d\n", x, y);
    return 0;
}
```

A) 7 7

B) 6 7

C) 7 6

D) 6 6

Answer: A) 7 7

Explanation: The comma operator `,` evaluates the left operand (and discards the result) and then evaluates the right operand, and the result of the whole expression is the result of the right operand. In `(x++, ++x)`, first `x++` is evaluated. `x` becomes 6. Then `++x` is evaluated. `x` becomes 7, and the result of this expression is 7. This result (7) is assigned to `y`. Finally, the value of `x` is 7 and the value of `y` is 7.

42. What happens when a reference variable in C++ is not initialized?

A) It points to a null location.

B) It gets a garbage value.

C) It results in a compile-time error.

D) It is automatically initialized to zero.

Answer: C) It results in a compile-time error.

Explanation: References in C++ must be initialized when they are declared. They cannot be null or uninitialized. A reference is an alias for an existing object, so it must be bound to an object at the time of its creation.

43. What is the output of this Java code?

```
public class Test {
    public static void main(String args[]) {
        Integer i1 = 127;
        Integer i2 = 127;
        System.out.println(i1 == i2);
    }
}
```

A) true

B) false

C) Compiler Error

D) Runtime Error

Answer: A) true

Explanation: Java caches `Integer` objects for values in the range of -128 to 127. When `i1` and `i2` are created with the value 127, they both point to the same cached object from the integer pool, so the reference comparison `i1 == i2` returns true.

44. What does the SQL `COALESCE` function do?

A) Converts a value to a different datatype.

B) Returns the first non-NULL value in a list.

C) Concatenates two strings.

D) Returns the current date and time.

Answer: B) Returns the first non-NULL value in a list.

Explanation: The `COALESCE` function accepts a list of arguments and returns the first argument that is not `NULL`. If all arguments are `NULL`, it returns `NULL`. It is useful for providing a default value for a column that might be null.

45. What is a dangling pointer in C?

A) A pointer that has not been initialized.

B) A pointer to a valid memory location.

C) A pointer that points to a memory location that has been deallocated.

D) Another name for a NULL pointer.

Answer: C) A pointer that points to a memory location that has been deallocated.

Explanation: A dangling pointer arises when a pointer continues to point to a memory address after the memory at that location has been freed (deallocated). Dereferencing a dangling pointer leads to undefined behavior.

46. What is the Big O complexity for searching in a balanced Binary Search Tree (BST)?

A) O(n)

B) O(1)

C) O(log n)

D) O(n log n)

Answer: C) O(log n)

Explanation: In a balanced BST, the height of the tree is proportional to log n, where n is the number of nodes. Since the search process involves traversing from the root down to a leaf, the maximum number of comparisons is the height of the tree, resulting in a time complexity of O(log n).

47. In Java, can a `private` method be overridden in a subclass?

A) Yes, but only if the subclass is in the same package.

B) Yes, by using the `@Override` annotation.

C) No, private methods are not inherited and cannot be overridden.

D) Yes, but it will have reduced visibility.

Answer: C) No, private methods are not inherited and cannot be overridden.

Explanation: `private` members are not visible to subclasses, so they are not inherited. Therefore, a subclass cannot override a `private` method from its superclass. A subclass can define a method with the same signature, but this is method hiding, not overriding.

48. What is the difference between `UNION` and `UNION ALL` in SQL?

A) `UNION` removes duplicate rows, while `UNION ALL` does not.

B) `UNION ALL` is faster because it does not remove duplicates.

C) There is no functional difference.

D) Both A and B are correct.

Answer: D) Both A and B are correct.

Explanation: Both `UNION` and `UNION ALL` combine the result sets of two or more `SELECT` statements. The key difference is that `UNION` performs an extra step to remove any duplicate rows from the combined result set. Because `UNION ALL` skips this step, it is generally faster.

49. What is the output of the following C code snippet?

```
#include <stdio.h>
int main() {
    int x = 1, y = 1, z = 1;
    if ((x-- && y++) || z++)
    printf("x=%d, y=%d, z=%d\n", x, y, z);
    return 0;
}
```

A) x=0, y=2, z=1

B) x=0, y=2, z=2

C) x=1, y=1, z=1

D) x=0, y=1, z=1

Answer: A) x=0, y=2, z=1

Explanation: The `&&` operator has higher precedence than `||`. The expression `x--` evaluates to 1 (true) and `x` becomes 0. Because the left side of `&&` is true, the right side `y++` is evaluated. `y++` evaluates to 1 (true) and `y` becomes 2. The result of `(x-- && y++)` is true. Because of short-circuit evaluation, the right side of the `||` operator (`z++`) is not evaluated. `z` remains 1. The `if` condition is true, and the `printf` is executed.

50. Which data structure is typically used to implement a First-In, First-Out (FIFO) queue?

A) Stack

B) Linked List

C) Graph

D) Tree

Answer: B) Linked List

Explanation: A linked list is an excellent choice for implementing a queue. Enqueuing (adding an element) can be done by adding a node to the tail of the list (O(1) if a tail pointer is maintained), and dequeuing (removing an element) can be done by removing a node from the head of the list (O(1)).

51. What will be the final value of `x`?

```
public class Test {
    public static void main(String[] args) {
        int x = 10;
        x += (x >> 1);
        System.out.println(x);
    }
}
```

A) 10

B) 5

C) 15

D) 20

Answer: C) 15

Explanation: The expression `x >> 1` is a bitwise right shift. The binary of 10 is `1010`. Shifting it right by 1 bit results in `0101`, which is 5 in decimal. The statement then becomes `x = x + 5`, which is `10 + 5`, resulting in 15.

52. What is the result of this SQL query, assuming `col1` can contain NULLs?

```
SELECT COUNT(col1) FROM MyTable;
```

A) The total number of rows in `MyTable`.

B) The number of non-NULL values in `col1`.

C) The number of unique values in `col1`.

D) The query will result in an error.

Answer: B) The number of non-NULL values in `col1`.

Explanation: `COUNT(<column_name>)` counts the number of rows where the specified column is not `NULL`. To count all rows in the table regardless of `NULL` values, you should use `COUNT(*)`.

53. What is the output of the C code?

```
#include <stdio.h>
int main() {
    int i = 3;
    int val = i / -2 * 2;
    printf("%d\n", val);
    return 0;
}
```

A) -2

B) -3

C) 2

D) 3

Answer: A) -2

Explanation: In C, integer division truncates towards zero. `i / -2` (i.e., `3 / -2`) results in -1. Then, `-1 * 2` results in -2. The expression is evaluated according to operator precedence, where `/` and `*` have the same precedence and are evaluated left-to-right.

54. In C++, if a class `D` inherits from class `B`, and an object of `D` is created, which constructor is called first?

A) The constructor of class `D`.

B) The constructor of class `B`.

C) It depends on the order of declaration.

D) They are called simultaneously.

Answer: B) The constructor of class `B`.

Explanation: When creating an object of a derived class, the base class constructor is always called first. This ensures that the base part of the object is properly initialized before the derived part is constructed.

55. In Java, what happens if a thread calls the `wait()` method?

A) The thread continues execution after a short pause.

B) The thread releases the lock it holds and goes into a waiting state.

C) The thread is terminated.

D) The thread enters a busy-waiting loop.

Answer: B) The thread releases the lock it holds and goes into a waiting state.

Explanation: A thread must own the monitor (lock) of an object to call `wait()` on it. When `wait()` is called, the thread releases the lock and becomes inactive. It can be woken up by another thread calling `notify()` or `notifyAll()` on the same object.

56. What does `ACID` stand for in the context of database transactions?

A) Atomicity, Consistency, Isolation, Durability

B) Association, Concurrency, Integrity, Durability

C) Atomicity, Concurrency, Isolation, Datagrams

D) Availability, Consistency, Integrity, Durability

Answer: A) Atomicity, Consistency, Isolation, Durability

Explanation: ACID is a set of properties of database transactions intended to guarantee data validity despite errors, power failures, and other mishaps. Atomicity (all or nothing), Consistency (brings DB from one valid state to another), Isolation (concurrent transactions don't affect each other), and Durability (committed changes are permanent).

57. Which data structure is most suitable for implementing a spell checker?

A) Queue

B) Stack

C) Trie (Prefix Tree)

D) Linked List

Answer: C) Trie (Prefix Tree)

Explanation: A Trie is an efficient tree-based data structure for storing strings. It is ideal for spell checkers and auto-complete features because it allows for very fast prefix searches. You can quickly check if a word is in the dictionary or find all words with a given prefix.

58. What is the output of this C++ snippet?

```
#include <iostream>
int main() {
    bool a = true;
    bool b = false;
    int x = 10, y = 5;
    int result = a * x + b * y;
    std::cout << result << std::endl;
    return 0;
}
```

A) 15

B) 5

C) 10

D) Compiler Error

Answer: C) 10

Explanation: In C++, when boolean values are used in an arithmetic context, `true` is converted to the integer `1` and `false` is converted to the integer `0`. Therefore, the expression becomes `1 * 10 + 0 * 5`, which evaluates to `10`.

59. What does the `static` keyword mean when applied to a method in Java?

A) The method can only be called from within the same class.

B) The method belongs to the class itself, not to any specific instance.

C) The method cannot be changed.

D) The method will be loaded into memory only once.

Answer: B) The method belongs to the class itself, not to any specific instance.

Explanation: A static method, also known as a class method, can be called without creating an instance of the class. It is associated with the class, not with any object. It cannot use non-static instance variables or methods directly.

60. Which sorting algorithm has a worst-case time complexity of O(n^2) but an average-case time complexity of O(n log n)?

A) Merge Sort

B) Bubble Sort

C) Insertion Sort

D) Quick Sort

Answer: D) Quick Sort

Explanation: Quick Sort's performance is highly dependent on the choice of the pivot element. In the average case, it performs very well at O(n log n). However, in the worst case (e.g., when the array is already sorted and the pivot is chosen as the first or last element), it degrades to O(n^2). Merge Sort is O(n log n) in all cases, while Bubble and Insertion Sort are O(n^2) in the average and worst cases.