B.E. COMPUTER SCIENCE AND ENGG. 4th YR 1st SEM. Exam.-2019
SOFTWARE ENGINEERING

Time: Three Hours **Full Marks:100**

GROUP-A

Answer all questions 40×2=80

Choose the unique correct answer.

1. The GUI part of a software system is almost always developed using the
(a) RAD model
(b) Spiral model
(c) Prototyping model
(d) Waterfall model

2. A sequence of evolutionary system integrations or prototypes, each restricted to a well-defined period of time, called a time-box, is employed in
(a) RAD model
(b) Spiral model
(c) Prototyping model
(d) Waterfall model

3. For long projects that may make managers or customers nervous, it is advisable to use the
(a) RAD model
(b) Spiral model
(c) Prototyping model
(d) Waterfall model

4. The Waterfall model may be used when
(a) porting an existing product to a new platform
(b) it is necessary to provide interim deliverables to the customer and users need to get used to the system gradually
(c) the deliverables of a phase can change
(d) there can be dynamic changes in requirements over the life cycle

5. Which of the following is a weakness of the RAD model ?
(a) considerable risk assessment expertise is required
(b) it can fail if reusable components are not available
(c) the customer may want to have the prototype delivered rather than waiting for the full, well-engineered version
(d) it is document-driven, and the amount of documentation can be excessive

6. Which of the following ia a weakness of the Spiral Model ?
(a) it requires a system that can be properly modularized
(b) use of the model may be expensive and even unaffordable-time spent planning, resetting objectives, doing risk analysis, and prototyping may be excessive
(c) it is ineffective for risks identified later during the development cycle
(d) it may not yield systems having optimal performance and reliability

7. The following is an advantage of function-point analysis
(a) it requires subjective evaluations, with much judgment involved
(b) it does not evaluate environmental factors
(c) users can relate more easily to this measure of size. They can more readily understand the impact of a change in functional requirements
(d) it takes advantage of the expertise of several people

8. External inquiries are
(a) things received by the software from outside of the system
(b) specific commands or requests that the software performs, generated from the outside
(c) machine-generated files used by the program
(d) the same as queries

9. External interface files are
(a) machine readable interfaces to other systems
(b) inputs from the user that provide distinct application-oriented data
(c) logical file within the program
(d) error messages

10. Internal logical files are
(a) direct access to a database that retrieve specific data
(b) units of business information input by the user to the software for storage
(c) primary logical group of user data permanently stored entirely within the software system boundary
(d) data stored outside the boundary of the software system being evaluated

11. An air-traffic-control system that must continuously provide accurate, timely positions of aircraft from radar data will have a high value for the following environmental factor
(a) reusability
(b) multiple sites
(c) stringent performance objectives
(d) online data entry

12. The size of an organic type software product is approximately 32,000 LOC. The effort required to develop the software product is
(a) 230 person-months
(b) 145 person-months
(c) 91 person-months
(d) 9 person-months

13. The effort for a project is estimated to be 1000 person-months and the estimated duration is 15 months. The project cost is Rs 200,000,000/-. If the product has to be developed in 12 months, what should be the new cost ?
(a) Rs 250,000,00/-
(b) Rs 488,281,250/-
(c) Rs 390,625,000/-
(d) Rs 312,500,000/-

14. Identifying, estimating, and evaluating risks are activities associated with
(a) Risk planning
(b) Risk control
(c) Risk monitoring
(d) Risk analysis

15. Guiding the risk management effort, integrating it into the overall software life-cycle, and determining when to conduct additional risk analysis are associated with
(a) Risk staffing
(b) Risk directing
(c) Risk identification
(d) Risk control

16. The activity of averting identified risks with greatest importance is associated with
(a) Risk control
(b) Risk planning
(c) Risk monitoring
(d) Risk evaluation

17. Which of the following is NOT a risk analysis technique ?
(a) Sensitivity Analysis
(b) Probability Analysis
(c) Program Evaluation and Review Technique
(d) Utility Theory

18. In CMMI-DEV (Capability Maturity Model Integration for Development), requirements development is a process area included in
(a) Level 2
(b) Level 3
(c) Level 4
(d) none of the above

19. The following is a Key Process Area of Level 2 (Repeatable) of the Capability Maturity Model:
(a) Peer Reviews
(b) Software Quality Management
(c) Software Quality Assurance
(d) Defect Prevention

20. The following requirement is classified as 'Process Control' in the ISO 9001 Requirements:
(a) a quality system must be maintained and documented
(b) purchase material, including bought-in software, must be checked for conforming to requirements
(c) the product must be identifiable at all stages of the process
(d) quality requirement must be identified in a quality plan

21. In PSP, the focus is on
(a) a precise framework for evolving the skills of a software engineer
(b) frameworks for developing software
(c) maturity level of an engineering organization
(d) risk minimization

22. Which of the following is NOT a characteristic of Inspection ?
(a) cost reduction in test and maintenance
(b) early removal of defects
(c) a review of the capabilities of the producer
(d) improved quality delivered to the user.

For Q23-Q28. With reference to the terminology of an SRS, answer the following:

---

23. Software Interfaces are classified under
(a) External Interface Requirements
(b) Performance Requirements
(c) Design Constraints
(d) Quality Characteristics

24. Portability is classified under
(a) Functional Requirements
(b) Performance Requirements
(c) Quality Characteristics
(d) Other Requirements

25. Standards Compliance is classified under
(a) Functional Requirements
(b) Other Requirements
(c) Quality Characteristics
(d) Design Constraints

26. The sizes of tables and files are described under
(a) Design Constraints
(b) Performance Requirements
(c) Quality Characteristics
(d) Other Requirements

27. If an SRS does not specify all the tasks that the user wants to perform, it is
(a) ambiguous
(b) incomplete
(c) inconsistent
(d) incorrect

28. If the requirements are not written in a language and with a vocabulary the user understands, the SRS is not
(a) verifiable
(b) modifiable
(c) traceable
(d) unambiguous

---

29. If the interactions between two modules occur through some shared data, the modules are
(a) tightly coupled
(b) loosely coupled

30. If the different functions of a module execute in a sequence, and the output from one function is input to the next in the sequence, then the module possesses
(a) functional cohesion
(b) sequential cohesion
(c) procedural cohesion
(d) temporal cohesion

31. If all the functions of a module refer to or update the same data structure, the module possesses
(a) temporal cohesion
(b) sequential cohesion
(c) functional cohesion
(d) communicational cohesion

32. The ability of software to run on as many platforms as possible is in accordance with the principle of
(a) Design for portability
(b) Design for flexibility
(c) Reuse existing design
(d) Design for testability

33. Design by contract is in accordance with the principle of
(a) Design for testability
(b) Divide and Conquer
(c) Design defensively
(d) Increase cohesion

34. A subsystem can be divided up into one or more
(a) clients and servers
(b) methods
(c) packages
(d) classes

35. Equivalence class partitioning is a
(a) white box testing technique
(b) compatibility testing technique
(c) black box testing technique
(d) none of the above

36. Boundary value testing for robust software is
(a) same as equivalence partition testing
(b) test boundary condition: on, below, and above the edges of input and output equivalence classes
(c) testing combination of input circumstances
(d) used in white-box testing

37. Errors at the lower modules are detected early in
(a) bottom-up integration
(b) top-down integration
(c) sandwich integration
(d) path-based integration

38. Testing how well a system recovers from crashes, hardware failures or other catastrophic problems is
(a) Security testing
(b) Regression testing
(c) Recovery testing
(d) none of the above

39. Informal software testing that is not based on formal test plans or test cases and testers may be learning the software as they test is referred to as
(a) Exploratory testing
(b) System testing
(c) Ad-hoc testing
(d) none of the above

40. Usability testing is
(a) testing the functionality
(b) testing the speed
(c) testing the ease of use
(d) testing the user documentation

GROUP-B

41. Halstead worked on software metrics. He considered any program to be a collection of tokens, which he classified as either operators or operands. **Operands** were tokens that had a value (e.g. variables and constants). Everything else was considered an **operator** (e.g. commas, parentheses, arithmetic operator, brackets, and so forth).

All tokens that always appear as a pair, triple, and so on will be counted together as one token. For example, a left parenthesis and a right parenthesis will be considered as one occurrence of the token parenthesis. The if-then construction will be considered to have an if-then token.

The count of unique operators in a program is η_1 and the count of unique operands in a program is η_2.

The basic measure of the size of a program is the total count of unique tokens, i.e. $\eta = \eta_1 + \eta_2$. The total count of operators is N_1 and the total count of operands is N_2. The length of the program in tokens is $N = N_1 + N_2$.

The estimate of the actual size of a program in terms of tokens is

$N' = \eta_1 * \log_2 \eta_1 + \eta_2 * \log_2 \eta_2$

Now consider the following program:

```
---------------------------------------------------------
Z = 0;
while X > 0
        Z = Z + Y;
        X = X – 1;
end-while;
print(Z);
---------------------------------------------------------
```

(a) Find η_1 and η_2 . 8

(b) Find N_1 , N_2 , and N. 6

(c) Compute N' . 6

----------END----------