**1. How does software differ from the artifacts produced by other engineering disciplines? Give examples.**

Software can be basically defined as a product as well as a vehicle for delivering a product. As a product, software is regarded as an information transformer. As a vehicle for delivering a product, software serves as the basis for computer communication, control as well as creation of other programs. A good example is visual studio a software product for Microsoft windows which is used by many developers today to develop programs therefore this serves as an information transformer and serves as a basis for computer communication as well as creation of other programs.

**2. Explain what is wrong with the notation that computer software does not need to evolve over time?**

With computer software, it is necessary to revise them as errors are discovered which need corrections. Software must be updated so that they can accommodate the changes in the computing environment. Many times, customers will request changes to add new functions to an existing product to accommodate changes in the business environment while sometimes an older system needs to be re-engineered to provide benefits to the user in a modern context. The bottom line is that software that does not evolve will at one time become unusable.

**3. Why has the personal software process not been widely adopted by industry?**

The personal software process is very challenging and demands level of commitment eg. lengthy and costly training required which is not possible to obtain. In addition, the required level of measurement is culturally hard for many software practitioners.

**4.Why are evolutionary models considered by many to be the best approach to software development in a modern context?**

The evolutionary model is an approach based on the idea of rapidly developing an initial software implementation from very abstract specifications and modifying this according to your appraisal. Evolutionary models are considered by many to be the best approach to software development in a modern context because timelines for the development of modern software are becoming shorter and shorter, customers are becoming more diverse making the understanding of requirements even harder, changes to requirements are becoming more common that is before delivery therefore the need of a way to provide incremental or evolutionary delivery. The evolutionary process accommodates uncertainty better than most process models; allows the delivery of partial solutions in an orderly and planned manner and most importantly reflects what really happens when complex systems are developed.

**5. What are the benefits of using analysis patterns during the analysis modeling process?**

-They facilitate the transformation of the analysis model into a design model by suggesting design patterns and reliable solutions for common problems.

-They speed up the development of abstract analysis models that capture the main requirements of the concrete problem by providing re-usable analysis models.

**6. Describe the contents of the webApp, content, functional, interaction and configuration models?**

* WebApp analysis model. The web app analysis model contains **structural elements** which identify classes and content objects required to create a webApp which meets the stakeholder’s needs as well as **dynamic elements** that describe how structural elements interact with one another and how they interact with the end users.
* Content model. The content model contains structural elements that represent webApp content requirements, webApp content objects, all analysis classes and analysis classes defined by class diagrams showing attributes, operations and class collaborations. Content model is derived from careful examination of webApp use-cases. Entity-relationship diagrams may also be part of the content model.
* Interaction model. This model contains use-cases (dominant element of webApp interaction model), sequence diagrams (basically provides representation of way user actions collaborate with analysis classes), state diagrams as well as user interface prototype (a layout of content presentation, interaction mechanism and overall aesthetic of user interface.)
* Configuration model. This model may be a list of server-side and client-side attributes for the webApp.UML deployment diagrams can also be used for complex configuration architectures. This model describes the environment and infrastructure in which the webApp resides.
* Functional model. In this model, user observable behavior is delivered to end users, operations contained in analysis classes to implement class behaviors and UML activity diagrams used to model both.

**7. Explain how a process specification (PSPEC) differs from a control specification (CSPEC)?**

A process specification describes the algorithm, the input to a function, the PSPEC restrictions and limitations imposed on the process, performance characteristics that are relevant to the process and at large the design constraints that may influence the way in which the process will be implemented. In other terms process specification is used to describe the inner workings of a process described in a flow diagram while control specification is used to indicate how a software behaves when an event or control signal is sensed as well as which processes are invoked because of the occurrence of the event.

**8. How does the object-oriented view of component–level design differ from the traditional view?**

The object-oriented view of component-level design differ from the traditional view in that the object-oriented view basically focuses on the elaboration of design classes that come from both the problem and infrastructure domains. Classes are elaborated by specifying messaging details, defining attribute data structures, identifying interfaces and describing process flow for operations. In the traditional view three of components are refined: domain modules, control modules and infrastructure modules. This requires representation to be created for data structures, interfaces as well as algorithms for each program module in enough detail to generate programming language source code.

**9. List four interface design issues present in the development of most user interfaces?**

* Clear. Clarity is the most important element of user interface design. The whole purpose of user interphase design is to enable people to interact with your system by communicating meaning and function.
* Concise. Clarity in a user interphase is great, however, one needs to be very careful not to over clarify. Adding definitions and explanations is easy but every time you do that you add mass. The interface grows and because of too many explanations users will have to spend a lot of time reading them. It is important to keep things clear as well as concise.
* Familiar. Most of the designer’s strain to make their interfaces intuitive. Being familiar means that something which appears like something else you have encountered before. When you are familiar with something you know how it behaves therefore you know what you expect.
* Consistent. Consistent interphases allow users to develop usage patterns, that is they will learn what the different buttons, tabs, icons and other interphase elements look like and will recognize them and realize what they do in different context.

**10. Describe practices that enable designers to think about using patterns?**

* Ensure that u understand the big picture that is context in which the software will reside.
* Extract the patterns which are present at the level of obstruction in the big picture.
* Begin your design with big picture patterns that establish context of skeleton for further design work.
* Work inward for the context that is looking forward for patterns at lower levels of abstraction that contributes to the design solution.
* Repeat the steps one to four until the complete design is fleshed out.
* Refine the design by adopting each pattern to the specific of the software that you are trying to build.

**11. What are three dimensions of software quality?**

* Compatibility. This is the suitability of software for use in different environments like different operating systems, browsers among others.
* Accessibility. This defines the degree to which software can be used comfortably by a wide variety of people inclusive of those who require assistive technologies like screen magnifiers or voice recognition.
* Concurrency. This is the ability of software to service multiple requests to the same resource at the same time.
* Efficiency. This is the ability of software to perform or achieve a result without wasted energy, effort, resources, time or money.

**12. Why is regression testing an important part of any integration testing procedure?**

Regression testing is basically a critical part of any integration testing procedure because you want to verify that the change made did not introduce some unexpected effect. Regression testing is then the verification that things still work and integration testing is verification that new things work. In the worst case, regression testing might require a full-blown acceptance test and that might take much longer to do than the testing for one or two lines of code that u changed. Modularization, good design and good record keeping can minimize regression testing because you can argue that certain things do not need to be retested in one area because u changed something in another area. You can also argue for minimization of regression testing if you have robust code controls in place.

**13. Describe object oriented unit testing?**

This is where the individual classes are tested to check whether the class attributes are implemented per design and whether the methods and interfaces are error free.

**14. List four types of systems tests?**

* Graphical user interphase testing. This process involves testing of products graphical user interphase to ensure it meets its specification.
* Usability testing. This is a technique used in what is referred to as user centered interaction design and evaluates a product by testing it on users.
* Compatibility testing. This is testing conducted on the application to evaluate the applications compatibility with the computing environment.
* Software performance testing. This is a testing practice performed to determine how a system performs in terms of responsiveness and stability under a workload.

**15. What are the key differences between validation testing goals and acceptance testing goals?**

In validation testing the test team seeks to ensure that each software function or performance characteristics conforms to its specification. In acceptance testing, the test team needs to ensure that the software works correctly for the intended user in his or her normal work environment.