1. Which of the following best describes recursion?
   1. A loop structure that executes a set of statements repeatedly
   2. **A function that calls itself directly or indirectly**
   3. A mechanism to handle exceptions in C++
   4. A data structure used to store function parameters
2. What is the base case in a recursive function?
   1. **The condition that determines whether the recursion continues or stops**
   2. The first case in a switch statement
   3. The final return statement in a function
   4. The default case in a switch statement
3. In a recursive function, what is the process of calling the function again within itself called?
   1. Nesting
   2. Repetition
   3. Looping
   4. **Self-invocation**
4. Which of the following is a requirement for a recursive function to work correctly?
   1. The function must have multiple base cases.
   2. The function must call itself at least twice.
   3. The function must have a loop inside.
   4. **The function must have a termination condition.**
5. What is the role of the call stack in recursion?
   1. It is used to store function names for future use.
   2. It is used to track the number of times a function is called.
   3. It is used to keep track of the memory used by a program.
   4. **It is used to manage the order of function calls and their local variables.**
6. In a recursive function, what happens if the base case is not reached?
   1. The program enters an infinite loop.
   2. The function returns a garbage value.
   3. The program terminates abruptly.
   4. **The function will call itself indefinitely.**
7. The process of a function calling itself directly is known as:
   1. Tail recursion
   2. **Head recursion**
   3. Linear recursion
   4. Indirect recursion
8. What is the maximum depth of recursion that a program can handle?
   1. It depends on the size of the stack.
   2. It is fixed and cannot be changed.
   3. It depends on the size of the heap.
   4. **It is implementation-dependent and may vary.**
9. In C++, which keyword is used to call a recursive function from within the function itself?
   1. repeat
   2. self
   3. redo
   4. **this**
10. In a recursive function, when does the backtracking occur?
    1. Before calling the function itself
    2. **After the function reaches the base case**
    3. During each recursive call
    4. After the function completes execution
11. What is the time complexity of a recursive function with multiple recursive calls?
    1. O(1)
    2. O(n)
    3. O(log n)
    4. **O(2^n)**
12. Which data structure is used to implement the call stack during recursion?
    1. Queue
    2. Linked list
    3. **Stack**
    4. Array
13. In C++, what happens when the maximum recursion depth is exceeded?
    1. The program continues execution with undefined behavior.
    2. **The program will throw a stack overflow exception.**
    3. The program will automatically reduce the recursion depth.
    4. The program will terminate abruptly with an error.
14. What is the default maximum recursion depth in C++?
    1. **It depends on the compiler and platform.**
    2. 1000
    3. 10000
    4. There is no limit to recursion depth.
15. In C++, which part of the call stack stores local variables of a function during recursion?
    1. Global stack
    2. Heap memory
    3. **Local stack frame**
    4. Static memory
16. Which type of recursion involves a function calling itself more than once in a single recursive call?
    1. Indirect recursion
    2. Tail recursion
    3. Linear recursion
    4. **Multiple recursion**
17. In C++, which of the following data structures is implemented using recursion?
    1. Linked list
    2. **Binary search tree**
    3. Array
    4. Queue
18. What is the difference between direct and indirect recursion in C++?
    1. **Direct recursion involves one function calling itself directly, while indirect recursion involves multiple functions calling each other.**
    2. Direct recursion is more efficient than indirect recursion.
    3. Indirect recursion involves one function calling itself directly, while direct recursion involves multiple functions calling each other.
    4. There is no difference; the terms are used interchangeably.
19. In C++, what is the result of calling a recursive function with a negative parameter value?
    1. It will cause a segmentation fault.
    2. **It will cause an infinite loop.**
    3. It will throw a stack overflow exception.
    4. It will trigger the base case immediately.
20. Which type of recursion is more memory-efficient and can be optimized by the compiler?
    1. **Tail recursion**
    2. Head recursion
    3. Indirect recursion
    4. Linear recursion
21. In C++, which method can be used to reduce the space complexity of a recursive function?
    1. Increase the recursion depth.
    2. Use global variables instead of local variables.
    3. Use dynamic memory allocation.
    4. **Optimize the function for tail recursion.**
22. In C++, what is the result of calling a recursive function with a negative base case value?
    1. It will cause a segmentation fault.
    2. It will throw a stack overflow exception.
    3. It will cause an infinite loop.
    4. **It will trigger the base case immediately.**
23. Which type of recursion involves a function calling itself directly at the end of the function?
    1. Indirect recursion
    2. Head recursion
    3. **Tail recursion**
    4. Multiple recursion
24. In C++, which keyword is used to terminate the execution of a recursive function?
    1. end
    2. stop
    3. break
    4. **return**
25. What is the result of calling a recursive function with a very large parameter value in C++?
    1. It will cause a segmentation fault.
    2. **It will throw a stack overflow exception.**
    3. It will cause an infinite loop.
    4. It will execute the function normally.
26. In C++, which type of recursion involves a function calling itself indirectly through other functions?
    1. **Indirect recursion**
    2. Tail recursion
    3. Linear recursion
    4. Direct recursion
27. In C++, what happens if a recursive function forgets to include a base case?
    1. The function will throw a compilation error.
    2. The function will execute normally without any issues.
    3. **The function will go into an infinite loop and cause a stack overflow.**
    4. The function will terminate abruptly with an error.
28. Which type of recursion involves a function calling itself multiple times during a single recursion step?
    1. Tail recursion
    2. **Multiple recursion**
    3. Linear recursion
    4. Indirect recursion
29. In C++, which data structure is used to implement a recursive algorithm to find the factorial of a number?
    1. Linked list
    2. Array
    3. Queue
    4. **Stack**
30. What is the maximum depth of recursion that a C++ program can handle?
    1. It is fixed and cannot be changed.
    2. It depends on the size of the heap.
    3. **It depends on the size of the stack.**
    4. It is implementation-dependent and may vary.
31. In C++, what happens when a recursive function exceeds the maximum recursion depth?
    1. The program will automatically reduce the recursion depth.
    2. **The program will throw a stack overflow exception.**
    3. The program will terminate abruptly with an error.
    4. The program will continue execution with undefined behavior.
32. What is the result of calling a recursive function with a parameter value of 0 in C++?
    1. It will cause a segmentation fault.
    2. It will throw a stack overflow exception.
    3. It will cause an infinite loop.
    4. **It will execute the base case immediately.**
33. Which type of recursion involves a function calling itself directly before executing any other statements?
    1. Indirect recursion
    2. **Head recursion**
    3. Tail recursion
    4. Linear recursion
34. #include <iostream>

using namespace std;

int recursiveSum(int n) {

if (n <= 0)

return 0;

return n + recursiveSum(n - 2);

}

int main() {

cout << recursiveSum(7);

return 0;

}

1. **16**
2. 20
3. 18
4. 15
5. #include <iostream>

using namespace std;

void printPattern(int n) {

if (n <= 0)

return;

cout << n << " ";

printPattern(n - 2);

cout << n << " ";

}

int main() {

printPattern(6);

return 0;

}

a) 6 4 2

b) 6 4 2 4 2 6

**c) 6 4 2 2 4 6**

d) 2 4 6

1. #include <iostream>

using namespace std;

int fun(int a, int b) {

if (b == 0)

return 1;

return base \* power(a, b - 1);

}

int main() {

cout << fun(2, 3);

return 0;

}

1. 2
2. 6
3. **8**
4. 16
5. #include <iostream>

using namespace std;

int fun(int n) {

if (n <= 1)

return n;

return fun(n - 1) + fun(n - 2);

}

int main() {

cout << fun(5);

return 0;

}

1. **5**
2. 8
3. 3
4. 13
5. #include <iostream>

using namespace std;

void fun(int n) {

if (n <= 0)

return;

fun(n / 10);

cout << n % 10 << " ";

}

int main() {

fun(572);

return 0;

}

1. 2 7 5
2. **5 7 2**
3. 7 5 2
4. 5 2 7
5. #include <iostream>

using namespace std;

int gcd(int a, int b) {

if (b == 0)

return a;

return gcd(b, a % b);

}

int main() {

cout << gcd(48, 18);

return 0;

}

a) 48

b) 18

**c) 6**

d) 12

1. In C++, what happens when a recursive function enters an infinite loop due to missing a base case?
   1. The program will automatically terminate the function.
   2. The program will execute normally without any issues.
   3. The program will throw a stack overflow exception.
   4. **The program will go into an infinite loop and cause a stack overflow.**
2. What is the role of the call stack in recursive function calls in C++?
   1. It is used to store function names for future use.
   2. It is used to track the number of times a function is called.
   3. It is used to keep track of the memory used by the program.
   4. **It is used to manage the order of function calls and their local variables.**