University of Colombo School of Computing
SCS 1304 - Problem Solving Strategies and Computation Approaches

Lab Sheet 04

Tracing a Recursive Function with a Trace Table & Trace Tree

01. Consider the following recursive function that calculates the factorial of only odd/even numbers

```
def double_factorial(n):
    if n <= 1:
        return 1
    else:
        return n * double_factorial(n - 2)
```

Use a trace table to trace the execution of ***double_factorial****(8).*

Instructions:

I. Create a trace table with columns for the function call, the value of **n**, and the return value.
II. Fill in the table step-by-step as the function is called and returns values.

02. Consider the following recursive function that calculates the sum of the first **n** squares

```
def sum_squares(n):
    if n == 0:
        return 0
    else:
        return n² + sum_squares(n - 1)
```

Draw a trace tree to trace the execution of ***sum_squares****(5).*

Instructions:

I. Start with the root node as ***sum_squares****(5).*
II. Draw branches for each recursive call until you reach the base case.
III. Label each node with the function call and its return value.

03. Consider the following recursive function that calculates the nth term of a sequence defined as:

$T(0)=1$

$T(1)=2$

For $n>1$, $T(n)=T(n-1)+2 \cdot T(n-2)$

```
def sequence(n):
    if n == 0:
        return 1
    elif n == 1:
        return 2
    else:
        return sequence(n-1) + 2 * sequence(n-2)
```

Use both a trace table and a trace tree to trace the execution of ***sequence(7)***

Quizzes.

01. What is the primary property of recursion?

A) Infinite loops

B) Iterative solutions

C) Parallel processing

D) Ability to solve a problem by breaking it down into smaller sub-problems.

02. What is a base case in recursion?

A) The first function call

B) The final function call

C) The simplest, smallest instance of the problem that can't be decomposed further.

D) The largest subproblem

03. What is the main disadvantage of recursion in terms of memory usage?

A) Memory leaks

B) Memory fragmentation

C) Recursive function calls consume memory as each call creates a new stack frame.

D) Memory compression

04. What type of recursion occurs when a function calls another function that eventually calls the original function?

A) Direct recursion

B) Tail recursion

C) Head recursion

D) Indirect recursion.

05. Which type of recursion is defined as a recursive function in which the recursive call is the last statement executed by the function?

A) Head recursion

B) Tail recursion.

C) Tree recursion

D) Nested recursion

06. In the context of recursion, what does the call stack do?

A) Optimizes the recursive calls

B) Manages the active function calls and their local variables. (Correct)

C) Reduces memory usage

D) Increases execution speed

07. What is the execution order in indirect recursion crucial for?

A) Reducing memory usage

B) Enhancing readability

C) To avoid infinite loops or incorrect results.

D) Increasing modularity

08. Why is recursion often preferred for problems that exhibit self-similar subproblems?

A) It is faster

B) It is more memory efficient

C) It provides a straightforward and intuitive solution.

D) It uses fewer resources

09. Which of the following is NOT a type of recursion?

A) Tail recursion

B) Head recursion

C) Tree recursion

D) Modular recursion.

10. What is the key difference between direct and indirect recursion?

A) Direct recursion involves multiple functions

B) Indirect recursion involves a function calling itself

C) Direct recursion involves a function calling itself, while indirect recursion involves a function calling another function that eventually calls the original function.

D) There is no difference

11. Which recursion technique can lead to stack overflow errors if the recursion depth is large?

A) Tail recursion

B) Direct recursion.

C) Indirect recursion

D) None of the above

12. Calculate the output of the following C code for fun(3):

```
int fun(int n) {
    if (n <= 0) return 0;
    return 2 + fun(n - 1);
}
```

A) 3 B) 6

C) 9 D) 12

13. Determine the value of fact(4) using the following recursive function:

```
int fact(int n) {
    if (n == 1) return 1;
    return n * fact(n - 1);
}
```

A) 4

B) 12

C) 24

D) 36

14. What will be the value of sum(4) in the following C function?

```
int sum(int n) {
   if (n == 0) return 0;
   return n + sum(n - 1);
}
```

A) 4 B) 6

C) 10 D) 16

15. Compute the value of fib(5) using the Fibonacci sequence function:

```
int fib(int n) {
    if (n <= 1) return n;
    return fib(n - 1) + fib(n - 2);
}
```

A) 3 B) 5

C) 8 D) 13

16. Find the result of compute(4) for the following C code:

```
int compute(int n) {
    if (n <= 1) return n;
    return n * compute(n - 2);
}
```

A) 6 B) 8

C) 10 D) 12

17. Calculate fun(5) for the following C function:

```
int fun(int n) {
    if (n == 0) return 1;
    return n + fun(n - 1);
}
```

A) 10 B) 12

C) 15 D) 18

18. What is the output of the following C function for power(2, 3)?

```
int power(int x, int y) {
    if (y == 0) return 1;
    return x * power(x, y - 1);
}
```

A) 4

B) 6

C) 8

D) 16

19. A company calculates its profit recursively. If the profit grows by 10% each year and the initial profit is $20000, what is the profit after 4 years?

A) $24000

B) $26000

C) $29282

D) $32000

20. A recursive algorithm is used to compute the total sales of a company. If the sales increase by 3% each year from an initial $15000, what are the sales after 4 years?

A) $15900.00

B) $16450.00

C) $16915.86

D) $17400.00

Reference Link.

Home Work:

Refer this link and try to finish the quizzes