Page No:
Date: / /

SOFTWARE ENGINEERING
CLASS WORK

SUBMITTED BY: AIMAN SIDDIQUA
2K18/MC/008

MULTI CHOICE QUESTIONS.

5.1 ~~(b) Data Coupling~~ (b) Data Coupling
5.2 ~~(a) External Coupling~~ Content Coupling
5.3 (c) Functional Cohesion
5.4 (d) Coincidental Cohesion
5.5 (c) Embedded design
5.6 (c) Cohesion with respect to time
5.7 (a) Operations are part of single functional task and are placed in same procedures.
5.8 (d) Common coupled.
5.9 (c) ~~Temporal Cohesion~~ Procedural Cohesion
5.10 (c) Cohesion
5.11 (a) Closed System
5.12 (a) Coupling

EXERCISES.

5.1 A design is a plan or specification for the construction of an object or system or for the implementation of an activity or process.

Page No:
Date: / /

Software design is the process of envisioning & defining software solutions to one or more sets of problems.

Conceptual Design:
- written in customer's language
- explains the observable external characteristics of the system.
- independent of implementation

Techinal design
- describes major hardware components and their functions
- shows hierarchy of software components
- shows data structures & data flow
- shows interfaces.

5.2 Objectives:
- Identify different types of softwares based on the usage
- shows differences among design and coding
- describes concepts of structured programming
- See how to design for testibility as well as maintainability.

The transformation of an informal design to a detailed design is done as:

Page No:
Date: / /

Informal design outline → Informal Design → More formal Design → Finished design.

5.3 No we do not design software when we "write" a program.
Design is the description of the logic, which is used in solving the problem.
Coding is the language specification which is implemented of the design. It runs on the computer and provides the expected result.

5.4 A module is a software component that is created by dividing the software.
The process of creating software modules is known as Modularity.

Important properties of a Modular system:
1. Modular Decomposability (Break down into smaller pieces.)
2. Modular Understandability (Make it easier for the user to understand each module so that it is easy to develop and change)
3. Modular Composability (combine modules that are created)
4. Modular Continuity (unbroken or uninterrupted connection for a long period)

Page No:
Date: / /

5. Modular Protection (Keep safe the other modules from abnormal condition occuring in a particular module at run time)

5.5 Module Coupling means to couple two or more modules with each other and with the outside world.

Types of coupling:

1. Data Coupling - Coupling of data i.e interaction b/w data when they are passed through parameters
2. Control Coupling - Control Data sharing between modules.
3. Common Coupling - Sharing of common data or global data between several modules.
4. Content Coupling - Using of data or control information maintained in other modules by one modules.
5. Stamp Coupling - Sharing of composite data structures between modules.
6. External Coupling - Sharing of data structure or format that are imposed externally between the modules.

Page No:
Date: / /

5.6 Define module cohesion and explain different types of cohesion.

Cohesion defines the degree to which the elements of a module belong together. It measures the strength of relationship b/w piece functionality within a given module.

Types:

1. Functional Cohesion : All elements contribute to execution of one task
2. Sequential cohesion : Elements are involved in activities such that output data from one activity becomes input data for the next.
3. Communicational Cohesion : Elements contribute to activities that use the same input or output data.
4. Procedural Cohesion : Elements are related only by sequence, otherwise activities are unrelated.
5. Temporal Cohesion : Elements are involved in activities that are related in time.
6. Logical Cohesion : Elements contribute to activities of the same general category
7. Coincidental Cohesion : Elements contribute to activities with no meaningful relationship to one another.

Page No:
Date: / /

5.7 Discuss the objectives of modular software design? What are the effects of module coupling and cohesion.

The objective of modular design are to enhance clarity, easy implementation, debugging testing, documenting and maintenance of the software.

A good modular system creates a well designed reusable, easy to use piece of software.

Coupling and cohesion are two complementary properties. Coupling shows relationship b/w modules, whereas cohesion shows relationship within the modules.

High coupling is bad for the software as it decreases maintainability whereas high cohesion is better as it makes it more reusable.