COMPUTER ENGINEERING DEPARTMENT

ASSIGNMENT NO-02

SUB: Software Engineering

COURSE: T.E. **Year: 2020-2021** **Semester: VI**
DEPT: Computer Engineering
SUBJECT CODE: CSC601 **SUBMISSION DATE: 10/04/2021**

==========

Name: Amey Thakur **Roll No.: 50**

Class: TE Comps B-50 **ID: TU3F1819127**

Assignment No 2

Sr. No.	Question	CO mapping	PO mapping	PSO Mapping
1	What is Software Design? Explain various fundamentals design concepts.	CO4	PO2,PO3, PO10, PO11, PO12	1
2	Differentiate between Cohesion and coupling. Explain various types of coupling with suitable examples.	CO4	PO2,PO3, PO10, PO11, PO12	1
3	Define Risk and Risk Projection. Develop an RMMM plan for the following risk; **i)** Staff turnover occur **ii)** Computer crash	CO5	PO1,PO2, PO3,PO4, PO5,PO7, PO8,PO9, PO10, PO11, PO12	1

4	Explain change control and version control in SCM.	CO5	PO1,PO2, PO3,PO4, PO5,PO7, PO8,PO9, PO10, PO11, PO12	1,2
5	Differentiate between white box and black box testing. Explain white box testing in detail. Also, explain cyclomatic complexity.	CO6	PO3,PO4, PO6,PO7, PO8,PO9, PO10, PO11, PO12	1,2
6	Explain different types of software maintenance in detail.	CO6	PO3,PO4, PO6,PO7, PO8,PO9, PO10, PO11, PO12	1,2

AMEY B 50 Amey

Page No.: YOUVA
Date:

Q1. What is Software Design ? Explain various fundamentals design concepts.

Ans:

Software Design

- It is a process to transform user requirements into some suitable form, which helps the programmer in software coding and implementation.
- It deals with representing the client's requirement, as described in SRS (Software Requirement Specification) document, into a form, i.e. easily implementable using programming language.

Fundamental of Software Design Concepts.

① Abstraction

- Hide irrelevant data.
- Abstraction simply means to hide the details to reduce complexity and increases efficiency or quality.
- Different levels of abstraction are necessary and must be applied at each stage of the design process. so that any error that is present can be removed to increase the efficiency of the software solution and to refine the software solution. The solution should be described in broadways that cover a wide range of different things at a higher level of abstraction. and a more detailed description of a solution of software should be given at the lower level of abstraction.

② Modularity

- Subdivide the system.
- Modularity simply means to divide the system or project into smaller parts to reduce the complexity of the system or project. In the same way, modularity in design means to subdivide a system into smaller parts so that these parts can be created independently and then use these parts in different systems to perform different functions. It is necessary to divide the software into components known as modules because nowadays there are different software available like Monolithic Software that is hard to grasp for software engineers, So, modularity in design has now become a trend and is also important.

③ Architecture

- Design a structure of something.
- Architecture simply means a technique to design a structure of something.
- Architecture in designing software is a concept that focuses on various elements and the data of the structure
- These components interact with each other and use the data of the structure in architecture.

④ Refinement

- Removes impurities.
- Refinement simply means to refine something to remove any impurities if present and increase the quality.
- The refinement concept of software design is actually a process of developing or presenting the software or system in a detailed manner that means to elaborate a system or software.

AMEY B 50 Amey

Page No.:
Date:
YOUVA

⑤ Pattern

- A repeated form
- The pattern simply means a repeated form or design in which the same shape is repeated several times to form a pattern.
- The pattern in the design process means the repetition of the solution to a common recurring problem within a certain context.

⑥ Information Hiding

- Hide the information
- Information hiding simply means to hide the information so that it cannot be accessed by an unwanted party.
- In software design, information hiding is achieved by designing the modules in a manner that the information gathered or contained in one module is hidden and it can't be accessed by any other modules.

⑦ Refactoring

- Reconstruct something
- Refactoring simply means to reconstruct something in such a way that it does not affect the behavior or any other features.
- Refactoring in software design means to reconstruct the design to reduce complexity and simplify it without affecting the behavior or its functions.
- Fowler has defined refactoring as "the process of changing the software system in a way that it won't affect the behavior of the design and improves the internal structure."

AMEY B 50 Amey

Page No.: Date: YOUVA

Q2 Differentiate between Cohesion and Coupling. Explain various types of coupling with suitable example.

Ans:

Basis of Comparison	Cohesion	Coupling
Description	Cohesion represents the degree to which a part of a code base forms a logically single atomic unit.	Coupling represents the degree to which a single unit is independent from others. (Coupling is the number of connections between two or more units.)
Concept	It is an intra-module concept.	It is an inter-module concept.
What does it depicts?	It depicts the module's relative functional strength	It depicts the relative independence among modules.
What does it entail?	High cohesion is about keeping parts of a code base that are related to each other in a single place.	Low coupling is about separating unrelated parts of the code base as much as possible.
Types / Classes	Coincidental cohesion Logical Cohesion Temporal Cohesion Procedural Cohesion Communication Cohesion Sequential Cohesion Function Cohesion	Data Coupling Control Coupling Stamp Coupling Common Coupling

AMEY B 50 Amey

Page No.: YOUVA
Date:

Manipulation	Cohesion is a kind of natural extension of data hiding. For ex; the class having all members visible with a package having default visibility.	Making private fields, private methods and non-public classes provides loose coupling.
Designing	While designing, you need to strive for high cohesion, that is focus on a single task with little interaction with other modules of the system.	While designing, you need to strive for low coupling that is, dependence between modules should be less.
Increase in Coupling vs Cohesion	Increase in Cohesion is good for software	Increase in coupling is avoided for software
How best software is achieved	High cohesion gives the best software	Loose Coupling gives the best software.
Possibility	It is possible to create fully cohesive code without introducing unnecessary coupling	It is impossible to achieve full decoupling without damaging Cohesion.

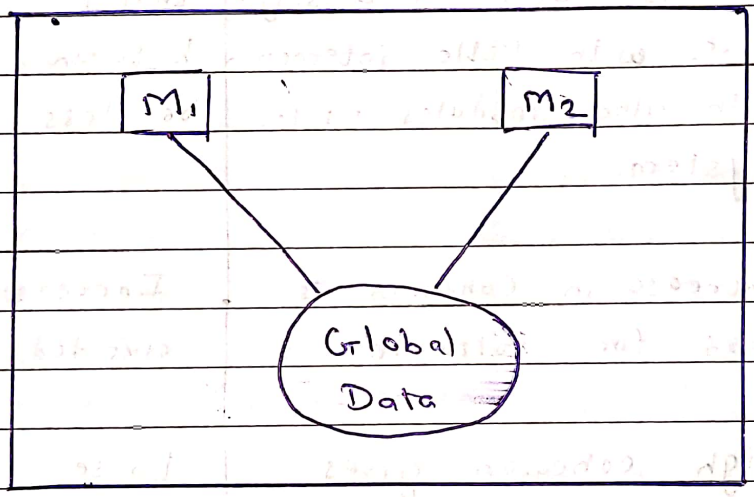
AMEY B 50 Amey

Page No.:
Date:
YOUVA

Types of coupling.

(1) Content coupling
- Content coupling exists among two modules if they share code,
- Example: A branch from one module into another module.

(2) Common coupling
- Two modules are common coupled if they share information through some global data items.



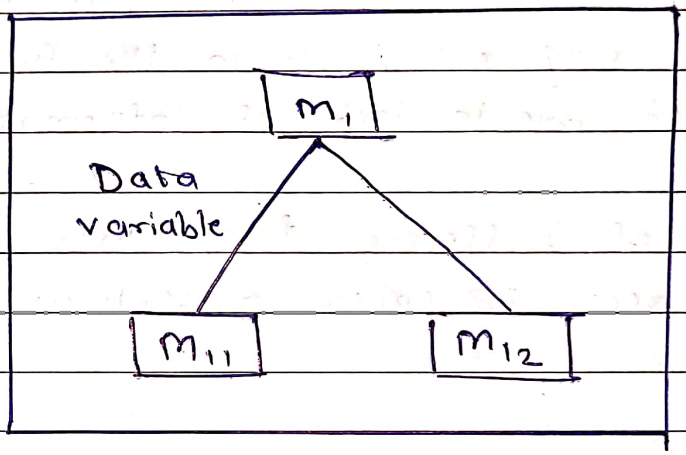
(3) Control coupling
- Control coupling exists among two modules if data from one module is used to direct the structure of instruction execution in another.

(4) Stamp Coupling
- Two modules are stamp coupled if they communicate using composite data items such as structure, objects, etc.
- When the module passes non-global data structure or entire structure to another module, they are said to be stamp coupled.

- Example: Passing structure variable in C or object in C++ language to a module.

⑤ Data Coupling

- When data of one module is passed to another module, this is called data coupling.



Note: Ideally no coupling is considered to be the best.

AMEY B 50 Amey

Page No.:
Date:
YOUVA

Q.3. Define Risk and Risk Projection.
Develop an RMMM plan for the following risk.
(i) Staff turnover occur
(ii) Computer crash.

Ans:

Risk

- Risk is an expectation of a loss, a potential problem that may or may not occur in the future. It is generally caused due to lack of information, control or time.
- A possibility of suffering from loss in software development process is called a software risk.

Risk Projection

- Risk projection, also called risk estimation, attempts to rate each risk in two ways - the likelyhood or probability that the risk is real and the consequences of the problem associated with the risk, should it occur.

Amey B 50 Amey

Page No.:
Date:
YOUVA

RMMM Plan

① Risk : Staff Turnover Occur

Risk Mitigation

- To mitigate this risk, project management must develop a strategy for reducing turnover. The possible steps to be taken are:
  - Meet the current staff to determine causes for turnover. (eg. poor working conditions, low pay, competitive job market).
  - Mitigate those causes that are under our control before the project starts.
  - Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave.
  - Organise project teams so that information about each development activity is widely dispersed.
  - Define documentation standards and establish mechanisms to ensure that documents are developed in a timely manner.
  - Assign a backup staff member for every critical technologist.

AMEY B 50 Amey :-

Page No.:
Date:
YOUVA

Risk Monitoring

- As the project proceeds, risk monitoring activities commence. The project manager monitors factors that may provide an indication of whether the risk is becoming more or less likely.
- In the case of high staff turnover, the following factors can be monitored.
  - General attitude of team members based on project pressures.
  - Interpersonal relationships among team members.
  - Potential problems with compensation and benefits.
  - The availability of jobs within the company and outside it.

Risk Management

- Risk management and contingency planning assumes that mitigation efforts have failed and that the risk has become a reality.
- In the case of high staff turnover, the project is well underway, and a number of people announce that they will be leaving.
- If the mitigation strategy has been followed, backup is available, information is documented, and knowledge has been dispersed across the team.
- In addition, the project manager may temporarily refocus resources (and readjust the project schedule) to those functions that are fully staffed, enabling newcomers who must be added to the team to "get up to the speed."

AMEY B.1 50 Amey

Page No.:
Date:
YOUVA

RMMM Plan

(11) Risk : Computer Crash

Risk Mitigation

- The cost associated with computer crash resulting in a loss of data is crucial. A computer crash itself is not crucial, but rather the loss of data
- A loss of data will result in not being able to deliver the product to the customer. This will result in a not receiving a letter of acceptance from the customer.
- Without the letter of acceptance, the group will receive a failing grade for the course. As a result the organization is taking steps to make multiple backup copies of the software in development and all documentation associated with it, in multiple locations.

Risk Monitoring

- When working on a product or documentation, the staff member should always be aware of the stability of the computing environment they're working in. Any changes in the stability of the environment should be recognized and taken seriously.

Risk Management

- The lack of a stable - computing environment is extremely hazardous to a software development team.
- In the event that the computing environment is found unstable, the development team should cease work on that system until the environment is made stable again, or should move to a system that is stable and continue working there.

AMEY B 50 Amey

Page No.: Date: YOUVA

Q.4 Explain change control and version control in SCM.

Ans:

Software Configuration Management (SCM)

- It is a software engineering discipline consisting of standard processes and techniques often used by organizations to manage the changes introduced to its software products.. SCM helps in identifying individual elements and configurations, tracking changes and version selection
- SCM is known as software control management. SCM aims to control changes introduced to large complex software systems through reliable version selection and version control.

Version Control.

- Software version control is a system or tool that captures the changes to a source code elements; files, folders, images or binaries.
- A version control system (also known as a Revision Control system) is a repository of files, often the files for the source code of computer programs, with monitored access. Every change made to the source is tracked, along with who made the change, why they made it, and references to problems fixed, or enhancements introduced, by the change.
- Version control systems are essential for any form of distributed, collaborative development. Whether it is the history of a wiki page or large software development project, the ability to track each change as it was made, and to reverse changes when necessary can make all the difference between well managed and controlled process and an uncontrolled 'first come, first served' system.

AMEY B 50 Amey

Page No.:
Date:
YOUVA

- Combines procedures and tools to manage the different versions of configuration objects created during the software process.
- Version control systems require the following capabilities.

(1) Project Repository
- Stores all relevant configuration objects.

(2) Version Management Capability
- Stores all versions of a configuration object. (enables any version to be built from past versions.)

(3) Make facility
- Enables collection of all relevant configuration objects and construct a specific software version.

(4) Issues (bug) tracking capability
- Enables team to record and track status of outstanding issues for each configuration object.

- Uses a system modeling approach (template - includes component hierarchy and component build order, construction rules, verification rules).

AMEY B 50 Amey

Page No.: Date: YOUVA

Change Control

- Change control is a systematic approach to managing all changes made to a product or system. The purpose is to ensure that no unnecessary changes are made, that all changes are documented, that services are not unnecessarily disrupted and that resources are used efficiently.
- Here's an example of a six-step process for a software change request.

① Documenting the change request

- When the client requests the change, that request is categorized and recorded, along with informal assessments of the importance of that change and the difficulty of implementing it.

② Formal assessment

- The justification for the change and risks and benefits of making / not making the change are evaluated. If the change request is accepted, a development team will be assigned.
If the change request is rejected, that fact is documented and communicated to the client.

③ Planning

- The team responsible for the change creates a detailed plan for its design and implementation, as well as a plan for rolling back the change should it be deemed unsuccessful.

AMEY B 50 Amey

Page No.:
Date:
YOUVA

④ Designing and Testing
- The team designs the program for the software change and tests it. If the change is deemed unsuccessful, the team requests approval and a date for implementation.

⑤ Implementation and review
- The team implements the program and stakeholders review the change.

⑥ Final assessment
- If the client is satisfied that the change was implemented satisfactorily, the change request is closed. If the client is not satisfied, the project is reassessed and steps may be repeated.

AMEY B 50 Amey

Page No.:
Date:
YOUVA

Q5 Differentiate between white box and black box testing. Explain white box testing in detail. Also explain cyclomatic complexity.

Ans:

White Box Testing	Black Box Testing
① It is a way of testing the software in which the tester has knowledge about the internal structure or the code or the program of the software.	① It is a way of software testing in which the internal structure or the program or the code is hidden and nothing is known about it.
② It is mostly done by software developers.	② It is mostly done by software testers.
③ Knowledge of implementation is required.	③ No knowledge of implementation is needed.
④ It is the inner or the internal software testing.	④ It can be referred as outer or external software testing.
⑤ It is structural test of the software.	⑤ It is functional test of the software.
⑥ This type of testing of software is started after detail design document.	⑥ This testing can be initiated on the basis of requirement specifications documents.

AMEY B. ..50 Amey

Page No.: Date: YOUVA

White Box Testing	Black Box Testing
(7) It is mandatory to have knowledge of programming	(7) No knowledge of programming is required
(8) It is the logic testing of the software	(8) It is the behavior testing of the software
(9) It is generally applicable to the lower levels of software testing	(9) It is applicable to higher levels of software testing
(10) It is also called as clear box testing	(10) It is also called as closed testing
(11) It is most time consuming	(11) It is least time consuming
(12) It is suitable for algorithm testing	(12) It is not suitable or preferred for algorithm testing
(13) Data domains along with inner or internal boundries can be better tested	(13) Can be done by trial and error ways and method
(14) Example: by input to check and verify loops	(14) Example: Search something on google using keyword
(15) Types of White Box Testing A. Path Testing B. Loop Testing C. Condition Testing	(15) Types of Black Box Testing A. Functional Testing B. Non-functional Testing C. Regression Testing

AMEY B 50 Amey

Page No.. Date: YOUVA

White Box Testing

- It is the testing of a software solution's internal coding and infrastructure. It focuses primarily on strengthening security, the flow of inputs and outputs through the application, and improving design and usability.
- White box testing is also known as Clear box testing, Open box testing, Structural testing, Transparent box testing, Code-based testing and Glass box testing.
- It is one of two parts of the "box testing" approach of software testing. Its counter part, black box testing involves testing from an external or end user type perspective.
- On the other hand, Whitebox testing is based on the inner workings of an application and revolves around internal testing.

White box testing involves the testing of the software code for the following:

- Internal security holes.
- Broken or poorly structured paths in the coding process.
- The flow of specific inputs through the code.
- Expected output.
- The functionality of conditional loops.

AMEY B 50 Amey

Page No.: Date: YOUVA

How do you perform white box testing?

- To give you a simplified explanation of white box testing we have divided it into two basic steps.
- This is what testers do when testing an application using white box testing technique:

Step 1: Understand the source code

- The first thing a tester will often do is learn and understand the source code of the application. Since white box testing involves the testings of the inner workings of an application, the tester must be very knowledgeable in the programming languages used in the applications they are testing. Also, the testing person must be highly aware of secure coding practices.

- Security is often one of the primary objectives of testing software. The tester should be able to find security issues and prevent attacks from hackers and naive users who might inject malicious code into the application either knowingly or unknowingly.

Step 2: Create test cases and execute.

- This second basic test to white box testing involves testing the application's source code for proper flow and structure. One way is by writing more code to test the application's source code. The tester will develop little tests for each process or series of process in the application.

Page No.: Date: YOUVA

- This method requires that the tester must have intimate knowledge of the code and is often done by the developer. Other methods include manual testing, trial and error testing and the use of testing tools.

White Box Testing Techniques

① Statement Coverage
- This technique is aimed at exercising all programming statements with minimal tests.

② Branch Coverage
- This technique is running a series of tests to ensure that all branches are tested at least once.

③ Path Coverage
- This techniques corresponds to testing all possible paths which means that each statement and branch is covered.

Advantages of white box testing
- Forces test developer to reason carefully about implementation
- Reveals errors in hidden code
- Spots the dead code or other issues with respect to best programming practices

Disadvantages of white box testing
- Expensive as one has to spend both time and money to perform white box testing
- Every possibility that few lines of code are missed accidentally
- In-depth knowledge about the programming language is necessary to perform white box testing

AMEY B 50 Amey

Page No.: Youva
Date:

Cyclomatic Complexity

- Cyclomatic complexity is a source code complexity measurement that is being correlated to a number of coding errors. It is calculated by developing a control flow graph of the code that measures the number of linearly independent paths through a program module.
- Cyclomatic complexity is used to gauge the overall intricacy of an application or specific functionality within it. The software metric quantitatively measures a program's logical strength based on existing decision paths in the source code. It is computed by using the control flow graph, where each node on the graph represents individual groups or commands within the program.
- For example, an application consisting of zero decision points (IF, FOR, etc.) has an intricacy score of 1 because it contains a single path in the source code. If the program contains an IF statement consisting of one condition, the code would contain a total of two paths : TRUE or FALSE.
  The cyclomatic complexity algorithm is used to derive a measurable value based on the number of edges and nodes within the graph as well as the total count of connected components or exit nodes.
- It is represented as

Complexity (M) = Edges (E) − Nodes (N) + Exit Nodes (P)

AMEY B 50 Amey

Page No.:
Date:
YOUVA

- Individual programs, subroutines, or methods always have one exit node, thus resulting in the value of P equaling one. This value remains consistent unless multiple applications with several exit points are to be assessed at the same time.
- Lower the Program's cyclomatic complexity, lower the risk to modify and easier to understand.
- It can be represented as:

Cyclomatic Complexity = E − N + 2 * P

where,

E = No. of Edges in the flow graph.
N = No. of Nodes in the flow graph.
P = No. of Nodes that have exit points.

Example:

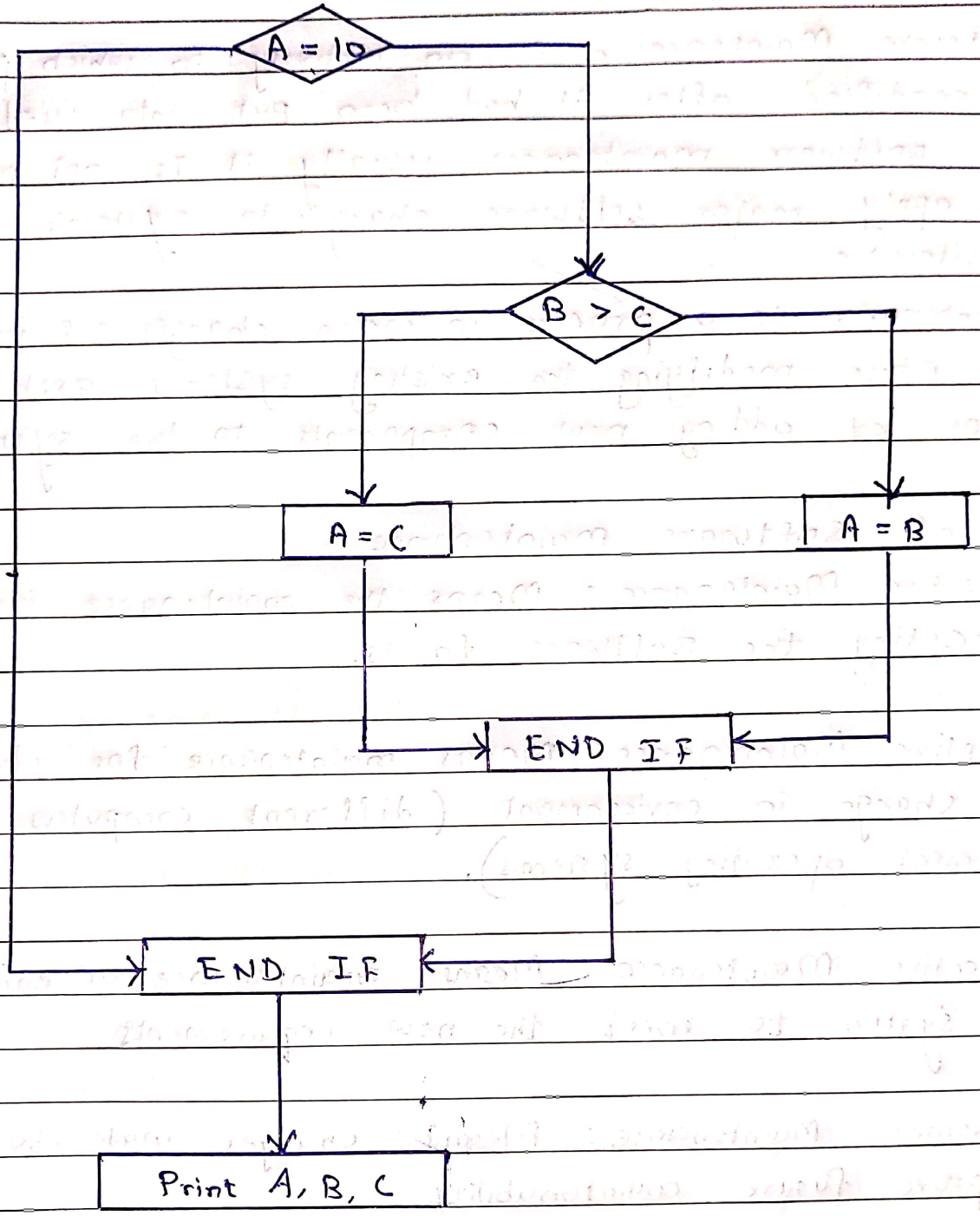
```
IF A = 10 THEN
IF B > C THEN
A = B
ELSE
A = C
ENDIF
ENDIF
Print A
Print B
Print C
```

AMEY B 50 Amey

Page No.:
Date:
YOUVA

Flowgraph

- Cyclomatic complexity in Test life Cycle.



- The cyclomatic complexity is calculated using the above control flow diagram that shows seven nodes (shapes) and eight edges (lines).
- Hence the cyclomatic complexity is $8 - 7 + 2 = 3$.

AMEY B 50 Amey

Page No.: Date: YOUVA

Q.6 Explain different types of software maintenance in detail.

Ans:

Software Maintenance

- Software Maintenance is an activity in which program is modified after it had been put into use.
- In software maintenance usually it is not preferred to apply major software changes to system's architecture.
- Maintenance is a process in which changes are implemented by either modifying the existing system's architecture p/or by adding new components to the system.

Types of Software maintenance

(1) Corrective Maintenance : Means the maintenance for correcting the software faults.

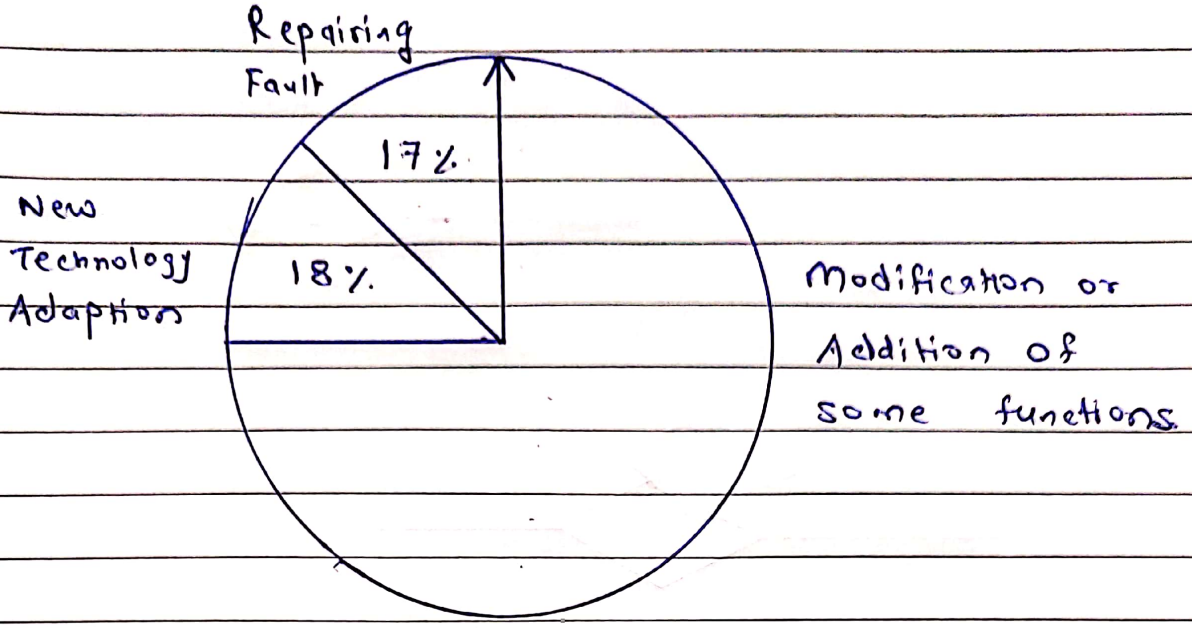
(2) Adaptive Maintenance: Means maintenance for adapting the change in environment (different computers or different operating systems).

(3) Perfective Maintenance : Means maintenance or enhancing the system to meet the new requirements.

(4) Preventive Maintenance: Means changes made to improve future maintainability.

AMEY B 50 Amey

Page No.:
Date:
YOUVA



Maintenance Cost

- Above figure shows that the making some modifications in the existing system or adding some new functionalities is very costly from maintenance point of view.
- Nearly 65% of efforts are required for such maintenance.
- If the operating environment gets changed then 18% of maintenance cost will be required. But repairing or correcting faults is less costly which may cost 17% of total cost effort.
- According to Lientz and Swanson (in 1980) and Nosek and Palvia (in 1990), if new requirements are added then it needs lot of efforts to maintain such systems.