School of Computing and Information Technologies

PROGCON - CHAPTER 3

CLASS NUMBER: 19 SECTION: AC192

NAME: RIEGO, MA. MONIQUE ISABEL DATE: 11/19/19

PART 1: Identify the following.

GO TO- LESS PROGRAMMING 1. A name to describe structured programming, because structured programmers do not use a "go to" statement.

WHILE...DO(WHILE) LOOP 2. A process continues while some condition continues to be true.

STACKING STRUCTURE 3. Act of attaching structures end to end.

NESTING STRUCTURE 4. Act of placing a structure within another structure.

REPETITION AND ITERATION 5. Alternate names for a loop structure.

IF - THEN - ELSE 6. Another name for a selection structure.

SELECTION STRUCTURE 7. Ask a question and, depending on the answer, take one of two courses of action. Then, no matter which path you follow, continue with the next task.

STRUCTURE 8. Basic unit of programming logic; each structure is a sequence, selection, or loop.

NULL CASE 9. Branch of a decision in which no action is taken.

SEQUENCE STRUCTURE 10. Contains a series of steps executed in order. A sequence can contain any number of tasks, but there is no option to branch off, skipping any of the tasks

LOOP STRUCTURE 11. Continue to repeat actions while a test condition remains true.

DUAL ALTERNATIVE IFS 12. Define one action to be taken when the tested condition is true, and another action to be taken when it is false.

END STRUCTURE STATEMENT 13. Designates the end of a pseudocode structure.

BLOCK 14. Group of statements that executes as a single unit.

UNSTRUCTURED PROGRAM 15. Programs that do not follow the rules of structured logic.

STRUCTURED PROGRAMS 16. Programs that follow the rules of structured logic.

LOOP BODY 17. Set of actions that occur within a loop.

SPAGHETTI CODE 18. Snarled, unstructured program logic.

PRIMING INPUT 19. Statement that reads the first input data record prior to starting a structured loop.

SINGLE ALTERNATIVE IFS. 20. Take action on just one branch of the decision.

RELIABILITY 51. The feature of modular programs that assures you a module has been tested and proven to function correctly.

PASCAL CASING 52. The format for naming variables in which the initial letter is lowercase, multiple-word variable names are run together, and each new word within the variable name begins with an uppercase letter.

CAMEL CASING 53. The format for naming variables in which the initial letter is uppercase, multiple-word variable names are run together, and each new word within the variable name begins with an uppercase letter.

MAIN LINE LOGIC 54. The logic that appears in a program's main module; it calls other modules.

LVALUE 55. The memory address identifier to the left of an assignment operator.

MODULARIZATION 56. The process of breaking down a program into modules.

ABSTRACTION 57. The process of paying attention to important properties while ignoring nonessential details.

CALL A MODULE 58. To use the module's name to invoke it, causing it to execute.

PROGRAM LEVEL 59. Where global variables are declared.

PROGRAM COMMENTS 60. Written explanations that are not part of the program logic but that serve as documentation for those reading the program.

Choose from the following

1. Abstraction
2. Alphanumeric values
3. Annotation symbol
4. Assignment operator
5. Assignment statement
6. Binary operator
7. Call a module
8. Camel casing
9. Data dictionary
10. Data type
11. Declaration
12. Detail loop tasks
13. Echoing input
14. Encapsulation
15. End-of-job tasks
16. External documentation
17. Floating-point
18. Functional cohesion
19. Functional decomposition
20. Garbage
21. Global
22. Hierarchy chart
23. Housekeeping tasks
24. Hungarian notation
25. Identifier
26. In scope
27. Initializing the variable
28. Integer
29. Internal documentation
30. Kebob case
31. Keywords
32. Left-to-right associativity
33. Local
34. Lower camel casing
35. Lvalue
36. Magic number
37. Main program
38. Mainline logic
39. Modularization
40. Module body
41. Module header
42. Module return statement
43. Modules
44. Named constant
45. Numeric
46. Numeric constant (literal numeric constant)
47. Numeric variable
48. Order of operations
49. Overhead
50. Pascal casing
51. Portable
52. Program comments
53. Program level
54. Prompt
55. Real numbers
56. Reliability
57. Reusability
58. Right-associativity and right-to-left associativity
59. Rules of precedence
60. Self-documenting