1. An important design activity that many agile methods suggest when building software systems using object-oriented technology is (C), which is a re-organization technique that simplifies the design of components without changing their functionality or behavior.

   A. Refinement B. Design Category C. Reconstruct D. Abstract

2. A software development process describes "who does it", "what to do", "how to do it" and "when to do it". RUP uses (a) to express "who does it". A. Role B. Activity C. Product D. Workflow

> RUP applies four important model elements: roles, activities, artifacts, and workflows. Roles represent "who does it", artifacts represent "what to do", activities represent "how to do it", and workflows represent "when to do it".

3. Software Capability Maturity Model (CMM) divides software capability maturity into five levels from low to high. Currently, achieving CMM level 3 (defined level) is the goal of many organizations, and the core of this level is (B).

   A. Establish basic project management and practices to track project cost, schedule, and functional features B. Build (or integrate) the system using a standard development process (or methodology) C. Management seeks to be more proactive in addressing system development issues D. Continuously monitor and improve the standardized system development process

> This question examines the basic concepts of the Software Maturity Model (CMM):
> Establish basic project management and practices to track project cost, schedule, and functionality at the core of the repeatable level; Build (or integrate) the system using a standard development process (or methodology) to a defined level of core;
>
> Management has sought to be more proactive in responding to system development issues at the core of the management level;
>
> Continuously monitoring and improving the standardized system development process is the core of the optimization level;

4. RUP has a primary goal in each phase and produces some artifacts at the end. At the end of (C), a "software product integrated on an appropriate platform" is produced.

   A. Initiation Phase B. Elaboration Phase C. Build Phase D. Handover Phases This question examines the artifacts produced by each phase in RUP:

> The Inception phase ends with a vision document, an investigation of the use case model, an initial business use case, a
>
> Artifacts such as an early risk assessment and a project plan that shows phases and iterations;
>
> At the end of the Elaboration phase, artifacts such as a supplemental requirements analysis, a software architecture description, and an executable architecture prototype are produced.
>
> The end product at the end of the Construction phase is a product ready for delivery to end users, including software products integrated on appropriate platforms with initial operational capabilities, user manuals, and descriptions of current versions.
>
> At that end of the handover phase, a release version of the product hand over to the us is generated;

5. ccording to the definition of ISO/IEC9126 software quality measurement model, the time and resource quality sub-characteristics of a software belong to (B) quality characteristics.

   A. Functionality B. Efficiency C. Reliability D. Ease of use

> This paper examines the quality characteristics of ISO/IEC9126 software quality metric model. The efficiency quality characteristics include two quality sub-characteristics: time characteristics and resource characteristics.

6. When developing an information system, the main documents used for communication between system developers and project managers are (). A. System Development Contract B. System design specification
   C. System Development Plan D. System test report

> This question examines the role of development documents. The documents that system developers and project managers communicate during the project period mainly include project management documents such as system development plan, system development monthly report and system development summary report.

7. At the end of each phase of software engineering, maintainability should be reviewed. During the review of the system design phase, the system shall be reviewed from

(B) to evaluate the structure and process of the software.

A. Point out portability issues and system interfaces that may affect software maintenance B. Easily modifiable, modular and functionally independent purpose C. Emphasize coding style and internal documentation

D. Testability

This question examines the basic concepts of software review:

Maintainability is a basic characteristic of all software. Software must be maintainable in the development phase. Testability is an evaluation index of maintainability.

- During the review of the system analysis phase, software portability issues and system interfaces that may affect software maintenance should be pointed out;
- During the review of the system design phase, the software structure and process should be evaluated for ease of modification, modularity, and functional independence;
- During the review of the system implementation phase, the code review should emphasize coding style and internal documentation, two factors that affect maintainability.

8. hen the structured method is used for system analysis, according to the principle of decomposition and bstraction and the process of data processing in the system, (B) is used to establish the logical odel of the system, so as to complete the analysis.

A. E-R Figure B. Data Flow Diagram C. Program flow chart D. Software architecture

This question examines the role of graphical tools in the structured analysis method: the data flow diagram gets rid of the physical content of the system and logically describes the function, input, output, and data storage of the system. It is an important part of the logical model of the system.

9. The basic idea of object-oriented development method is to analyze and solve problems as much as possible according to the way human beings understand the objective world. (D) method does not belong to object-oriented method.

A. Booch B . Coad C . OMT D . Jackson

This question examines the object-oriented development method:

Object-oriented development methods include Booch method, Coad method and OMT method.

Jackson method is a data structure-oriented development method.

10. o not consider (A) when determining the number of people required to build a software system

A. Market prospect of the system B. The size of the system

C. Technical complexity of the system D. Project plan

11. A change was made to a project to fix a bug; But when the bug was fixed, it caused errors in code that previously ran correctly. (C) most likely to detect the problem.

A. Unit testing B. Take the test

C. Regression testing D. Installation test

This question examines the knowledge of software testing:

Regression testing is the testing after the software has been changed to find out other errors that may be caused during the change.

12. The risk profile assesses the risk in two ways, the likelihood of the risk occurring and (D)

A. Causes of risk B. Risk

C. Whether the D. Consequences of the

13. The core of CMM Level 4 (Managed Level) is (C). A. Establish basic project management and practices to track project cost, schedule, and functional features B. The organization has a standard software process C. Quantitative understanding and control of both software processes and products D. Advanced new ideas and technologies facilitate continuous process improvement
14. The main purpose of software system design is to make a blueprint for the system. (D) is not the concern of the software design model. A. Overall system structure B. Data Structure C. Interface Model D. Project scope
15. In the ISO/IEC9126 software quality model, the reliability quality characteristic includes many sub-characteristics. After a software failure occurs, it is required to be within 90 Restores its performance and the affected data within seconds. The software property associated with achieving this is the (C) subfeature.
    A. Fault tolerance B. Maturity C. Recoverability D. Easy to operate
16. The system development plan is used for communication between the system developer and the project manager during the project period. It includes (A) and the budget allocation table.

A. PERT diagram B. Master plan C. Test plan D. Development contract

> The documents used for communication between system developers and project managers during the project period mainly include system development plan, including task breakdown table, PERT chart, Gantt chart and budget allocation table, etc. Master planning and development contracts are used to communicate with system analysts during the system planning and system analysis phases;Test plans are used to communicate between system testers and system developers.

17. Correcting errors that have occurred in the software system development phase but have not yet been discovered in the system testing phase belongs to (A) maintenance.

A. Correct B. Adaptability C. Perfection D. Preventive

> The contents of software maintenance generally include accuracy maintenance, adaptability maintenance, perfection maintenance and preventive maintenance. Accuracy maintenance refers to the correction of errors that have occurred in the system development phase but have not yet been found in the system testing phase. Adaptive maintenance refers to the modification of application software to adapt to changes in information technology and management requirements.
> Perfection maintenance mainly refers to adding some functions and performance characteristics that are not specified in the system analysis and design stage to the existing software system.
> Preventive maintenance refers to the active addition of new preventive functions in order to adapt to future changes in the software and hardware environment, so that the application system can adapt to various changes without being eliminated.

18. If a system reuses a third-party component (but its source code is not available), the component should be tested with (D).
    A. Basic path coverage B. Branched cover C. Loop coverage D.

> Black-box testing software testing methods are divided into static testing and dynamic testing:
>
> - Static testing is to test the program by means of manual testing and computer-aided static analysis without running the tested program on the machine.
> - Dynamic testing is to find errors by running the program. Black-box testing and white-box testing can be used in dynamic testing of software products:
>   - The black box testing method tests the external characteristics of the software without considering the internal structure and characteristics of the software.
>   - For third-party components for which the source code is not available, a black box can be used to test the component;

> Basic path coverage, branch coverage, and loop coverage need to be tested according to the internal structure and logic of the program.

19. Extreme Programming (XP) consists of four parts: values, principles, practices, and behaviors. Values include communication, simplicity, (C).

    A. Good Plan B. Continuous release of C. Feedback and Courage D. Continuous integration

> Extreme Programming (XP) is one of the typical methods of agile development. It is a lightweight (agile), efficient, low-risk, flexible,
> A flexible, predictable and scientific approach to software development, which consists of four parts: values, principles, practices and behaviors; The four values are communication, simplicity, feedback and courage.

20. In the following statement about object-oriented analysis, the error is (B). A. Object-oriented analysis focuses on the problem domain and system responsibilities B. Object-oriented analysis needs to consider the testing problem of the system C. Object-oriented analysis ignores issues related to system implementation D. Object-oriented analysis builds an implementation-independent system analysis model

21. In the following statement about object-oriented design, the error is (D).

    A. Higher level modules should not depend on lower level modules B. Abstraction should not depend on details C. The details can depend on the abstraction D. A higher-level module cannot be independent of a lower-level module

22. Component-based software development (CBSD) emphasizes the use of reusable software "components" to design and build software systems, to qualify the required components, (C), and to integrate them into new systems.

    A. Scale Measure B. Data Validation C. Adaptive modification D. Correctness test

23. In the process of developing software with object-oriented method, the process of extracting and sorting out user requirements and establishing an accurate model of the problem domain is called (D). A. Object-Oriented Testing B. Object-oriented implementation
    C. Object-Oriented Design D. Object-oriented analysis

> Using object-oriented software development, there are usually object-oriented analysis, object-oriented design, object-oriented implementation:
>
> Object-Oriented Analysis (OOA) is to acquire the understanding of application problems, and its main task is to extract and sort out user requirements and establish an accurate model of the problem domain.
>
> Object-oriented design is a way of describing software solutions using collaborative objects, object attributes, and methods. It emphasizes defining software objects and how these software objects collaborate to meet requirements, continuing object-oriented analysis.
>
> Object-oriented implementation mainly emphasizes the use of object-oriented programming language to implement the system. Object-oriented testing is to verify the correctness of the system design according to the specification.

24. when using the white-box testing method, the test data shall be determined according to (A) and the specified coverage criteria.

    A. The internal logic of the program B. Complexity of program structure C. Instructions for use D. The function of the program

> White-box testing, also known as structural testing, designs test cases according to the internal structure and logic of the program, tests the execution path and process of the program, and checks whether it meets the design needs. The common techniques of white-box testing involve different coverage standards, and the test data should be determined according to the specified coverage standards.

25. For a large piece of software, uncontrolled changes can quickly lead to chaos. To effectively implement change control, we need to rely on the concept of configuration database and baseline. (C) Not part of the configuration database

    A. Development Library B. Controlled Library C. Information Base D. Product library

> Software change control is an important part of change management. To effectively carry out change control, the concepts of configuration database and baseline are needed. The configuration database generally includes a development library, a controlled library, and a product library.

26. The principles of abstraction, modularization, information hiding and module independence should be followed in software design. When dividing the software system modules, () should be done as far as possible.

A. High cohesion and high coupling B. High cohesion and low coupling

C. Low cohesion and high coupling D. Low cohesion and low coupling

27. Capability Maturity Model Integration (CMMI) is the latest version of CMM, which has two representations: continuous and staged. CMMI based on continuous representation has six (0 ~ 5) capability levels, each of which corresponds to a general goal and a set of general implementation methods and specific methods, among which the capability level (C) focuses on the organizational standardization and deployment of processes.

A . 1 B . 2 C . 3 D . 4

> This question examines the basics of the software capability maturity integration model.
>
> Capability Maturity Model Integration (CMMI) is the latest version of CMM. There are six CMMIs based on continuous expression (0
> 5) Capability level, corresponding to unfinished level, executed level, managed level, defined level, quantified management level and optimized level; Each capability level corresponds to a generic goal and a set of generic implementation methods and specific methods.
>
> - A capability level of 0 is an unexecuted process, indicating that one or more specific objectives of the process area are not being met;
> - Capability Level 1 refers to processes that focus on the completion of specific objectives in the process area by transforming identifiable input work products to produce identifiable output work products;
> - Capability level 2 refers to the ability of a process to be institutionalized as a managed process for a single process instance;
> - Capability level 3 refers to the institutionalization of the process as a defined process, focusing on the organization-level standardization and deployment of the process;
> - Capability level 4 refers to the institutionalization of the process as a quantitative management process;
> - Capability level 5 refers to the institutionalization of the process as an optimized process, indicating that the process is well executed and continuously improved;

28. The Unified Process (UP) defines the Inception, Elaboration, Construction, Handover, and Production phases, each of which ends with the achievement of a milestone, where the milestone of (B) is the lifecycle architecture.

A. Initiation Phase B. Elaboration Phase C. Build Phase D. Handover phase

> The Unified Process (UP) defines the Inception, Elaboration, Construction, Handover, and Production phases, each of which ends at a milestone:
>
> Milestones in the Inception phase are life cycle objectives, milestones in the Elaboration phase are life cycle architecture, milestones in the Build phase are initial operational functions, and milestones in the Handover phase are product releases.

29. C) Activities that are not software configuration management.

A. Change ID B. Change Control C. Quality Control D. Version control

30. The determination of system boundaries and normalization of relationships are performed separately in phase (A) of the database design. A. Requirements Analysis and Logic Design B. Requirements Analysis and Conceptual Design C. Requirements Analysis and Physical Design D. Logical design and conceptual design

31. A project team plans to develop a large-scale system, and has experience in the development of related fields and similar scale systems. Of the following process models, () (B) is the most appropriate for developing this project.

A. Prototype Model B. Waterfall model C. V model D. Spiral model

> Common software life cycle models include waterfall model, evolution model, spiral model, fountain model and so on.
>
> Waterfall model is a model that divides the activities of the software life cycle into several stages connected in a linear order, which is suitable for software projects with clear software requirements.
>
> V model is an evolution model of waterfall model, which associates test and analysis with design, and enhances the verification of analysis and design.
>
> Prototype model is a kind of evolution model, which can quickly build a prototype system that can run, and then improve it according to the user feedback obtained in the process of running.

> The evolution model is especially suitable for the case of lack of accurate understanding of software requirements.
>
> The spiral model combines the waterfall model and the evolutionary model, and adds the risk analysis that both models ignore.

32. Agile development method XP is a lightweight, efficient, low-risk, flexible, predictable and scientific software development method. Its features are included in 12 best practices. Systems are designed to be delivered as early as possible, which is a (C) best practice.

    A. Metaphor B. Reconstruct C. Small Release D. Continuous integration

> Agile development method XP is a lightweight, efficient, low-risk, flexible, predictable and scientific software development method. Its features are included in 12 best practices:
>
> (1)Planning game: Make a plan quickly and improve it as the details change. (2) Small release: the system shall be designed to be delivered as early as possible;
>
> (3)Metaphor: find the right metaphor to convey information;
>
> (4)Simple design: Keep the design simple by dealing only with the current requirements. (5) Test first: write the test code first and then write the program;
>
> (6) Refactoring: revisiting requirements and designs and redefining them explicitly to conform to new and existing requirements; (7) Team programming;
>
> (8)Collective code ownership;
>
> (9)Continuous integration: providing customers with a runnable version on a daily or even hourly basis; (10) Work 40 hours per week;
>
> (11)On-site customers;
>
> (12)Coding Standard

33. hen conducting risk analysis during software development, the purpose of (D) activities is to assist he project team in establishing a strategy to deal with risks. Effective strategies should consider isk avoidance, risk monitoring, risk management, and contingency planning.

    A. Risk Identification B. Risk profile C. Risk Assessment D. Risk control

> Risk analysis is actually four different activities: risk identification, risk prediction, risk assessment and risk control. Risk identification is an attempt to systematically identify threats to the project plan (estimate, schedule, resource allocation);
>
> Risk prediction, also known as risk estimation, assesses a risk from two aspects: the likelihood or probability of the risk occurring; And the consequences if the risk occurs;
>
> The risk assessment predicts whether to affect the reference level value according to the risk, the probability of occurrence and the impact;
>
> The purpose of risk control is to assist the project team to establish a strategy to deal with risks. An effective strategy should consider risk avoidance, risk monitoring, risk management and contingency planning.

34. In the following statement about process improvement, the error is (B).

```
A. The Process Capability Maturity Model is based on the belief that

improving processes will improve products, especially software products

B. The software process improvement framework includes four parts:

assessment, planning, improvement and monitoring.
```

> Software Maturity Model (CMM) is a description of the evolution stages of software organizations. The model has achieved great success in solving the problems existing in the software process, so it has a great impact on the software industry and urges the software industry to pay attention to and take seriously the process improvement work.
>
> The Capability Maturity Model is based on the belief that improving processes will improve products, especially software products.

In order to improve the process capability of software organizations, four aspects are involved in upgrading immature processes to more mature processes, which constitute the framework of software process improvement, namely, process improvement infrastructure, process improvement roadmap, software process evaluation method and software process improvement plan.

After the evaluation, the problems found need to be transformed into the software process improvement plan, and the process improvement is usually not one-time, it needs to be repeated. Each improvement goes through four steps: evaluate, plan, improve, and monitor.

35. he parameters of software complexity measurement do not include (B)

A. The size of the software B. Size of development team

C. The difficulty of the software D. Structure of the software

Software complexity measurement is an important branch of software measurement. There are many parameters to measure software complexity, which mainly include: (1) scale, that is, the number of instructions or the number of source program lines;

(2) The degree of difficulty, usually expressed as a measure of the number of operands present in the program;

(3) A structure, usually expressed in terms of measures related to the structure of a program; (4) Intelligence, that is, the degree of difficulty of the algorithm;

36. Maintainability evaluation index of software system does not include (C).

A. Intelligibility B. Testability C. Scalability D. Modifiable

37. In the following statement about the software system documentation, the error is (C).

A. Software system documents include not only standard documents with certain format requirements, but also non-standard documents formed by various correspondence documents, meeting minutes, accounting documents and other information in the process of system construction.

B. Software system documentation can

improve the visibility of software

38. In the following statement about software testing, the correct one is (A).

A. Software testing can not only show that there are errors in

the software, but also show that there are no errors in the

software. Software testing activities should begin at the

coding stage.

39. What is not part of the black box testing technique is (B).

A. Wrong guess B. Logical override C. Boundary Value Analysis D. Partition of equivalence classes

Black box testing, also known as functional testing, tests the external characteristics of the software without considering the internal structure and characteristics of the software. Commonly used black-box testing techniques include equivalence partitioning, boundary value analysis, error guessing, and causality reporting.

> White-box testing, also known as structural testing, designs test cases according to the internal structure and logic of the program, tests the execution path and process of the program, and checks whether it meets the design needs. Commonly used white-box testing techniques include logic coverage and base path testing.

40. Coupling is a measure of relative independence between modules (how closely they are connected to each other), and the degree of coupling does not depend on (D).

    A. How to call the module B. Complexity of interfaces between modules

    C. Info type D through interface. Number of functions provided by the module

41. In order to effectively capture system requirements, (C) should be used

    A. Waterfall Model B. V Model C. Prototype model D. Spiral model

42. With regard to process improvement, the following statement is incorrect (D).

```
A. Software quality depends on the quality of software

development process, in which personal factors play a leading

role. For process improvement to be effective, process

improvement objectives need to be established
```

43. The reliability of a software product does not depend on (D).

```
A. Number of potential errors B. The location of the potential error

C. How the software product is used   D. The way software products are developed
```

44. Software (A) is the probability that a system will operate without failure within a given time interval and under given conditions.

```
A. Reliability          B. Availability     C. Maintainability      D. Scalability
```

45. The characteristics that a high-quality document should have do not include.

```
A. Targeted, documentation should take into account the target audience

B. Accuracy. The text of the document should be very

precise, and there should be no ambiguous descriptions.

Integrity. Any document should be complete, independent,
```

46. In the software maintenance phase, adding monitoring facilities for the operation of the software belongs to (C) maintenance.

```
A. Corrective             B. Adaptability       C. Perfection    D. Preventive
```

> After the software is developed and delivered to the user, it enters the software operation/maintenance phase. Software maintenance activities can be divided into four types according to their contents:
>
> Corrective maintenance, the process of diagnosing and correcting errors in order to identify and correct software errors, correct defects in software performance, and eliminate misuse in implementation;
>
> Adaptive maintenance, the process of modifying software to adapt to changes in the external or data environment in which it operates due to the rapid development of information technology;

> Perfect maintenance. In the process of using software, users often put forward new functional and performance requirements for the software. In order to meet these requirements, it is necessary to modify or redevelop the software to expand the software function, enhance the software performance, improve the processing efficiency and improve the maintainability of the software.
>
> Preventive maintenance is to improve the maintainability and reliability of the software, and to lay a good foundation for further improvement of the software.

47. If a university bookkeeping system is to be developed using new technology to replace the existing system, (A) should be selected for development.

```
A. Waterfall Model B. Evolution model    C. Spiral model    D. Prototype model
```

48. Isolating each user's data from the data of other users takes into account the (A) quality characteristics of the software.

```
A. Functionality         B. Reliability C. Maintainability    D. Ease of use
```

49. It is the task of stage (B) to determine the module division of the software and the calling relationship between modules.

```
A. Requirements Analysis B. Outline design C. Detailed design D. Coding
```

> Requirement analysis determines the functional and non-functional requirements of the software to be completed; The outline design transforms the requirements into the module division of the software, and determines the calling relationship between the modules. Detailed design of the module will be refined to get detailed data structures and algorithms; Code according to the detailed design of the preparation of the code to run the software, and unit testing.

50. The interface design using the structured analysis model shall be based on (A).

```
A. Data Flow Diagram B. Entity-relationship diagram   C. Data dictionary     D. State-one
transition diagram
```

51. The data flow diagram (DFD) models the functions of the system and the data flow between the functions, where the top-level DFD describes the (B).

```
A. Process     B. Input and output            C. Data storage D. Data entity
```

52. A module a performs several logically similar functions, and the module has (C) cohesion by determining which function the module performs through a parameter.

```
A. Order   B. Process C. Logic   D. Function
```

53. (C) a developer organization without a master programmer group is the least appropriate.

```
A. Develop projects with a small number of people (e.g. 3 ~ 4 people) B. Projects
incorporating new technologies


C. Large-scale projects       D. Less deterministic project
```

54. If the software project team adopts an active risk control method, (A) is the best risk control strategy.

```
A. Risk avoidance    B. Risk monitoring


C. Risk elimination  D. Risk Management and Contingency Plan
```

55. Incremental Model of Software Development (B)

A. Works best when requirements are clearly defined

B. Is a good way to quickly build a

runnable product C. Projects best suited

for large-scale team development

56. In the following description of the data flow diagram, (C) is incorrect.

A. The start or end point of each

data flow must be a process B.

Balance between parent and child

graph must be maintained

57. In the software design phase, the principle of module division is: A module of (A).

A. The scope of action should be within its control. B. The scope of control should be within its scope of action.

C. The scope of action and the scope of control are mutually exclusive D. The scope

58. Defining the risk reference level is a common technique for (C) activities.

A. Risk identification B. Risk prediction C. Risk assessment D. Risk control

59. In the following statement about the document, the incorrect one is (A).

A. The document only describes and specifies the scope of use of the software and related operating commands.

B. Documentation is a part of a software product. Software

without documentation cannot be called a software product C.

The compilation of software documentation occupies a prominent

60. If you want to represent the physical relationship between software components and hardware in the software system to be developed, you usually use (B) in UML.

A. Component diagram B. Deployment diagram C. Class diagramD. Network diagram

61. When designing test cases, the (C) principle should be followed.

A. Determine only the input data for the test case, without considering the output result

B. It is only necessary to check whether the program performs its

proper function, and it is not necessary to consider whether the

program has done redundant functions. We should not only design

62. In unit testing, you do not need to consider (D) when checking the module interface.

```
A. Test whether the input parameters and formal parameters of

the module are consistent in number, attribute and unit.

Whether the definition and usage of global variables in each

module are consistent
```

63. In the following statements about UML deployment diagrams, the correct one is (B). A. Because a message always has some kind of response, the dependencies between deployed components are bidirectional B. The dependencies between deployed components are similar to the package diagram C. Deployment diagrams are not used to describe the physical modules of the code D. Deployment diagrams are not intended to depict the physical distribution of a system across different computer systems