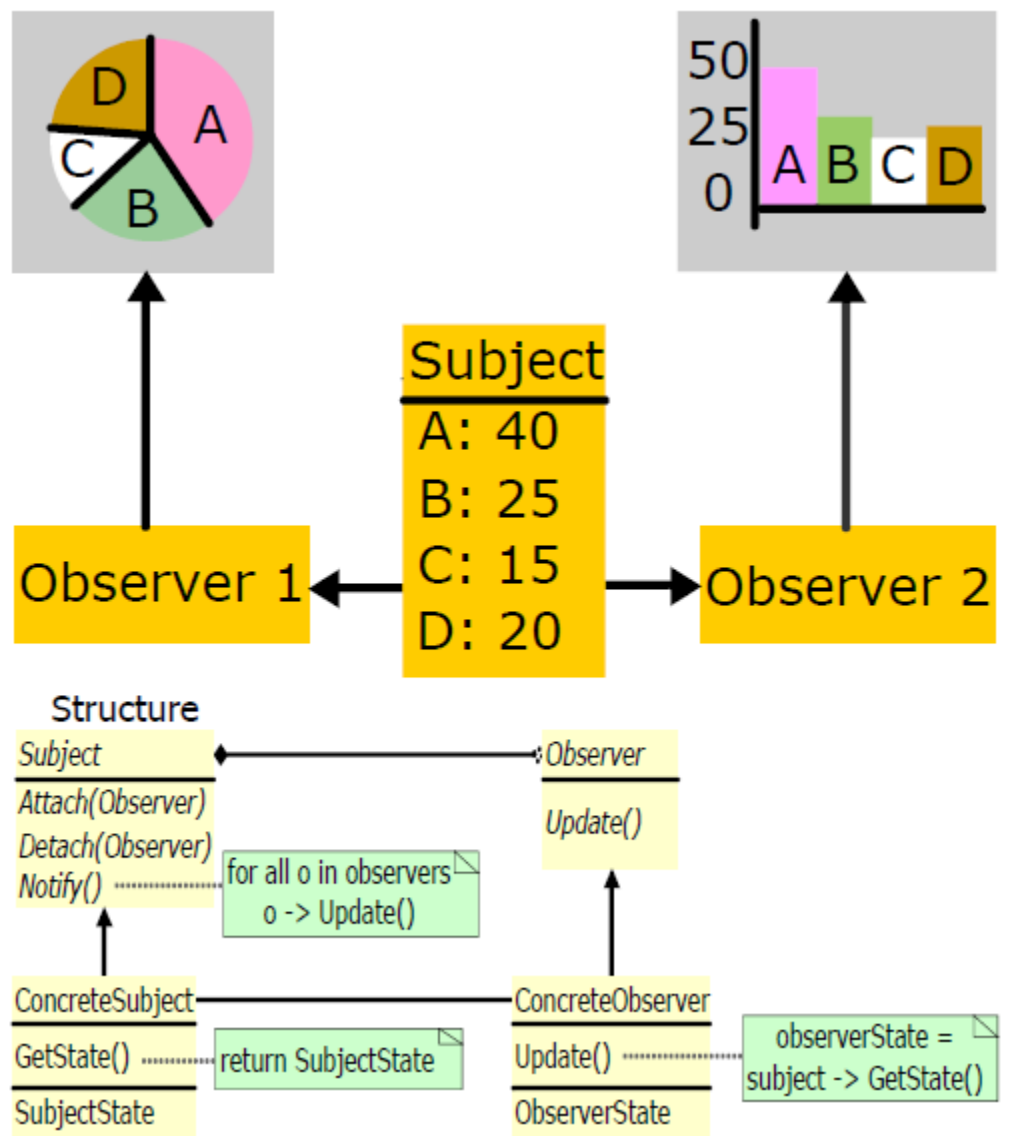
CS504 SOLVED FINAL TERM CURRENT PAPER FALL 2021

1) Define Logical error and its 3 symptoms?

Logical Errors: A logical error occurs when the code is syntactically correct but does not do what you expect it to do.

Symptoms: 1) The code is misbehaving in a way that isn't easily explained. 2) Results are the opposite of what is expected. 3) Output looks strange, but has no obvious symptoms of corruption.

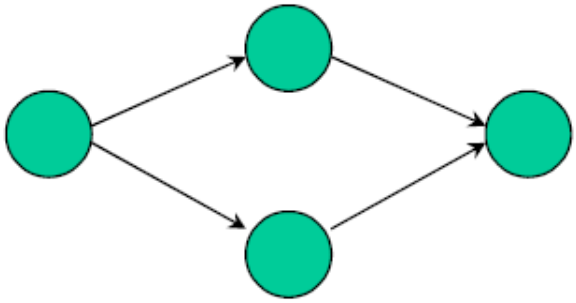
2) design pattern given thy unki class btani thi table me 5 marks scenario given tha uska design pattern btana tha and why?



3) Flow notation draw krni thi if loop or case statement ki

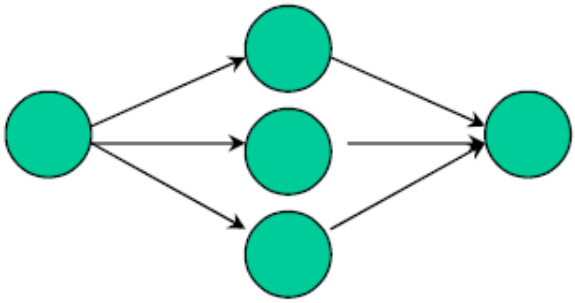
If

Second structural form is the If statement. In the following graph, the first node at the left depicts the if statement and the two nodes next to the first node correspond to the successful case (if condition is true) and unsuccessful case (if condition is false) consecutively. The control comes to the same instruction from either of these intermediate instructions.



Case

In Case statement, control can take either of several branches (as opposed to only two in If statement.) First node represents the switch statement (C/C++) and nodes in middle correspond to all different cases. Program can take one branch and result into the same instruction.



4) what do you know about byte order and give example

Byte Order and Data Exchange

The order in which bytes of one word are stored is hardware dependent. For example in Intel architecture the lowest byte is the most significant byte while in Motorola architecture the highest byte of a word is the most significant one.

5) Get/Set and is statement related?

1. The terms get/set must be used where an attribute is accessed directly.

employee.getName();
matrix.getElement (2, 4);
employee.setName (name);
matrix.setElement (2, 4, value);

2. is prefix should be used for boolean variables and methods.

isSet, isVisible, isFinished, isFound, isOpen

5) Aik long cyclomatic complexity se tha

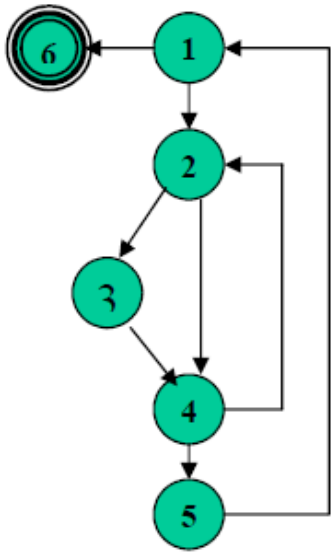
Cyclomatic complexity

The concept of cyclomatic complexity is extremely useful in white box testing when analyzing the relative complexity of the program to be tested. Cyclomatic Complexity, V(G), for a flow graph G is defined as: V(G) = E - N + 2

Cyclomatic Complexity of a Sort Procedure

Following is the same bubble sort program that we discussed above. This time we shall calculate its cyclomatic complexity and see how many test cases are needed to test this function.

```
Sorted = FALSE;
while (!sorted) {                 //1
   sorted = TRUE;
   for (i=0; i < N-1; i++) {      //2
      if a[i] > a[i+1] {
         swap(a[i], a[i+1]);      //3
         sorted = FALSE;
      }                           //4
   }                              //5
}                                 //6
```



Cyclomatic complexity

- Number of edges = 8
- Number of nodes = 6
- C(G) = 8-6+2 = 4

Paths to be tested

- Path1: 1-6
- Path2: 1-2-3-4-5-1-6
- Path3: 1-2-4-5-1-6
- Path4: 1-2-4-2-3-4-5-6-1

6) aik short inspection conditions ka th

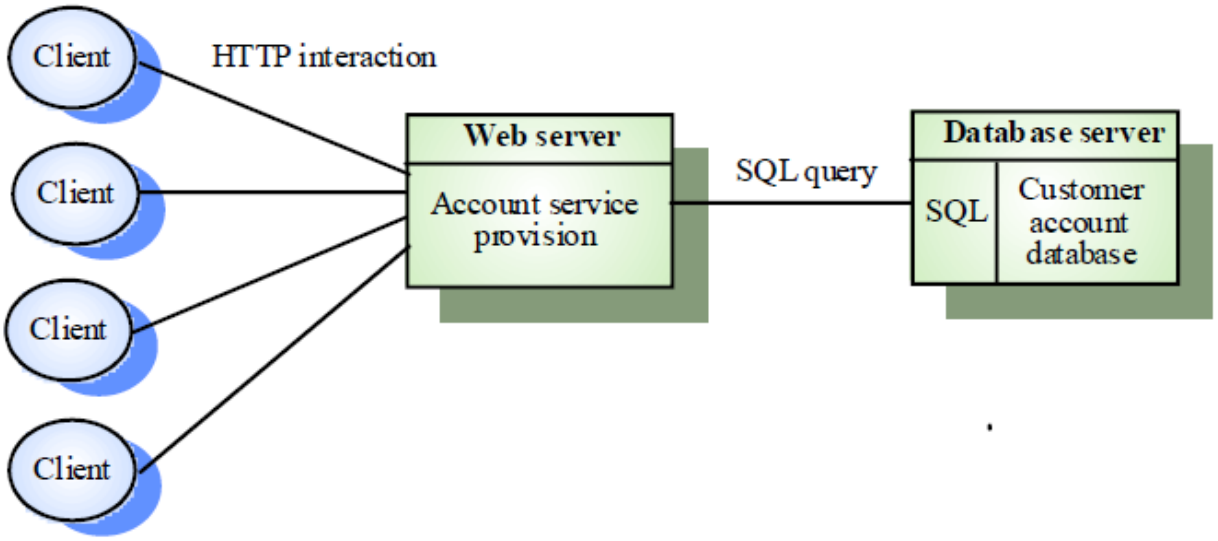
Inspection pre-conditions: A precise specification must be available before inspections. Team members must be familiar with the organization standards. In addition to it, syntactically correct code must be available to the inspectors. Inspectors should prepare a checklist that can help them during the inspection process.

7) Draw a diagram representing Three-Tier architecture of Virutal University Learning Management System (VULMS).

Three-tier Architecture

In this architecture, each application architecture layers (presentation, application, database) may run on separate processors. It therefore allows for better performance than a thin-client approach. It is simpler to manage than fat client approach and highly scalable (as demands increase add more servers).

A typical 3-tier architecture is depicted in the following diagram.



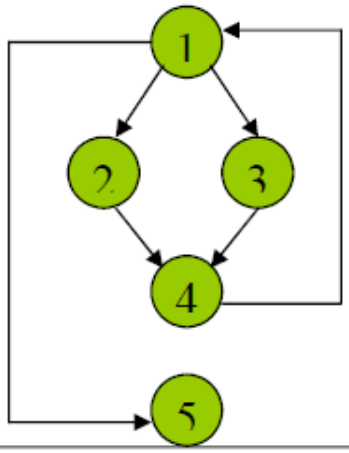
9) code tha 3 number ka jis main circle ka area nikalna hota bs os main correction krni thi
Solution:
main(){
Int a;
Int r;
a=3.14*r*r;
Cout<<a;
}
Is main bs r ki jaga radius krna tha aur a ki jaga area
10) code ki correction krni thi
while(i=0;i<10;i++){
Cout<<i;
}
Isko for loop say b theek kr skte aur do while say b
Solution:
1. while (i =0; i < 10; i++) {
2. cout << i << newl;
3. }

11) find path coverage of the given code

Paths in a program containing loops

Now we shall apply the path coverage scheme on a piece of code that contains a loop statement and see how many test cases can possibly be developed.

```
for (i = 0; i < N; i++) {          //1
  if (condition1)
        // do something here        //2
  else
        // do something here        //3
  // something here                 //4
}
                                   //5
```



Paths

The following is an analysis of the above-mentioned code and the flow diagram. It determines the number of paths against different iterations of the loop.

- N = 0: If the control does not enter into the loop then only one path will be traversed. It is 1-5.
- N=1: Two different paths can possibly be traversed (depending on condition).
  - 1-2-4-1-5
  - 1-3-4-1-5
- N=2: Four possible paths can be traversed.
  - 1-2-4-1-2-4-1-5
  - 1-2-4-1-3-4-1-5
  - 1-3-4-1-2-4-1-5
  - 1-3-4-1-3-4-1-5
- Generalizing the relation between loop variable N and the number of possible paths, for the value of N, 2^N paths are possible
  - Thus if N = 20 it means more then 1 million paths are possible.

12) GOF Design Pattern Format

The basic template includes ten things as described below

Name: □ Name has to be meaningful

Problem: □ A statement of the problem which describes its intent

Context: □ Preconditions under which the problem and its solutions seem to occur

Forces: □ Relevant forces and constraints and their interactions and conflicts.

Solution: □ Static and dynamic relationships describing how to realize the pattern.

Examples: □ One or more sample applications to illustrate

Resulting context: □ the state or configuration after the pattern has been applied

Rationale: □ justification of the steps or rules in the pattern

Related patterns: □ the static and dynamic relationships between this pattern and other patterns
Known uses: □ to demonstrate that this is a proven solution to a recurring problem

13) Aik general naming conventions ka question tha
General Naming Conventions
1. Names representing types must be nouns and written in mixed case starting with upper case.
Line, FilePrefix
2. Variable names must be in mixed case starting with lower case.
line, filePrefix
3. Names representing constants must be all uppercase using underscore to separate words.
MAX_ITERATIONS, COLOR_RED

14) Exception handling or iska try catch exception ka code likhna tha
Exception handling is a powerful technique that separates error-handling code from normal code. It also provides a consistent error handling mechanism. The greatest advantage of exception handling is its ability to handle asynchronous errors. The following code segment illustrates the general exception handling mechanism.

```
try {
___...
___...
___throw Exception()
___...
___...
} catch( Exception e )
{
___...
___...
}
```

15) Aik abused flag waly ka tha
Use (or abuse) of Zero
The number 0 is the most abused symbol in programs written in C or C++. One can easily find code segment that 0 in a fashion similar to the examples below in almost every C/C++ program.

```
flag = 0; // flag is boolean
str = 0; // str is string
name[i] = 0; // name is char array
x = 0; // x is floating pt
i = 0; // i is integer
```

16) main parts of test case
Thus a test case involves
□ Input and output specification plus a statement of the function under test.
□ Steps to perform the function
□ Expected results that the software application produces

17) run-over memory bug two point write krny thy

Memory over-runs: a memory overrun occurs when you use memory that does not belong to you. This can be caused by overstepping an array boundary or by copying a string that is too big for the block of memory it is defined to hold
Symptoms: 1) Program crashes quite regularly after a given routine is called, that routine should be examined for a possible overrun condition. 2) If the routine in question does not appear to have any such problem the most likely cause is that another routine, called in the prior sequence, has already trashed variables or memory blocks. 3) Checking the trace log of the called routines leading up to one with the problem will often show up the error.

18) Data centered architecture:
In any system, sub-systems need to exchange information and data. This may be done in two ways:
1. Shared data is held in a central database or repository and may be accessed by all subsystems
2. Each sub-system maintains its own database and passes data explicitly to other subsystems

19) write about vertical partitioning and horizontal
Partitioning of an architecture may be "horizontal" and/or "vertical".
In the horizontal partitioning we define separate branches of the module hierarchy for each major function and control modules are used to coordinate communication between functions.
Vertical partitioning divides the application from a decision making perspective. The architecture is partitioned in horizontal layers so that decision making and work are stratified with the decision making modules residing at the top of the hierarchy and worker coming at the bottom.

20) Participants of observer pattern
Participants
Subject: □ Knows its observers. Any number of Observer objects may observe a subject.
Observer: □ Defines an updating interface for objects that should be notified of changes in a subject.
ConcreteSubject: □ Stores state of interest to concreteObserver objects.
ConcreteObserver: □ Maintains a reference to a ConcreteSubject object.

21) What is client-server configurations
The client-server model can have many different configurations. In the following sections, we look at some of these configurations.
Thin Client Model
Fat Client Model
Zero Install
N-Tier architecture
Three-tier Architecture

21) guidelines for avoiding hazards side-effect
If the following guidelines are observed, one can avoid hazards caused by side effects.
1. never use "," except for declaration
2. if you are initializing a variable at the time of declaration, do not declare another variable in the same statement
3. never use multiple assignments in the same statement

4. Be very careful when you use functions with side effects – functions that change the values of the parameters.
5. Try to avoid functions that change the value of some parameters and return some value at the same time.

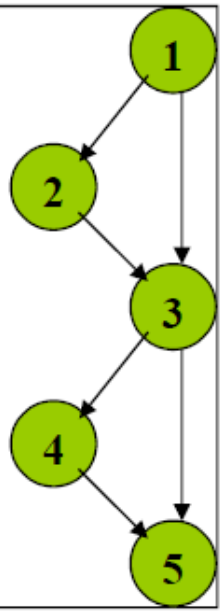
22) code was given for infeasible path... We have to correct the code

Infeasible paths

Infeasible path is a path through a program which is never traversed for any input data.

Example

```
if (a == b)          //1
    c = c-1;         //2
if (a != b)          //3
    c = c+1;         //4
                     //5
```



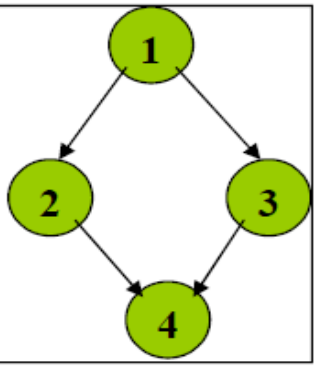
In the above-mentioned example, there are two infeasible paths that will never be traversed.

- Path1: 1-2-3-4-5
- Path2: 1-3-5

Modified code segment

Infeasible paths can be analyzed and fixed.

```
if (a == b)          //1
      c = c-1;       //2
else
      c = c+1;       //3
                     //4
```



There are no infeasible paths now!

23) Software bugs examples
Below, you can find a list of some of the main types of software bugs:
Crash, Functional error, Acknowledgement message error, Typos, Missing command, Calculation error, Hardware usage error & Control flow error.

24) During the testing phase of software, when are two Tests considered to be equivalent? OR When we can say that the two software tests are equivalent
Two tests are considered to be equivalent if it is believed that:
 if one discovers a defect, the other probably will too, and
 if one does not discover a defect, the other probably won't either.

25) What is short circuiting? 3marks
Short-circuiting is a very useful tool. It can be used where one boolean expression can be placed first to "guard" a potentially unsafe operation in a second boolean expression.

26) Correct the errors in the given program of short circuiting 5marks
```
struct Node {
int data;
Node * next;
};
Node *ptr;
...
while (ptr->data < myData && ptr != NULL){
// do something here
}
```

Answer:
The second part of condition, ptr! = NULL, is supposed to be the guard. When the guard is misplaced, if the pointer is NULL then the program will crash because it is illegal to access, This code is rewritten as follows.
```
while (ptr != NULL && ptr->data < myData){
// do something here
}
```

27) Enlist client server configuration
In a client/server configuration, by putting servers into logical member groups, you can control which servers your clients use and target specific servers for specific data or tasks.

28) Self Documenting Code
A self documenting code is a code that explains itself without the need of comments and extraneous documentation, like flowcharts, UML diagrams, process-flow state diagrams, etc.
Example
```
flag = false;
str = NULL;
name[i] = '\0';
x = 0.0;
```

i = 0;

29) Explain styles marks: 1. Function and methods 2. Template class
A function is directly called by its name, whereas a method includes a code that is called by the object's name. A function can pass and then return the data that is operated, whereas a method operates data in a Class. A function is independent, whereas a method is a function linked with an object.
template class in C++ is a class that allows the programmer to operate with generic data types. This allows the class to be used on many different data types as per the requirements without the need of being re-written for each type

30) Black box r white box ma difference btana tha
Black box testing: In this type of testing, a component or system is treated as a black box and it is tested for the required behavior.
Structural testing (white box): As opposed to black box testing, in structural or white box testing we look inside the system and evaluate what it consists of and how is it implemented.

31) Comment ki 2 guidelines btani thi
1. All comments should be written in English. In an international environment English is the preferred language.
2. Use // for all comments, including multi-line comments.
// Comment spanning
// more than one line

32) Unit testing
A software program is made up of units that include procedures, functions, classes etc. The unit testing process involves the developer in testing of these units. Unit testing is roughly equivalent to chip-level testing for hardware in which each chip is tested thoroughly after manufacturing.

33) Effective testing
The objective of testing is to discover the maximum number of defects with a minimum number of resources before the system is delivered to the next stage.

34) Krutchen architecture ka sawal tha
In this model, the architecture is again prepared and analyzed from 5 different perspectives. The 4 main views are Functional View, the Concurrency View, the Physical View, and the Development View. Code view is not present in the original Krutchen model and is basically an extension of the development view. His architectural model is known as Krutchen's 4+1 architectural view model.

35) What are the symptoms for Memory Leak bug
A memory leak bug is one in which memory is somehow allocated from either the operating system or an internal memory "pool"
Symptoms
- System slowdowns
- Crashes that occur "randomly" over a long period of time

36) What are the Software Verification Validation statements

Verification and validation are the processes in which we check a product against its specifications and the expectations of the users who will be using it. According to a known software engineering expert Berry Boehm, verification and validation are

Verification

 Does the product meet system specifications?
 Have you built the product right?

Validation

 Does the product meet user expectations?
 Have you built the right product?

37) Fix the Code Statements for Side Effects

Following is short list of common mistakes made due to side-effects.

1.

array[i++] = i;

If i is initially 3, array[3] might be set to 3 or 4.

2.

array[i++] = array[i++] = x;

Due to side effects, multiple assignments become very dangerous.

3.

"," is very dangerous as it causes side effects. Let's look at the following statement:

38) Portability

Many applications need to be ported on to many different platforms. As we have seen, it is pretty hard to write error free, efficient, and maintainable software. So, if a major rework is required to port a program written for one environment to another, it will be probably not come at a low cost.

39) What is umbrella activities?

Management activities are kind of umbrella activities that are used to smoothly and successfully perform the construction activities Some of the major management activities are listed below.

 Project Planning and Management
 Configuration Management
 Software Quality Assurance
 Installation and Trainin

40) Spilt line ka tha

Split lines occurs when a statement exceed the 80 column limit.

In general: •Break after a comma. •Break after an operator. •Align the new line with the beginning of the expression on the previous line.

41) Can a tester write test cases without having an in-depth knowledge about the system and requirements from users

It's pretty rare but the situation can arise where we are expected to review without being given a functional specification document. Without a set of the usual testing documents, we will be unable to follow the standard testing practices.

42) name at least three phases of the software development lifecycle where there is high chance of defect entrance into the software.
A: debugging and testing, implementation and maintenance.

43) according to a famous software engineer martin fowler, "any fool can write code that computers can understand, good programmers write code that humans can understand." what do you understand by above statement?
A: it is not important to write code that works, it is important to write code that works and is easy to understand so that it can be maintained.

44) : write down exception handling syntax with the help of a generalized try and catch block of code.
A:

```
try
{
    //code
      throw parameter;
}
catch(exceptionname ex)
{
      //code to handle exception
}
```

45) what do you understand by the term effective testing? which software document is considered as the base document to start the testing phase of software?
A: efficiency testing tests the amount of resources required by a program to perform a specific function. in software companies, this term is used to show the effort put in to develop the application and to quantify its user-satisfaction. software testing documentation is considered as the base document to start the testing phase of software

46) write name of krutchen's architectural view models.
A: logical view, process view, development view, physical view

47) differentiate between bug and debugging
A: when an error is found in a software, it is called a bug. the process of finding the error in a software testing is called debugging.

48) what are the main elements which must be part of any test case?
A: test case id and test case objective

49) name any two areas of programming other than object oriented programming (oop) where the concept of design patterns has been applied.

A: c and pascal

50) State one advantage and disadvantage of Bit Fields and write its code?
Advantage: Bit fields allow the packing of data in a structure. Bit fields are a suitable way to express many difficult operations.
Disadvantage: Bit fields suffer from a lack of portability.
Code:

```
Struct packed_struct {
Unsigned int f1:1;
Unsigned int f2:1;
Unsigned int f3:1;
Unsigned int f4:1;
Unsigned int type: 4;
Unsigned int funny_int:9;
} pack;
```

51) Consider the following values and calculate cyclomatic complexity.
Number of edges=8
Number of nodes=6
Answer:
$M = E - N + 2$
Where
M = cyclomatic complexity
E = number of edges of the graph
N = number of nodes of the graph
So $M= 8 – 6 + 2 = 4$

Q 38: Why Special characters like TAB and page break must be avoided? Explain Special characters like TAB and page break must be avoided. These characters are sure to cause problem for editors, printers, debuggers when used in a multi-programed or multi-platform environment.