1. What is recursion in C++?
   1. A looping mechanism
   2. A method of repeating code
   3. **A function calling itself**
   4. A type of conditional statement
2. Which of the following is an example of a recursive function?
   1. for loop
   2. while loop
   3. switch statement
   4. **factorial function**
3. In recursion, what is the base case?
   1. The last step of the recursive function
   2. The starting point of the recursive function
   3. **The condition that terminates the recursion**
   4. The maximum depth of the recursive function
4. What happens if there is no base case in a recursive function?
   1. **The program will execute forever**
   2. The program will terminate with an error
   3. The program will skip the recursive calls
   4. The program will execute only the base case
5. Which of the following best describes the order of execution in recursion?
   1. **Last In, First Out (LIFO)**
   2. First In, Last Out (FILO)
   3. First In, First Out (FIFO)
   4. Last In, Last Out (LILO)
6. In a recursive function, where is the return value of the base case used?
   1. It is ignored
   2. **It is passed to the calling function**
   3. It is stored in a global variable
   4. It is used as the input for the recursive calls
7. What is the purpose of the recursive step in a recursive function?
   1. To define the base case
   2. To repeat the same code multiple times
   3. To handle the intermediate results
   4. **To break down the problem into smaller subproblems**
8. Which of the following is a disadvantage of using recursion?
   1. It is slower compared to iterative solutions
   2. **It requires more memory than iterative solutions**
   3. It is more difficult to implement than iterative solutions
   4. It cannot solve problems involving loops
9. What is the maximum depth of recursion in C++?
   1. **It depends on the compiler**
   2. It depends on the operating system
   3. It depends on the available memory
   4. It is a fixed value defined by the C++ standard
10. In recursion, what is the role of the call stack?
    1. **To store the return addresses of function calls**
    2. To store the local variables of function calls
    3. To store the intermediate results of function calls
    4. To store the order of execution of function calls
11. What happens when a recursive function is called?
    1. The program jumps to the base case
    2. The program executes the recursive step
    3. **The program pushes the return address onto the stack**
    4. The program pops the return address from the stack
12. Which of the following is true about the order of execution in recursion?
    1. **The recursive step is always executed first**
    2. The base case is always executed first
    3. The order depends on the specific implementation
    4. The order is determined by the compiler
13. What is the output of the following recursive function?

int factorial(int n) {

if (n <= 1)

return 1;

else

return n \* factorial(n - 1);

}

* 1. Compilation error
  2. Runtime error
  3. Stack overflow error
  4. **The factorial of n**

1. What is the output of the following code?

void printNumbers(int n) {

if (n > 0) {

printNumbers(n - 1);

cout << n << " ";

}

}

int main() {

printNumbers(5);

return 0;

}

* 1. 5 4 3 2 1
  2. **1 2 3 4 5**
  3. 1 2 3 4 5 5 4 3 2 1
  4. 5 5 5 5 5

1. What is the output of the following code?

int fibonacci(int n) {

if (n <= 1)

return n;

else

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

}

int main() {

cout << fibonacci(6);

return 0;

}

* 1. 6
  2. **8**
  3. 13
  4. 21

1. What is the output of the following code?

void foo(int n) {

if (n > 0) {

foo(n - 1);

cout << n << " ";

foo(n - 1);

}

}

int main() {

foo(3);

return 0;

}

* 1. 1 2 3 1 1 2 1 3 1
  2. 1 2 3 3 2 1
  3. **1 2 1 3 1 2 1**
  4. 1 1 1 2 3 1 2 1 1 1 2 3

1. What is the output of the following code?

int sumDigits(int n) {

if (n < 10)

return n;

else

return n % 10 + sumDigits(n / 10);

}

int main() {

cout << sumDigits(12345);

return 0;

}

* 1. **15**
  2. 10
  3. 14
  4. 20

1. What is the output of the following code?

int power(int base, int exponent) {

if (exponent == 0)

return 1;

else

return base \* power(base, exponent - 1);

}

int main() {

cout << power(2, 4);

return 0;

}

* 1. 8
  2. **16**
  3. 32
  4. 64

1. What is the output of the following code?

void bar(int n) {

if (n > 0) {

bar(n - 1);

cout << n << " ";

}

}

int main() {

bar(3);

return 0;

}

* 1. **1 2 3**
  2. 3 2 1
  3. 1 1 2 1 3
  4. 1 2 3 3 2 1

1. What is the output of the following code?

int gcd(int a, int b) {

if (b == 0)

return a;

else

return gcd(b, a % b);

}

int main() {

cout << gcd(36, 48);

return 0;

}

* 1. **12**
  2. 24
  3. 36
  4. 48

1. What is the output of the following code?

int countDigits(int n) {

if (n < 10)

return 1;

else

return 1 + countDigits(n / 10);

}

int main() {

cout << countDigits(12345);

return 0;

}

* 1. 1
  2. 2
  3. 3
  4. **5**

1. What is the output of the following code?

void baz(int n) {

if (n > 0) {

baz(n - 1);

cout << n << " ";

baz(n - 1);

}

}

int main() {

baz(2);

return 0;

}

* 1. 1 1 2 1 2 2
  2. **1 2 1**
  3. 1 1 2 2
  4. 1 2 1 2

1. What is the output of the following code?

int sumArray(int arr[], int size) {

if (size == 0)

return 0;

else

return arr[size - 1] + sumArray(arr, size - 1);

}

int main() {

int arr[] = {1, 2, 3, 4, 5};

cout << sumArray(arr, 5);

return 0;

}

* 1. 5
  2. 10
  3. **15**
  4. 20

1. What is the output of the following code?

void printArray(int arr[], int size) {

if (size > 0) {

printArray(arr, size - 1);

cout << arr[size - 1] << " ";

}

}

int main() {

int arr[] = {1, 2, 3, 4, 5};

printArray(arr, 5);

return 0;

}

* 1. 5 4 3 2 1
  2. **1 2 3 4 5**
  3. 1 2 3 4 5 5 4 3 2 1
  4. 5 5 5 5 5

1. What is the output of the following code?

void reverseArray(int arr[], int start, int end) {

if (start >= end)

return;

else {

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

reverseArray(arr, start + 1, end - 1);

}

}

int main() {

int arr[] = {1, 2, 3, 4, 5};

reverseArray(arr, 0, 4);

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

cout << arr[i] << " ";

return 0;

}

* 1. **5 4 3 2 1**
  2. 1 2 3 4 5
  3. 1 2 3 4 5 5 4 3 2 1
  4. 5 5 5 5 5

1. What is the output of the following code?

int findMax(int arr[], int size) {

if (size == 1)

return arr[0];

else {

int max = findMax(arr, size - 1);

return (arr[size - 1] > max) ? arr[size - 1] : max;

}

}

int main() {

int arr[] = {5, 2, 9, 1, 7};

cout << findMax(arr, 5);

return 0;

}

* 1. 1
  2. 2
  3. 5
  4. **9**

1. What is the output of the following code?

void printPattern(int n) {

if (n > 0) {

cout << n << " ";

printPattern(n - 5);

cout << n << " ";

}

}

int main() {

printPattern(20);

return 0;

}

* 1. 20 15 10 5 0
  2. 0 5 10 15 20
  3. **20 15 10 5 5 10 15 20**
  4. 0 5 10 15 20 20 15 10 5 0

1. What is the output of the following code?

int sumRange(int start, int end) {

if (start == end)

return start;

else

return start + sumRange(start + 1, end);

}

int main() {

cout << sumRange(1, 5);

return 0;

}

* 1. 1
  2. 5
  3. **15**
  4. 25

1. What is the output of the following code?

void printBinary(int n) {

if (n > 0) {

printBinary(n / 2);

cout << n % 2;

}

}

int main() {

printBinary(10);

return 0;

}

* 1. **1010**
  2. 0101
  3. 1111
  4. 0000

1. What is the output of the following code?

int countOccurrences(int arr[], int size, int target) {

if (size == 0)

return 0;

else {

if (arr[size - 1] == target)

return 1 + countOccurrences(arr, size - 1, target);

else

return countOccurrences(arr, size - 1, target);

}

}

int main() {

int arr[] = {2, 3, 2, 4, 2, 5};

cout << countOccurrences(arr, 6, 2);

return 0;

}

* 1. 1
  2. 2
  3. **3**
  4. 4

1. What is the output of the following code?

int multiply(int a, int b) {

if (b == 0)

return 0;

else

return a + multiply(a, b - 1);

}

int main() {

cout << multiply(4, 5);

return 0;

}

* 1. 4
  2. 5
  3. **20**
  4. 25

1. What is the output of the following code?

void printFibonacci(int n) {

static int a = 0, b = 1, c;

if (n > 0) {

c = a + b;

a = b;

b = c;

cout << c << " ";

printFibonacci(n - 1);

}

}

int main() {

cout << "0 ";

printFibonacci(5);

return 0;

}

* 1. **0 1 2 3 5 8**
  2. 1 1 2 3 5
  3. 1 2 3 4 5
  4. 0 1 2 3 4 5

1. What is the output of the following code?

int multiplyDigits(int n) {

if (n < 10)

return n;

else

return (n % 10) \* multiplyDigits(n / 10);

}

int main() {

cout << multiplyDigits(12345);

return 0;

}

* 1. 1
  2. **120**
  3. 720
  4. 10000