CS 430 - Clicker Questions

Collin Roberts

December 15, 2023

CQ 1: I brought my iClicker today.

A True.
B False.
C I am not sure.

CQ 2: My favourite subject in mathematics is:

A Computer Science.
B Algebra.
C Geometry.
D Combinatorics.
E Statistics.

CQ 3: If a new coding method becomes available which is 10% faster than the current method, then should we adopt it immediately?

A Yes.
B No.
C I am not sure.

CQ 4: Your instructor used to work at

A The Mutual Group.
B Manulife Financial.
C Clarica.
D SunLife Financial.
E All of A–D.

CQ 5: S/W engineering aims to develop software

A on time.
B within budget.
C which is fault-free.
D which meets the client's needs.
E all of A–D.

CQ 6: I have worked in a co-op job where I had to track my time for the organization.

A True.
B False.
C I am not sure.

CQ 7: I have worked in a co-op job where my team used the Waterfall life-cycle model.

A True.
B False.
C I am not sure.

CQ 8: I have worked in a co-op job where my team used the Object-Oriented paradigm.

A True.
B False.
C I am not sure.

CQ 9: The Classical life cycle model handles

A Data and not operations on the data.

B Data and operations on the data together, equally.

C Operations on the data and not the data.

D A or C.

E All of A–C.

CQ 10: The Object-oriented life cycle model handles

A Data and not operations on the data.

B Data and operations on the data together, equally.

C Operations on the data and not the data.

D A or C.

E All of A–C.

CQ 11: I think that the Winburg Case Study is

A worse than reality.

B similar to reality.

C better than reality.

D I am not sure.

CQ 12: The Winburg Case Study is similar to experiences I have had while working in IT.

A True.
B False.
C I am not sure.

CQ 13: Updating a system to comply with new regulatory requirements is an example of

A Corrective maintenance.
B Perfective maintenance.
C Adaptive maintenance.
D I am not sure.

CQ 14: Updating a system to upgrade one of its screens to add a minor feature is an example of

A Corrective maintenance.
B Perfective maintenance.
C Adaptive maintenance.
D I am not sure.

CQ 15: Updating a system to fix a bug on one of its screens is an example of

A Corrective maintenance.
B Perfective maintenance.
C Adaptive maintenance.
D I am not sure.

CQ 16: The cost of software development is the most important consideration in S/W engineering.

A True.
B False.
C I am not sure.

CQ 17: There is no Testing phase in the Classical life cycle model because

A Testing is not important.

B Testing should be performed throughout the life cycle.

C Testing is too costly.

D All of A–C.

E None of A–C.

CQ 18: Running production code and the modified code against a set of old test cases and confirming there are no differences is an example of

A the **moving target problem**.

B **feature / scope creep**.

C a **regression test**.

D a **regression fault**.

E **Miller's Law**.

CQ 19: The fact that no more than seven requirements can be refined simultaneously is an example of

A the **moving target problem**.
B **feature / scope creep**.
C a **regression test**.
D a **regression fault**.
E **Miller's Law**.

CQ 20: A user request to add a new error check on a screen, which was not included in the original specification is an example of

A the **moving target problem**.

B **feature / scope creep**.

C a **regression test**.

D a **regression fault**.

E **Miller's Law**.

CQ 21: Accepting a user request to add a new error check on a screen, which was not included in the original specification is an example of

A the **moving target problem**.

B **feature / scope creep**.

C a **regression test**.

D a **regression fault**.

E **Miller's Law**.

CQ 22: Accidentally breaking an existing screen which is not mentioned in the project specification is an example of

A the **moving target problem**.
B **feature / scope creep**.
C a **regression test**.
D a **regression fault**.
E **Miller's Law**.

CQ 23: Once a S/W development project starts, we can always prevent prevent any changes that affect the project scope from occurring.

A True.
B False.
C I am not sure.

CQ 24: A strength of the Iteration and Incrementation Life-Cycle Model is:

A Multiple opportunities for checking that the S/W product is correct.

B Opportunity to determine the robustness of the underlying **architecture** early.

C Mitigate risks early.

D We have a working version of the software at the end of every increment.

E All of A–D.

CQ 25: The Code-And-Fix Life-Cycle Model is a good choice for a large project to be carried out by a team.

A True.
B False.
C I am not sure.

CQ 26: The purpose of a rapid prototype is for the developers to practise their programming skills before building the real thing.

A True.
B False.
C I am not sure.

CQ 27: An example of a successful open source software product is:

A MySQL
B Notepad++
C R
D Linux
E All of A–D

CQ 28: Give the best answer to describe yourself:

A I have never worked in an Agile environment.

B I have worked in an Agile environment and I found it less productive than a traditional environment.

C I have worked in an Agile environment and I found it equally productive to a traditional environment.

D I have worked in an Agile environment and I found it more productive than a traditional environment.

E I have worked in an Agile environment and I could not compare how productive it was with a traditional environment.

CQ 29: Agile processes:

A use rapid prototyping.
B use pair programming.
C use extreme programming.
D All of A–C.
E Exactly two of of A–C.

CQ 30: The Synchronize and Stabilize Life-Cycle model:

A was developed by Microsoft.

B uses pair programming.

C is an adaptation of iteration-and-incrementation.

D All of A–C.

E Exactly two of of A–C.

CQ 31: The Spiral Life-Cycle model:

A has ingredients of several earlier life-cycle models.
B mitigates risks as best as possible.
C is an adaptation of iteration-and-incrementation.
D All of A–C.
E Exactly two of of A–C.

CQ 32: A SPMP is a

A sound pressure measuring platform.
B simple plan to manage people.
C software project management plan.
D silly person major problem.
E none of A–D.

CQ 33: UML stands for

A Unified Modelling Language.

B Uniform Meta Life-Cycle.

C Unique Metro Links.

D Uniforms Make Light.

E Unopened Macaroni Label.

CQ 34: A **model**:

A is a set of UML diagrams.

B has at most seven elements, in accordance with Miller's Law.

C represents one or more aspects of the S/W product to be developed.

D All of A–C.

E Exactly two of of A–C.

CQ 35: Deliverable and **artifact** mean the same thing.

A True.
B False.
C I am not sure.

CQ 36: A problem with requirements is that they can be

A ambiguous
B incomplete
C contradictory
D exactly two of of A–C.
E all of A–C.

CQ 37: The Unified Process assumes an Object-Oriented paradigm.

A True.
B False.
C I am not sure.

CQ 38: The Spiral Life-Cycle Model is Two-Dimensional.

A True.
B False.
C I am not sure.

CQ 39: The Unified Process is Two-Dimensional.

A True.
B False.
C I am not sure.

CQ 40: Under the Unified Process, we expect some tasks from each workflow to be performed during each phase.

A True.
B False.
C I am not sure.

CQ 41: Two-Dimensional Life-Cycle models are important because:

A they arise from iteration-and-incrementation, which reflects the fact that change cannot be avoided.
B they yield pretty pictures.
C they provide extra challenge to keep us from getting bored with our work.
D All of A–C.
E Exactly two of of A–C.

CQ 42: The fact that adding programmers to an already late project makes the project later is known as

A Miller's Law.
B Brooks' Law.
C Murphy's Law.
D L.A. Law.
E Newton's Third Law.

CQ 43: A feature of Democratic teams is that

A code is owned by the team, not by an individual.

B the team has no leader - all team members have equal status.

C rapid detection of faults results in high quality code.

D Exactly two of of A–C.

E All of A–C.

CQ 44: A feature of a Chief Programmer team is that

A programmers communicate only with the chief programmer, not directly with each other.

B the chief programmer is responsible for every line of code.

C the programmers employ egoless programming.

D All of A–C.

E Exactly two of of A–C.

CQ 45: I have worked on a Democratic team.

A True.
B False.
C I am not sure.

CQ 46: I have worked on a Chief Programmer team.

A True.
B False.
C I am not sure.

CQ 47: For the classical Chief Programmer team:

A The Chief Programmer must attend all code reviews.
B The Chief Programmer must not attend any code reviews.
C neither A nor B.
D Both of A and B.

CQ 48: The (modified) Chief Programmer model is the best team organization in all situations.

A True.
B False.
C I am not sure.

CQ 49: The purpose of a business model is to:

A determine whether a S/W project should proceed or not.

B define the domain for a S/W project.

C describe how the client operates within the domain.

D All of A–C.

E Exactly two of of A–C.

CQ 50: The purpose of a business case is to:

A determine whether a S/W project should proceed or not.

B define the domain for a S/W project.

C describe how the client operates within the domain.

D All of A–C.

E Exactly two of of A–C.

CQ 51: For an open source project to be successful, it requires:

A a key individual who is a superb motivator.
B a team of highly talented programmers.
C a perception that the project is a winner.
D All of A–C.
E Exactly two of of A–C.

CQ 52: The Synchronize-and-Stabilize model pioneered by Microsoft has now been used extensively outside of Microsoft.

A True.
B False.
C I am not sure.

CQ 53: The Synchronize-and-Stabilize model pioneered by Microsoft has one feature in common with both democratic and Chief Programmer teams.

A True.
B False.
C I am not sure.

CQ 54: Stepwise refinement is a response to

A the S/W crisis.
B the moving target problem.
C Brooks' Law.
D Miller's Law.
E Murphy's Law.

CQ 55: Dividing a S/W product into components with minimal overlap in their functionalities is:

A Divide and Conquer.

B Separation of Concerns.

C Both A and B.

D Neither A nor B.

CQ 56: Dividing a S/W product into components each of which is easier to code than the original is:

A Divide and Conquer.

B Separation of Concerns.

C Both A and B.

D Neither A nor B.

CQ 57: The number of faults found during the project is not the best measure of S/W quality because

A those faults get fixed before the S/W product is deployed.

B the more costly faults are the ones not found during the project.

C Both of A and B.

D Neither A nor B.

CQ 58: There is universal agreement among S/W Engineers that the five key metrics given in the text are the most important ones in all projects:

A True.
B False.
C I am not sure.

CQ 59: Our taxonomy of CASE is:

A environments $\rightarrow$ workbenches $\rightarrow$ tools
B environments $\rightarrow$ tools $\rightarrow$ workbenches
C tools $\rightarrow$ workbenches $\rightarrow$ environments
D workbenches $\rightarrow$ tools $\rightarrow$ environments
E None of A–D.

CQ 60: A code artifact that is modified to run in Firefox instead of in Chrome is a:

A variation.
B revision.
C derivation.
D configuration.
E None of A–D.

CQ 61: A code artifact that is modified to fix a fault is a:

A variation.
B revision.
C derivation.
D configuration.
E None of A–D.

CQ 62: A derivation contains a configuration.

A True.
B False.
C I am not sure.

CQ 63: A **baseline** addresses the problem of one developer undoing another developer's changes when both work on fixing faults in the same code artifact.

A True.
B False.
C I am not sure.

CQ 64: The strict definition of a **baseline** can be too slow in practice.

A True.
B False.
C I am not sure.

CQ 65: In terms of difficulty, I thought that the mid-term exam was

A too difficult.

B just right.

C too easy.

CQ 66: In terms of length, I thought that the mid-term exam was

A too long.

B just right.

C too short.

CQ 67: A benefit of implementing CASE tools is

A fewer faults.
B better usability.
C easier maintenance.
D improved morale.
E All of A–D.

CQ 68: Development & SQA teams should be led by independent managers, neither of whom can overrule the other, because

A no individual could understand both development and SQA.

B more managers make the workplace more fun.

C deeply hierarchical organizations are always preferred.

D Often major faults are found as the delivery deadline approaches.

E None of A–D.

CQ 69: In the Cohesion / Coupling Case Study, the modules of Solution 1 have

A low cohesion.

B high cohesion.

C I am not sure.

CQ 70: In the Cohesion / Coupling Case Study, the classes of Solution 2 have

A low cohesion.

B high cohesion.

C I am not sure.

CQ 71: In the Cohesion / Coupling Case Study, the modules of Solution 1 have

A tight coupling.

B loose coupling.

C I am not sure.

CQ 72: In the Cohesion / Coupling Case Study, the classes of Solution 2 have

A tight coupling.

B loose coupling.

C I am not sure.

CQ 73: An SQA professional should chair a review because

A SQA professionals have the greatest interest in making certain the artifacts are all correct.

B SQA professionals are highly trained at running meetings.

C SQA professionals are highly trained at resolving conflicts between team members.

D SQA professionals get lonely in their work and they need social contact with other team members.

E None of A–D.

CQ 74: Reviews should never be used for performance evaluations because

A the number of faults found in an artifact is not a useful measure for performance purposes.

B team members then have an incentive to throw each other under the bus by pointing out each other's faults.

C this creates an incentive for team members to conceal faults.

D this would make the review take too long.

E None of A–D.

CQ 75: A team exercise having the two steps **preparation** and **team analysis of the document** is called a

A walkthrough.

B inspection.

C review.

D performance appraisal.

E None of A–D.

CQ 76: A team exercise having the five steps **overview**, **preparation**, **inspection**, **rework** and **follow-up** is called a

A walkthrough.
B inspection.
C review.
D performance appraisal.
E None of A–D.

CQ 77: The goal of a walkthrough is to identify and fix faults.

A True.
B False.
C I am not sure.

CQ 78: Using an OO paradigm (as in the unified process) facilitates conducting reviews on its artifacts.

A True.
B False.
C I am not sure.

CQ 79: Since inspections are more expensive than walkthroughs, therefore walkthroughs are preferred whenever possible.

A True.
B False.
C I am not sure.

CQ 80: A problem with the definition of execution-based testing is

A that testing is an **inferential process**.
B what do we mean by a **known environment**?
C what do we mean by **selected inputs**?
D Exactly two of A–C.
E All of A–C.

CQ 81: A S/W product that has a long average time between failures scores highly with respect to

A utility.
B reliability.
C robustness.
D performance.
E correctness.

CQ 82: A S/W product that is very user-friendly scores highly with respect to

A utility.
B reliability.
C robustness.
D performance.
E correctness.

CQ 83: A S/W product which satisfies its output specifications when operated under permissible pre-conditions scores highly with respect to

A utility.
B reliability.
C robustness.
D performance.
E correctness.

CQ 84: A S/W product which produces a useful error message instead of crashing when supplied with unacceptable input scores highly with respect to

A utility.
B reliability.
C robustness.
D performance.
E correctness.

CQ 85: Real-time S/W products have the most strict requirements with respect to

A utility.
B reliability.
C robustness.
D performance.
E correctness.

CQ 86: The correctness of a S/W product guarantees its acceptability.

A True.
B False.
C I am not sure.

CQ 87: The acceptability of a S/W product guarantees its correctness.

A True.
B False.
C I am not sure.

CQ 88: Proofs of program correctness can be cost-justified when human lives depend on program correctness.

A True.
B False.
C I am not sure.

CQ 89: A programmer should not be ultimately responsible for testing his/her own code because

A attachment to the code creates an incentive not to expose faults in the code.

B the programmer may have misunderstood the specification.

C both of A and B.

D neither A nor B.

CQ 90: The expected results for each test case should be determined **after** the test run is completed.

A True.
B False.
C I am not sure.

CQ 91: Testing stops when the client prefers not to spend more money maintaining the S/W product.

A True.
B False.
C I am not sure.

CQ 92: An advantage of encapsulation is:

A design is done effectively with stepwise refinement.

B maintenance is done effectively because of abstraction (data/procedural).

C both of A and B.

D neither A nor B.

CQ 93: An abstract data type is a class if it supports

A information hiding.

B polymorphism.

C inheritance.

D encapsulation.

E none of A–D.

CQ 94: A module has **encapsulation** if it contains

A a data structure.
B operations to be performed on that data structure.
C both of A and B.
D neither A nor B.

CQ 95: During design, if we focus on a particular common level of methods, temporarily ignoring methods that call them and the methods they call, this is an example of

A data abstraction.
B procedural abstraction.
C functional abstraction.
D universal abstraction.
E None of A–D.

CQ 96: Daylight Savings Time should be abolished.

A True.
B False.
C I am not sure.

CQ 97: During design, if we focus on a particular common level of table design, temporarily ignoring tables above and below this level, this is an example of

A data abstraction.
B procedural abstraction.
C functional abstraction.
D universal abstraction.
E None of A–D.

CQ 98: If a module has encapsulation, then it has information hiding.

A True.
B False.
C I am not sure.

CQ 99: If a module has information hiding, then it has encapsulation.

A True.
B False.
C I am not sure.

CQ 100: In our example of a `File` class with derived classes `Hard Disk File`, `Flash Drive File` and `Tape File`, the phenomenon where the system determines which method `Open` method to execute at run time is called

A dynamic binding.
B polymorphism.
C inheritance.
D encapsulation.
E none of A–D.

CQ 101: In our example of a `File` class with derived classes `Hard Disk File`, `Flash Drive File` and `Tape File`, the phenomenon where the `Open` applies to different methods is called

A dynamic binding.
B polymorphism.
C inheritance.
D encapsulation.
E none of A–D.

CQ 102: **Information hiding** means

A keeping the data and the operations on that data together.

B creating methods to access the data.

C putting data out of sight.

D identifying the relevant data items and temporarily ignoring the rest.

E none of A–D.

CQ 103: Data abstraction means

A keeping the data and the operations on that data together.

B creating methods to access the data.

C putting data out of sight.

D identifying the relevant data items and temporarily ignoring the rest.

E none of A–D.

CQ 104: The problem that occurs when modifying a base class affects every class derived from that base class is called the

A moving target problem.
B software crisis.
C fragile base class problem.
D Brooks' problem.
E none of A–D.

CQ 105: The first project done using the OO paradigm will take significantly longer than doing the same project using the classical paradigm.

A True.
B False.
C I am not sure.

CQ 106: A benefit of adopting the OO paradigm, which offsets the higher cost of doing the first project in OO is that

A post-delivery maintenance costs are reduced.

B subsequent projects become faster because of re-use.

C both of A and B.

D neither A nor B.

CQ 107: Inheritance can make the **base** class get large.

A True.
B False.
C I am not sure.

CQ 108: Inheritance can make a **derived** class get large.

A True.
B False.
C I am not sure.

CQ 109: The negative impacts of poor programming on post-delivery maintenance are magnified when using the OO paradigm.

A True.
B False.
C I am not sure.

CQ 110: An impediment to re-use is

A a new compiler version forces coding changes

B (in SQA) test cases outdated: no longer represent current business logic

C Ego: unwillingness to use someone else's code ("Not Written Here" syndrome)

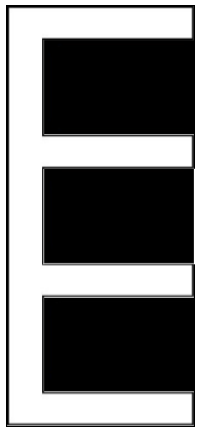
D Quality - lack of trust in past work.

E All of A–D.

CQ 111: The OO paradigm is superior to the classical paradigm with respect to fostering re-use.

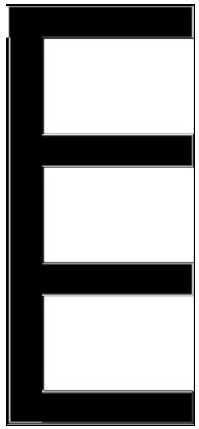
A True.
B False.
C I am not sure.

CQ 112: The given diagram represents



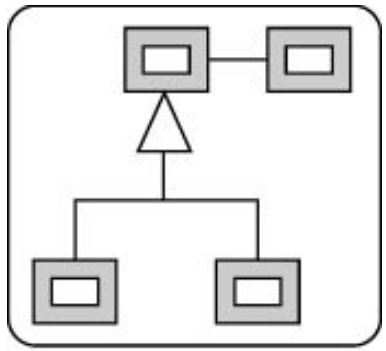
A Library re-use.
B Application Framework re-use.
C Design Pattern re-use.
D None of A–C.

CQ 113: The given diagram represents



A Library re-use.
B Application Framework re-use.
C Design Pattern re-use.
D None of A–C.

CQ 114: The given diagram represents



A Library re-use.
B Application Framework re-use.
C Design Pattern re-use.
D None of A–C.

CQ 115: Design Pattern re-use can occur with the OO paradigm or the classical paradigm.

A True.
B False.
C I am not sure.

CQ 116: Application Framework re-use can occur with the OO paradigm or the classical paradigm.

A True.
B False.
C I am not sure.

CQ 117: Library re-use can occur with the OO paradigm or the classical paradigm.

A True.
B False.
C I am not sure.

CQ 118: A potential obstacle to portability with respect to hardware is

A the first platform might use ASCII, while the target platform might use EBCDIC.

B the first platform might use 64-bit words, while the target platform might use 32-bit words.

C Neither A nor B.

D Both of A and B.

CQ 119: A potential obstacle to portability with respect to operating systems is

A different syntax for JCL (Job Control Language).

B the first O/S might support virtual memory, while the target O/S might not.

C Neither A nor B.

D Both of A and B.

CQ 120: Portability benefits an organization that develops S/W products because

A If the organization's primary business is selling S/W, then more portability means more revenue.
B If the organization's primary business is not selling S/W, then more portability means longer S/W life as H/W changes.
C neither A nor B.
D Both of A and B.

CQ 121: Developing a S/W product in UNIX is superior to developing under another operating system with respect to fostering portability.

A True.
B False.
C I am not sure.

CQ 122: The OO paradigm is superior to the classical paradigm with respect to fostering portability.

A True.
B False.
C I am not sure.

CQ 123: Estimates given after the completion of the **analysis workflow** are more accurate than estimates given after the completion of the **requirements workflow**.

A True.
B False.
C I am not sure.

CQ 124: The accuracy of estimates can be negatively affected by

A different individuals' notions of S/W quality.
B staff turnover.
C varying experience levels between staff members.
D exactly two of of A–C.
E all of A–C.

CQ 125: The metric of **function points** provides a way to compare the sizes of different S/W products on a consistent basis.

A True.
B False.
C I am not sure.

CQ 126: The metric of function points can be used under the classical or the OO paradigm.

A True.
B False.
C I am not sure.

CQ 127: The function point score for a S/W product is independent of the choice of programming language used to develop it.

A True.
B False.
C I am not sure.

CQ 128: Two S/W products with equal unadjusted function points (UFP) are the same size.

A True.
B False.
C I am not sure.

CQ 129: The function point score for a S/W product is an estimate of the effort required to develop it.

A True.
B False.
C I am not sure.