ITMD 413/513 (chap. 12- Recursion) - **solutions**

1. If the problem cannot be solved now, then a recursive function reduces it to a smaller but similar problem and \_\_\_\_\_.

|  |  |
| --- | --- |
| a. | exits |
| b. | returns to the main function |
| c. | returns to the calling function |
| d. | calls itself to solve the smaller problem |

ANS: D

2. What is the first step that needs to be taken in order to apply a recursive approach?

|  |  |
| --- | --- |
| a. | Identify at least one case in which the problem can be solved without recursion. |
| b. | Determine a way to solve the problem in all other circumstances using recursion. |
| c. | Identify a way to stop the recursion. |
| d. | Determine a way to return to the main function. |

ANS: A

3. What is the second step that needs to be taken in order to apply a recursive approach?

|  |  |  |  |
| --- | --- | --- | --- |
| a. | Identify at least one case in which the problem can be solved without recursion. |  |  |
| b. | Determine a way to use recursion in order to solve the problem in all circumstances which cannot be solved without recursion. |  |  |
| c. | Identify a way to stop the recursion. |  |  |
| d. | Determine a way to return to the main function. |  |  |

ANS: B

4. Function A calls function B, which calls function C, which calls function A. This is called \_\_\_\_\_ recursion.

|  |  |
| --- | --- |
| a. | continuous |
| b. | direct |
| c. | three function call |
| d. | indirect |

ANS: D

5. A problem can be solved with recursion if it can be broken down into \_\_\_\_\_ problems.

|  |  |
| --- | --- |
| a. | smaller |
| b. | one-line |
| c. | manageable |
| d. | modular |

ANS: A

6. The base case is a case in which the problem can be solved without \_\_\_\_\_.

|  |  |
| --- | --- |
| a. | loops |
| b. | if |
| c. | objects |
| d. | recursion |

ANS: D

7. What is referred to as the recursive case?

|  |  |  |  |
| --- | --- | --- | --- |
| a. | At least one case in which the problem can be solved without recursion |  |  |
| b. | A way to solve the problem in all other circumstances using recursion |  |  |
| c. | The way to stop the recursion |  |  |
| d. | The way to return to the main function |  |  |

ANS: B

8. What is referred to as the base case?

|  |  |  |  |
| --- | --- | --- | --- |
| a. | At least one case in which the problem can be solved without recursion |  |  |
| b. | The circumstances to solve the problem using recursion |  |  |
| c. | The way to stop the recursion |  |  |
| d. | The way to return to the main function |  |  |

ANS: A

9. If the problem can be solved immediately without recursion, then the recursive function \_\_\_\_\_.

|  |  |
| --- | --- |
| a. | solves it and returns |
| b. | Exits |
| c. | returns the result |
| d. | generates a run-time error |

ANS: A

10. The process of calling a function requires \_\_\_\_\_.

|  |  |
| --- | --- |
| a. | a long memory access |
| b. | a quick memory access |
| c. | several actions to be performed by the computer |
| d. | one action to be performed by the computer |

ANS: C

11. Recursion is \_\_\_\_\_.

|  |  |
| --- | --- |
| a. | never required to solve a problem |
| b. | required to solve mathematical problems |
| c. | never required to solve string problems |
| d. | required to solve some problems |

ANS: A

12. A function is called from the main function for the first time. It then calls itself seven times. What is the depth of recursion?

|  |  |
| --- | --- |
| a. | Eight |
| b. | Two |
| c. | One |
| d. | Seven |

ANS: D

13. What defines the depth of recursion?

|  |  |
| --- | --- |
| a. | The length of the algorithm |
| b. | The numbers of function calls |
| c. | The number of times the function calls itself |
| d. | The number of times it goes back to the main function |

ANS: C

14. Recursive functions are \_\_\_\_\_ iterative algorithms.

|  |  |  |
| --- | --- | --- |
| a. | more efficient than |  |
| b. | less efficient than |  |
| c. | as efficient as |  |
| d. | incomparable to |  |

ANS: B

15. A recursive function includes \_\_\_\_\_ which are not necessary in a loop structure.

|  |  |  |
| --- | --- | --- |
| a. | function calls |  |
| b. | conditional clauses |  |
| c. | overhead actions |  |
| d. | object instances |  |

ANS: C