

17624

21718

3 Hours / 100 Marks

Seat No.

--	--	--	--	--	--	--	--

- Instructions :**
- (1) All Questions are *compulsory*.
  - (2) Answer each next main Question on a new page.
  - (3) Illustrate your answers with neat sketches wherever necessary.
  - (4) Figures to the right indicate full marks.
  - (5) Assume suitable data, if necessary.

**Marks**

1. (A) Attempt any **THREE** of the following : **12**
- (a) Give any four difference between Quality Assurance and Quality Control.
  - (b) Describe technical review under static testing.
  - (c) With the help of neat diagram, describe unit testing.
  - (d) Give any two advantages and any two limitations of Function/test matrix.
- (B) Attempt any **ONE** of the following : **06**
- (a) Prepare and write six test cases for simple calculator application.
  - (b) Describe Defect Management Process with neat & labelled diagram.
2. Attempt any **FOUR** of the following : **16**
- (a) What is Boundary Value Analysis ? List any three guidelines for boundary value analysis.
  - (b) Describe how to select testing tool.

[1 of 4]

P.T.O.

- (c) Prepare and write four test cases for Library Management System of college.
- (d) Give any four difference between Manual and Automated testing.
- (e) Explain Requirement defects and Design defects.
- (f) Explain Equivalence partitioning with respect to equivalence classes.

**3. Attempt any FOUR of the following :**

**16**

- (a) Explain any one type of structural white box testing in detail.
- (b) Give any four difference between alpha and beta testing.
- (c) What is external test standard ? Give any three types of external standards.
- (d) Describe how to perform usability and GUI testing.
- (e) Give any two advantages and any two disadvantages of using testing tools.
- (f) Explain Defect template.

**4. (A) Attempt any THREE of the following :**

**12**

- (a) Give any four advantages of test planning.
- (b) What do you mean by Software Metrics ? List any three types of metrics.
- (c) What are different techniques for finding defects ? Explain in detail.
- (d) Describe the factors considered to decide test strategy or test approach.

**(B) Attempt any ONE of the following :**

**06**

- (a) Differentiate between verification and validation.
- (b) Distinguish between white box testing and black box testing. (any six)

**5. Attempt any TWO of the following :****16**

- (a) Describe test reporting in detail. How to prepare a summary report ?
- (b) With the help of an example, explain Graph-based testing.
- (c) Explain following concepts related to Integration testing with neat & labelled diagram :
  - (i) Top-down testing.
  - (ii) Bottom-up testing.

**6. Attempt any FOUR of the following :****16**

- (a) Give any two root causes of defects. Also give any two effects of defects.
  - (b) What is Software testing ? Give any three objectives of testing.
  - (c) Explain Defect Life Cycle with diagram.
  - (d) What is Test plan ? List test planning activities.
  - (e) Describe compatibility testing with an example.
  - (f) Explain Client-Server testing.
-





SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

**Important Instructions to examiners:**

- 1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
- 2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
- 3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
- 4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q. N.	Answers	Marking Scheme																				
1.	(A)	Attempt any THREE of the following:	12Marks																				
	(a)	Give any four differences between Quality Assurance and Quality Control.	4M																				
	Ans:	<table><tr><th>Quality Assurance</th><th>Quality Control</th></tr><tr><td>Process oriented activities.</td><td>Product oriented activities.</td></tr><tr><td>QA is the process of managing for quality.</td><td>QC is used to verify the quality of the output</td></tr><tr><td>They measure the process, identify the deficiencies/weakness and suggest improvements.</td><td>They measure the product, identify the deficiencies/weakness and suggest improvements.</td></tr><tr><td>Relates to all products that will ever created by a process</td><td>Relates to specific product</td></tr><tr><td>Activities of QA are Process Definition and Implementation, Audits and Training</td><td>Activities of QC are Reviews and Testing</td></tr><tr><td>Verification is an example of QA</td><td>Validation/Software Testing is an example QC</td></tr><tr><td>Preventive activities.</td><td>It is a corrective process.</td></tr><tr><td>Quality assurance is a proactive process</td><td>Quality control is a reactive process.</td></tr><tr><td>QA is a managerial tool</td><td>QC is a corrective tool</td></tr></table>	Quality Assurance	Quality Control	Process oriented activities.	Product oriented activities.	QA is the process of managing for quality.	QC is used to verify the quality of the output	They measure the process, identify the deficiencies/weakness and suggest improvements.	They measure the product, identify the deficiencies/weakness and suggest improvements.	Relates to all products that will ever created by a process	Relates to specific product	Activities of QA are Process Definition and Implementation, Audits and Training	Activities of QC are Reviews and Testing	Verification is an example of QA	Validation/Software Testing is an example QC	Preventive activities.	It is a corrective process.	Quality assurance is a proactive process	Quality control is a reactive process.	QA is a managerial tool	QC is a corrective tool	(Any Four differences: 4 marks)
Quality Assurance	Quality Control																						
Process oriented activities.	Product oriented activities.																						
QA is the process of managing for quality.	QC is used to verify the quality of the output																						
They measure the process, identify the deficiencies/weakness and suggest improvements.	They measure the product, identify the deficiencies/weakness and suggest improvements.																						
Relates to all products that will ever created by a process	Relates to specific product																						
Activities of QA are Process Definition and Implementation, Audits and Training	Activities of QC are Reviews and Testing																						
Verification is an example of QA	Validation/Software Testing is an example QC																						
Preventive activities.	It is a corrective process.																						
Quality assurance is a proactive process	Quality control is a reactive process.																						
QA is a managerial tool	QC is a corrective tool																						



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

(b)	<b>Describe technical review under static testing.</b>	<b>4M</b>
Ans:	<p><b>Technical Review:</b></p> <p><b>i. Formal Review:</b></p> <ul style="list-style-type: none"><li>• A formal review is the process under which static white box testing is performed .</li><li>• A formal review can range from a simple meeting between two programmers to a detailed, rigorous inspection of the code.</li></ul> <p>There are four essential elements to a formal review</p> <ol style="list-style-type: none"><li>1. Identify Problems:</li><li>2. Follow Rules:</li><li>3. Prepare: -</li><li>4. Write a Report:</li></ol> <p><b>ii. Peer Reviews:</b></p> <ul style="list-style-type: none"><li>• The easiest way to get team members together and doing their first formal reviews of the software is through peer reviews, the least formal method.</li><li>• Sometimes called buddy reviews, this method is really more of a discussion.</li><li>• Peer reviews are often held with just the programmer who wrote the code and one or two other programmers or testers acting as reviewers.</li><li>• Small group simply reviews the code together and looks for problems and oversights.</li><li>• To assure that the review is highly effective all the participants need to make sure that the four key elements of a formal review are in place: Look for problems , follow rules, prepare for the review, and write a report.</li><li>• As peer reviews are informal , these elements are often scaled back. Still, just getting together to discuss the code can find bugs.</li></ul>	<p><b>(Explanation: 4 marks)</b></p>
(c)	<b>With the help of neat diagram, describe unit testing.</b>	<b>4M</b>
Ans:	<p><b>{{**Note:Any other relevant diagram can be considered**}}</b></p> <p><b>Unit Testing :</b> Software product is made up of many units , each unit needed to be tested to find whether they have implemented the design correctly or not.</p> <p><b>Additional Requirements :</b> The module under consideration might be getting inputs from another module or the module is calling some another module . Some interface modules has to be simulated if required like drivers and stubs.</p> <p><b>Drivers:</b> The module where the required inputs for the module under test are simulated for the purpose of module or unit testing is known as a Driver module. The driver module may print or interpret the result produced by the module under test.</p>	<p><b>(Diagram:2 marks, Explanation :2 marks)</b></p>

SUMMER – 18 EXAMINATION

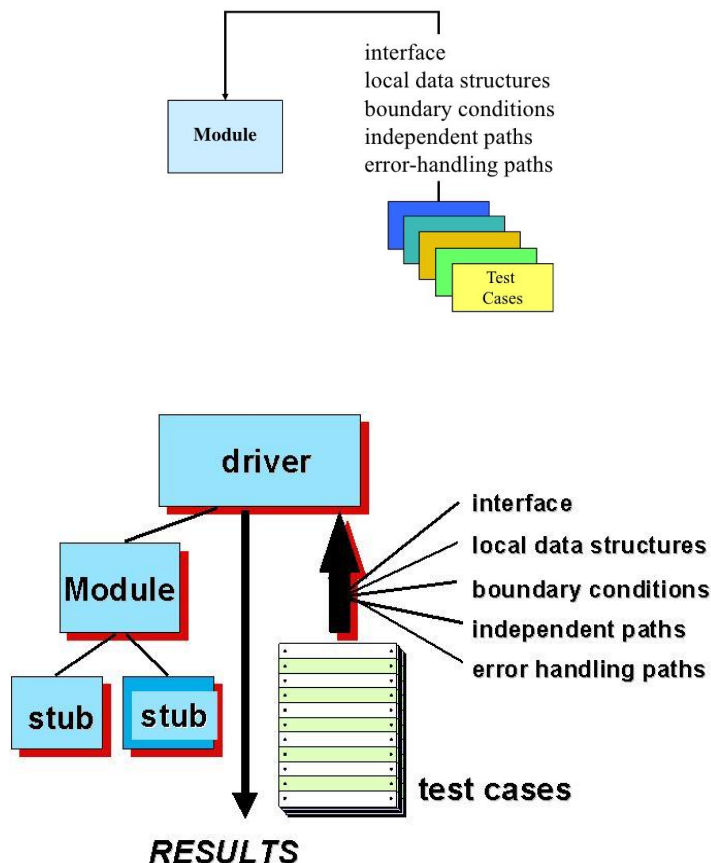
Subject Name: Software Testing

Model Answer

Subject Code:

17624

**Stubs:** The module under testing may also call some other module which is not ready at the time of testing. There is need of dummy modules required to simulate for testing, instead of actual modules. These are called stubs.



**Unit testing procedure:**

Unit testing is normally considered as an adjunct to the coding step. The design of unit test can be performed before coding begins or after source code has been generated. Guidance for establishing test cases for finding out undiscovered errors can be taken by the review of design information. Each test case need to be associated with set of expected results with it. In many applications a driver is known as “main program” that takes input from test case data, also passes that data to the component that need to be tested and prints concerned results.

Stubs are useful for replacing modules which are subordinate of the component that we are going to test. A dummy sub program or stub can make use of subordinate modules interface. It generally does very less data manipulation, also provides verification of entry and used to return control to the module i.e. currently undergoing the testing.

(d) Give any two advantages and any two limitations of Function/test matrix.

4M

Ans: Two advantages :

(Any 2 advantages:



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

1. It increases the function traceability.
2. It helps to improve the serialization of processes.
3. It is able to identify/locate different tests involved in processes.

**OR**

- Track a requirement from conception through to delivery,
- Plan and manage testing and defect triages better,
- Reduce leakage, wastage of precious resource on non-priority, or simply non-requirements,
- Document adequately, and
- Work effectively in a world of integrations.

**Two disadvantages :**

1. This needs advance planning.
2. It becomes time consuming in many of the cases where requirements are more.
3. Incomplete work results into:
  - Poor or unknown test coverage, more defects found in production
  - It will lead to miss some bugs in earlier test cycles which may arise in later test cycles. Then a lot of discussions arguments with other teams and managers before release.
  - Difficult project planning and tracking, misunderstandings between different teams over project dependencies, delays, etc

**2 marks,  
Any 2  
Disadvantages: 2 marks)**

**(B) Attempt any ONE of the following:**

**6 Marks**

**(a) Prepare and write six test cases for simple calculator application.**

**6M**

**Ans:**

Sr. No	Test Case_ID	Test Case Objectives	Pre-requisites	Steps	Input data	Expected Result	Actual Result	Status

**(Any other relevant test case should be considered.  
6 test cases 6 marks)**





SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

		1	TC_1	To add two integer and display the result on ten digit calculator	Calculator is switched on	1.Key in a valid integer from -9999999999 to +9999999999  2.Key in operator +  3.Key in second operand,a valid integer from -9999999999  To +9999999999	135 + 100	235(addition, above ten digits will be expressed in exponential form)	235	Pass	
		2	TC_2	To subtract two integer and display the result on ten digit calculator	Calculator is switched on	1.Key in a valid integer from -9999999999 to +9999999999  2.Key in operator -  3.Key in second operand,a valid integer from -9999999999  To +9999999999	135- 100	35(subtraction, above ten digits will be expressed in exponential form)	35	Pass	
		3	TC_3	To multiply two integer and display the result on ten digit calculator	Calculator is switched on	1.Key in a valid integer from -9999999999 to +9999999999  2.Key in operator x  3.Key in second operand,a valid integer from -9999999999  To	100 x 400	40000(multiplication, above ten digits will be expressed in exponential form)	40000	Pass	



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

					+999999999					
		4	TC_4	To divide two integer and display the result on ten digit calculator	Calculator is switched on	1.Key in a valid integer from -9999999999 to +9999999999  2.Key in operator /  3.Key in second operand,a valid integer from -9999999999  To +9999999999	100/25	40(division,above ten digits will be expressed in exponential form)	40	Pass
		5	TC_5	To clear the screen	Calculator is switched on	Press C		Symbol '0' should appear on screen	Symbol '0' appears on screen	Pass
		6	TC_6	To delete digits one by one	Calculator is switched on	Press <- (backspace)		One Digit should be deleted from right hand side	One Digit is deleted from right hand side	Pass

(b)	<b>Describe Defect Management Process with neat &amp; labeled diagram.</b>									<b>6M</b>
Ans:	<p><b>Defect management process diagram:</b></p> <pre> graph LR     A[Defect Prevention] --&gt; B[Deliverable Baseline]     B --&gt; C[Defect Discovery]     C --&gt; D[Defect Resolution]     D --&gt; E[Process Improvement]     E --&gt; A     F[Management Reporting] --&gt; A     F --&gt; B     F --&gt; C     F --&gt; D     F --&gt; E </pre> <p>As shown in the above diagram the defect management process is divided into</p>									<b>(Diagram:2 marks, Description: 4 marks)</b>



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

		<p><b>following tasks:</b></p> <p>i). <b>Defect Prevention</b> - Implementation of techniques, methodology and standard processes to reduce the risk of defects.</p> <p>ii). <b>Deliverable Baseline</b> - Establishment of milestones where deliverables will be considered complete and ready for further development work. When a deliverable is base lined, any further changes are controlled. Errors in a deliverable are not considered defects until after the deliverable is base lined.</p> <p>iii). <b>Defect Discovery</b> - Identification and reporting of defects for development team acknowledgment. A defect is only termed discovered when it has been documented and acknowledged as a valid defect by the development team member(s) responsible for the component(s) in error.</p> <p>iv). <b>Defect Resolution</b> - Work by the development team to prioritize, schedule and fix a defect, and document the resolution. This also includes notification back to the tester to ensure that the resolution is verified.</p>	
2.		<b>Attempt any FOUR of the following:</b>	<b>16Marks</b>
	(a)	<b>What is Boundary Value Analysis? List any three guidelines for boundary value analysis.</b>	<b>4M</b>
	<b>Ans:</b>	<p>Most of the defects in software products hover around conditions and boundaries. By conditions, we mean situations wherein, based on the values of various variables, certain actions would have to be taken. By boundaries, we mean “limits” of values of the various variables.</p> <p>This is one of the software testing technique in which the test cases are designed to include values at the boundary. If the input data is used within the boundary value limits, then it is said to be Positive Testing. If the input data is picked outside the boundary value limits, then it is said to be Negative Testing. Boundary value analysis is another black box test design technique and it is used to find the errors at boundaries of input domain rather than finding those errors in the center of input.</p> <p>Each boundary has a valid boundary value and an invalid boundary value. Test cases are designed based on the both valid and invalid boundary values. Typically, we\ choose one test case from each boundary.</p> <p>There are three guidelines for boundary value analysis :</p> <ol style="list-style-type: none"> <li>1) One test case for exact boundary values of input domains.</li> <li>2) One test case for just below boundary value of input domains .</li> <li>3) One test case for just above boundary values of input domains .</li> </ol> <p>Some examples of Boundary value analysis concept are:</p> <p>One test case for exact boundary values of input domains each means 1 and 100.</p>	<p><b>(Definition: 1mark, Explanation :3 marks)</b></p>



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

One test case for just below boundary value of input domains each means 0 and 99.  
One test case for just above boundary values of input domains each means 2 and 101.  
For Example: A system can accept the numbers from 1 to 10 numeric values. All other numbers are invalid values. Under this technique, boundary values 0, 1,2, 9,10,11 can be tested.  
· Another Example is in exam has a pass boundary at 40 percent, merit at 75 percent and distinction at 85 percent. The Valid Boundary values for this scenario will be as follows:  
49, 50 - for pass  
74, 75 - for merit  
84, 85 - for distinction  
Boundary values are validated against both the valid boundaries and invalid boundaries. The Invalid Boundary Cases for the above example can be given as follows  
0 - for lower limit boundary value  
101 - for upper limit boundary value  
Boundary value analysis is a black box testing and is also applies to white box testing.  
Internal data structures like arrays, stacks and queues need to be checked for boundary or limit conditions; when there are linked lists used as internal structures, the behavior of the list at the beginning and end have to be tested thoroughly.  
Boundary value analysis help identify the test cases that are most likely to uncover defects.

(b) Describe how to select testing tool.

4M

Ans: **{{\*\* Note: Criteria or guidelines can be considered as an answer\*\*}}**

Criteria for Selecting Test Tools:

The Categories for selecting Test Tools are,

1. Meeting requirements;
  2. Technology expectations;
  3. Training/skills;
  4. Management aspects.
1. Meeting requirements-

There are plenty of tools available in the market but rarely do they meet all the requirements of a given product or a given organization. Evaluating different tools for different requirements involve significant effort, money, and time. Given of the plethora of

(Any 4  
Criteria : 4  
marks)  
  
OR  
  
(Any 4  
Guidelines:4  
marks)



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

choice available, huge delay is involved in selecting and implementing test tools.

**2. Technology expectations-**

Test tools in general may not allow test developers to extend/modify the functionality of the framework. So extending the functionality requires going back to the tool vendor and involves additional cost and effort. A good number of test tools require their libraries to be linked with product binaries.

**3. Training/skills-**

While test tools require plenty of training, very few vendors provide the training to the required level. Organization level training is needed to deploy the test tools, as the user of the test suite are not only the test team but also the development team and other areas like configuration management.

**4. Management aspects-**

A test tool increases the system requirement and requires the hardware and software to be upgraded. This increases the cost of the already- expensive test tool.

**OR**

Guidelines for selecting a tool:

1. The tool must match its intended use. Wrong selection of a tool can lead to problems like lower efficiency and effectiveness of testing may be lost.

2. Different phases of a life cycle have different quality-factor requirements. Tools required at each stage may differ significantly.

3. Matching a tool with the skills of testers is also essential. If the testers do not have proper training and skill then they may not be able to work effectively.

4. Select affordable tools. Cost and benefits of various tools must be compared before making final decision.

5. Backdoor entry of tools must be prevented. Unauthorized entry results into failure of tool and creates a negative environment for new tool introduction.

**(c) Prepare and write four test cases for Library Management System of college.**

**4M**

**Ans:**

Sr. No	Test Case_ID	Test Case Objectives	Pre-requisites	Steps	Input data	Expected Result	Actual Result	Status
1	TC_1	To issue book	Book should	1.Request for name of the	Account	Book should be issued on	Book is issued	Pass

**(4 marks for 4 Test cases. Any other relevant test cases can be considered )**



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

				be available	book 2.Mention author name 3.Mention publication 4.Give account no.	no.	the students account	on the student's account		
	2	TC_2	To request other books which are unavailable in the stock	Name and author of the other books should be known	1.Request for name of the book of out of stock books 2.Mention author name 3.Mention publication	NA	Books should be seen in library erp	Books are made available in erp	Pass	
	3	TC_3	To return book	Details of issue of book should be seen	1.Give account no 2.Check for penalty if any 3.Return the book	Account no.	Update should be done in the library erp	Erp is updated	Pass	
	4	TC_4	To clear library account of a student	No issued books should be in the account	1.Give account no. 2.Check erp for clearance	Account no.	Account of the student should be clear	Account of the student is clear	Pass	
	(d)	Give any four differences between Manual & Automated testing.								4M
	Ans:	Manual testing				Automated testing				(1mark for each difference)
		1)Manual Testing is not accurate at all times due to human error, hence it is less reliable				1) Automated testing is more reliable, as it is performed by tools and/or scripts.				
		2)It is time consuming, taking up human resources				2)It is executed by software tools, so it is significantly faster than a manual approach				
		3)Investment is required for human				3) Investment is required for testing tools.				



17624



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

**Module-Interface Defects:** Module-interface defects are about communication problem between various modules. If one module gives some parameters which are not recognized by another, it creates module-interface defects.

**System-Interface Defects:** System-interface defects may be generated when application communication with environmental factors is hampered system may not be able to recognize Inputs coming from the environment, or may not be able to give outputs which can be used by the environment.

**User-Interface Defects:** User-interface defects may be a part of system-interface defects where the other system working with the application is a human being. User-interface defects may be a part of navigation, look and feel type of defects which affect usability of an application.

OR

**Requirements Defects:**

Requirement related defects arise in a product when one fails to understand what is required by the customer.

These defects may be due to customer gap, where the customer is unable to define his requirements, or producer gap, where developing team is not able to make a product as per requirements.

Defects injected in early phases can persist and be very difficult to remove in later phases.

Since any requirements documents are written using natural language representation, there are very often occurrences of ambiguous, contradictory, unclear, redundant and imprecise requirements.

Specifications are also developed using natural language representations.

**Design Defects:**

Design defects occur when system components, interactions between system components, interactions between the outside software/hardware or users are incorrectly designed.

This covers in the design of algorithms, control, logic/ data elements, module interface descriptions and external software/hardware/user interface descriptions.

Design defects generally refer to the way of design creation or its usage while creating a product. The customer may or may not be in a position to understand these defects, if structures are not correct.

They may be due to problems with design creation and implementation during software development life cycle.

( f )

**Explain Equivalence partitioning with respect to equivalence classes.**

4M





SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

	<p><b>Ans:</b> Equivalence partitioning is a software technique that involves identifying a small set of representative input values that produce as much different output condition as possible.</p> <p>This reduces the number of permutation &amp; combination of input, output values used for testing, thereby increasing the coverage and reducing the effort involved in testing.</p> <p>The set of input values that generate one single expected output is called a partition. When the behavior of the software is the same for a set of values, then the set is termed as equivalence class or partition.</p> <p>Example: An insurance company that has the following premium rates based on the age group.</p> <p>A life insurance company has base premium of \$0.50 for all ages. Based on the age group, an additional monthly premium has to pay that is as listed in the table below. For example, a person aged 34 has to pay a premium=\$0.50 +\$ 1.65=\$2.15</p> <table><tr><th>Age group</th><th>Additional Premium</th></tr><tr><td>Under 35</td><td>\$ 1.65</td></tr><tr><td>35-59</td><td>\$ 2.87</td></tr><tr><td>60+</td><td>\$ 6.00</td></tr></table> <p>Based on the equivalence portioning technique, the equivalence partitions that are based on age are given below:</p> <p>Below 35 years of age (valid input)</p> <p>Between 35 and 59 years of age (valid input)</p> <p>Above 6 years of age (valid input)</p> <p>Negative age (invalid input)</p> <p>Age as 0(invalid input)</p> <p>Age as any three-digit number (valid input)</p>	Age group	Additional Premium	Under 35	\$ 1.65	35-59	\$ 2.87	60+	\$ 6.00	<p><b>(Explanation: 2 marks, Example:2 marks)</b></p>
Age group	Additional Premium									
Under 35	\$ 1.65									
35-59	\$ 2.87									
60+	\$ 6.00									
3.	<p><b>Attempt any FOUR of the following:</b></p>	<p><b>16Marks</b></p>								
	<p><b>(a) Explain any one type of structural white box testing in detail.</b></p>	<p><b>4M</b></p>								
	<p><b>Ans:</b> <b>1. Data Flow (Code Functional Testing)</b></p> <p>i. Data flow coverage involves tracking a piece of data completely through the software.</p> <p>ii. At the unit test level this would just be through an individual module or function.</p> <p>iii. The same tracking could be done through several integrated modules or even through the entire software product—although it would be more time consuming to do so.</p> <p>iv. During data flow, the check is made for the proper declaration of variables declared and the loops used are declared and used properly.</p> <p>For example</p> <p>1. #include&lt;stdio.h&gt;</p> <p>2. void main()</p>	<p><b>(Any of these methods can be considered for marking.</b></p> <p><b>One method with description</b></p>								



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

```
3. {  
4. int i , fact= 1, n;  
5. printf("enter the number ");  
6. scanf("%d",&n);  
7. for(i =1 ;i <=n;i++)  
8. fact = fact * i;  
9. printf ("the factorial of a number is %d", fact);  
10. }
```

**2. Data Coverage (Code Coverage Testing)**

- i. The logical approach is to divide the code just as you did in black-box testing into its data and its states (or program flow).
- ii. By looking at the software from the same perspective, you can more easily map the white-box information you gain to the black-box cases you've already written.
- iii. Consider the data first. Data includes all the variables, constants, arrays, data structures, keyboard and mouse input, files and screen input and output, and I/O to other devices such as modems, networks, and so on.

For example

```
1. #include<stdio.h>  
2. void main()  
3. {  
4. int i , fact= 1, n;  
5. printf("enter the number ");  
6. scanf("%d",&n);  
7. for(i =1 ;i <=n;i++)  
8. fact = fact * i;  
9. printf ("the factorial of a number is %d", fact);  
10. }
```

The declaration of data is complete with the assignment statement and the variable declaration statements. All the variable declared are properly utilized.

**3. Program Statements and Line Coverage (Code Complexity Testing)**

- i. The most straightforward form of code coverage is called statement coverage or line coverage.
- ii. If you're monitoring statement coverage while you test your software, your goal is to make sure that you execute every statement in the program at least once.
- iii. With line coverage the tester tests the code line by line giving the relevant output.

For example

```
1. #include<stdio.h>  
2. void main()  
3. {  
4. int i , fact= 1, n;  
5. printf("enter the number ");  
6. scanf("%d", &n);  
7. for(i =1 ;i <=n; i++)  
8. fact = fact * i;  
9. printf ("the factorial of a number is %d", fact);  
10. }
```

:4 marks)



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

**4. Branch Coverage (Code Complexity Testing)**

- i. Attempting to cover all the paths in the software is called path testing.
- ii. The simplest form of path testing is called branch coverage testing.
- iii. To check all the possibilities of the boundary and the sub boundary conditions and it's branching on those values.
- iv. Test coverage criteria requires enough test cases such that each condition in a decision takes on all possible outcomes at least once, and each point of entry to a program or subroutine is invoked at least once.
- v. Every branch (decision) taken each way, true and false.
- vi. It helps in validating all the branches in the code making sure that no branch leads to abnormal behavior of the application.

**5. Condition Coverage (Code Complexity Testing)**

- i. Just when you thought you had it all figured out, there's yet another Complication to path testing.
- ii. Condition coverage testing takes the extra conditions on the branch statements into account.

**(b) Give any four difference between alpha and beta testing.**

**4M**

**Ans:**

Alpha Testing	Beta Testing
Performed at developer's site	Performed at end user's site
Performed in controlled environment as developer is present	Performed in uncontrolled environment as developer is not present
Less probability of finding of error as it is driven by developers.	High probability of finding of error as end user can use it the way he wants.
It is not considered as live application	It is considered as live application
It is done during implementation phase of software	It is done as pre-release of software
Less time consuming as developer can make necessary changes in given time.	More time consuming. As user has to report bugs if any via appropriate channel.

**OR**

Alpha Testing	Beta Testing (Field Testing)
1. It is always performed by the developers at the software development site.	1. It is always performed by the customers or end users at their own site.
2. Sometimes it is also performed by Independent Testing Team.	2. It is not performed by Independent Testing Team.

**(Any valid four differences : 4 marks, 1 mark each)**



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

		3. Alpha Testing is not open to the market and public	3. Beta Testing is always open to the market and public.	
		4. It is conducted for the software application and project.	4. It is usually conducted for software product.	
		5. It is always performed in <b>Virtual Environment</b> .	5. It is performed in <b>Real Time Environment</b> .	
		6. It is always performed within the organization.	6. It is always performed outside the organization.	
		8. Alpha Testing is definitely performed and carried out at the developing organizations location with the involvement of developers.	8. Beta Testing (field testing) is performed and carried out by users or you can say people at their own locations and site using customer data.	
		9. It comes under the category of both White Box Testing and Black Box Testing.	9. It is only a kind of Black Box Testing.	
		10. Alpha Testing is always performed at the time of Acceptance Testing when developers test the product and project to check whether it meets the user requirements or not.	10. Beta Testing is always performed at the time when software product and project are marketed.	
		11. It is always performed at the developer's premises in the absence of the users.	11. It is always performed at the user's premises in the absence of the development team.	
		12. Alpha Testing is not known by any other different name.	12 Beta Testing is also known by the name <b>Field Testing</b> means it is also known as field testing.	
		13. It is considered as the User Acceptance Testing (UAT) which is done at developer's area.	13. It is also considered as the User Acceptance Testing (UAT) which is done at customers or users area.	
	(c)	<b>What is external test standard? Give any three types of external standards.</b>		<b>4M</b>
	<b>Ans:</b>	<p><b>External Standard:</b> These are the standards made by an entity external to an organization. These standards are standards that a product should comply with, are externally visible and are usually stipulated by external parties.</p> <p>The three types of external standards are :</p> <ul style="list-style-type: none"> <li>Customer standard: refer to something defined by the customer as per his /her business requirement for the given product.</li> <li>National Standard: refer to something defined by the regulatory entities of the</li> </ul>		<b>(External test Standard :1 mark, any 3 types of external test standards :</b>



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

		<p>country where the supplier / customer reside.</p> <ul style="list-style-type: none"> <li>International Standard: are defined at international level and these are applicable to all customers across the globe.</li> </ul> <p>Some IEEE standards devoted for software :</p> <p>By Different Organizations are:</p> <p>ISO - International Organization for Standardization</p> <p>SPICE - Software Process Improvement and Capability Determination</p> <p>NIST - National Institute of Standards and Technology</p> <p>DoD - Department of Defense</p>	<b>3 marks)</b>
	<b>(d)</b>	<b>Describe how to perform usability and GUI testing.</b>	<b>4M</b>
	<b>Ans:</b>	<p><b>Usability testing:</b> it is a non-functional testing technique that is a measure of how easily the system can be used by end users. It is difficult to evaluate and measure but can be evaluated based on the below parameters:</p> <ul style="list-style-type: none"> <li>Levels of Skill required learn/use the software. It should maintain the balance for both novice and expert user.</li> <li>Time required to get used to in using the software.</li> <li>The measure of increase in user productivity if any.</li> <li>Assessment of a user's attitude towards using the software.</li> </ul> <p>GUI testing : <b>GUI testing</b> is the process of testing the system's Graphical User Interface of the Application Under Test. GUI testing is the process of ensuring proper functionality of the graphical user interface (GUI ) for a given application and making sure it conforms to its written specifications.</p> <p>In addition to functionality, GUI testing evaluates design elements such as layout, colors, fonts, font sizes, labels, text boxes, text formatting, captions, buttons, lists, icons, links and content. GUI testing processes can be either manual or automatic, and are often performed by third -party companies, rather than developers or end users.</p> <p>GUI is what user sees. A user does not see the source code. The interface is visible to the user. Especially the focus is on the design structure, images that they are working properly or not.</p> <p style="text-align: center;"><b>OR</b></p> <p><b>GUI Testing.</b></p> <ol style="list-style-type: none"> <li>GUI testing is a testing technique in which the application's user interface is tested whether the application performs as expected with respect to user interface behavior.</li> <li>GUI Testing includes the application behavior towards keyboard and mouse movements and how different GUI objects such as toolbars, buttons, menu bars, dialog boxes, edit fields, lists, behaviour to the user input. GUI Testing Guidelines <ul style="list-style-type: none"> <li>➤ Check Screen Validations</li> <li>➤ Verify All Navigations</li> <li>➤ Check usability Conditions</li> <li>➤ Verify Data Integrity</li> <li>➤ Verify the object states</li> </ul> </li> </ol>	<b>(Usability testing :2 marks, GUI testing : 2 marks)</b>



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

	➤ Verify the date Field and Numeric Field Formats	
(e)	<b>Give any two advantages and any two disadvantages of using testing tools.</b>	<b>4M</b>
<b>Ans:</b>	<p><b>Advantages of using testing tools :</b></p> <ol style="list-style-type: none"> <li>1. Speed. The automation tools tests the software under tests with the very faster speed. There's a vast difference between the speed of user entering the data and the automated tools generating and entering the data required for the testing of the software. Speed of this software also completes the work faster.</li> <li>2. Efficiency. While testers are busy running test cases, testers can't be doing anything else. If the tester have a test tool that reduces the time it takes for him to run his tests, he has more time for test planning and thinking up new tests.</li> <li>3. Accuracy and Precision. After trying a few hundred cases, tester's attention span will wane and he may start to make mistakes. A test tool will perform the same test and check the results perfectly, each and every time.</li> <li>4. Resource Reduction. Sometimes it can be physically impossible to perform a certain test case. The number of people or the amount of equipment required to create the test condition could be prohibitive. A test tool can be used to simulate the real world and greatly reduce the physical resources necessary to perform the testing.</li> <li>5. Simulation and Emulation. Test tools are often used to replace hardware or software that would normally interface to your product. This "fake" device or application can then be used to drive or respond to your software in ways that you choose and ways that might otherwise be difficult to achieve.</li> <li>6. Relentlessness. Test tools and automation never tire or give up. they can keep going and going and on and on without any problem; whereas the tester gets tired to test again and again.</li> </ol> <p style="text-align: center;"><b>OR</b></p> <ul style="list-style-type: none"> <li>• Reduce time of testing</li> <li>• Improve the bugs finding</li> <li>• Deliver the quality software/product</li> <li>• Allow to run tests many times with different data</li> <li>• Getting more time for test planning</li> <li>• Save resources or reduce requirement</li> <li>• It is never tired and expert person can work at a time many tools.</li> </ul> <p><b>Disadvantages of using testing tools :</b></p> <ul style="list-style-type: none"> <li>• It's more expensive to automate. Initial investments are bigger than manual testing</li> <li>• Manual tests can be very time consuming.</li> <li>• You cannot automate everything; some tests still have to be done manually.</li> <li>• You cannot rely on testing tools always.</li> </ul>	<p><b>( Any two advantage : 2 marks, Any two disadvantage : 2 marks )</b></p>
(f)	<b>Explain Defect template.</b>	<b>4M</b>
<b>Ans:</b>	<b>Defect template:</b> A defect report documents an anomaly discovered during testing. It	<b>(Defect</b>



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

includes all the information needed to reproduce the problem, including the author, release/build number, open/close dates, problem area, problem description, test environment, defect type, how it was detected, who detected it, priority, severity, status, etc.

After uncovering a defect (bug), testers generate a formal defect report. The purpose of a defect report is to state the problem as clearly as possible so that developers can replicate the defect easily and fix it.

**DEFECT REPORT TEMPLATE**

In most companies, a defect reporting tool is used and the elements of a report can vary. However, in general, a defect report can consist of the following elements.

<b>ID</b>	Unique identifier given to the defect. (Usually Automated)
<b>Project</b>	Project name.
<b>Product</b>	Product name.
<b>Release Version</b>	Release version of the product. (e.g. 1.2.3)
<b>Module</b>	Specific module of the product where the defect was detected.
<b>Detected Build Version</b>	Build version of the product where the defect was detected (e.g. 1.2.3.5)
<b>Summary</b>	Summary of the defect. Keep this clear and concise.
<b>Description</b>	Detailed description of the defect. Describe as much as possible but without  Repeating anything or using complex words. Keep it simple but comprehensive.
<b>Steps to Replicate</b>	Step by step description of the way to reproduce the defect. Number the steps.
<b>Actual Result</b>	The actual result you received when you followed the steps.
<b>Expected Results</b>	The expected results.
<b>Attachments</b>	Attach any additional information like screenshots and logs.
<b>Remarks</b>	Any additional comments on the defect.
<b>Defect Severity</b>	Severity of the Defect.

**template:  
description  
:1 mark,  
template : 3  
marks )**





**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

		<table><tr><td><b>Defect Priority</b></td><td>Priority of the Defect.</td></tr><tr><td><b>Reported By</b></td><td>The name of the person who reported the defect.</td></tr><tr><td><b>Assigned To</b></td><td>The name of the person that is assigned to analyze/fix the defect.</td></tr><tr><td><b>Status</b></td><td>The status of the defect.</td></tr><tr><td><b>Fixed Build Version</b></td><td>Build version of the product where the defect was fixed (e.g. 1.2.3.9)</td></tr></table>	<b>Defect Priority</b>	Priority of the Defect.	<b>Reported By</b>	The name of the person who reported the defect.	<b>Assigned To</b>	The name of the person that is assigned to analyze/fix the defect.	<b>Status</b>	The status of the defect.	<b>Fixed Build Version</b>	Build version of the product where the defect was fixed (e.g. 1.2.3.9)		
<b>Defect Priority</b>	Priority of the Defect.													
<b>Reported By</b>	The name of the person who reported the defect.													
<b>Assigned To</b>	The name of the person that is assigned to analyze/fix the defect.													
<b>Status</b>	The status of the defect.													
<b>Fixed Build Version</b>	Build version of the product where the defect was fixed (e.g. 1.2.3.9)													
4.	(A)	Attempt any THREE of the following:	12Marks											
	(a)	Give any four advantages of test planning.	4M											
	Ans:	<p><b>Advantages of test planning are :</b></p> <ol style="list-style-type: none"><li>1. Work involved in test planning and setup pays in the long term. It gives insight testing activity completely. One knows scope and deliverables of test plan execution.</li><li>2. Test plan describes the way in which testing team will show whether software work correctly as per requirements and the acceptance criteria as defined by customer or development team with customer. It defines various objectives for testing to measure its performance and coverage offered.</li><li>3. Test plan addresses various levels of testing such as unit testing module testing, System testing, integration testing, black box testing as well as white box testing. Some time there may be single master test plan with several child test plan at each level for a number of a small plans or one monolithic test plan covering every aspect of testing.</li><li>4. Test plan explain who does testing. Why test are performed how test are conducted and when tests are scheduled (calendar date and milestone). It defines various criteria such as entry criteria, exit criteria, suspension criteria and resumption criteria at various stages of testing.</li><li>5. Test plan must contain procedures, environment and tools necessary to implement an orderly, controlled process for test execution, defect tracking, coordination of rework and configuration, and change control.</li></ol> <p style="text-align: center;"><b>OR</b></p> <p>It is strategic document describes how to perform a task in as effective, efficient and optimized manner. It also specifies, the scope and objectives for testing Primary tasks in testing are :.( Explanation of followings related to):</p> <ol style="list-style-type: none"><li>1. Scope of testing</li><li>2. Set objectives of test planning</li><li>3. Methodology for testing</li></ol>	(4 advantages of test planning: 4 marks, 1 mark each )											





SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

		4. Requirement, 5. Entry and exit criteria 6. Develop Test matrix 7. Test estimation and administrative component 8. Write and execute test cases. 9. Deciding Criteria for test-to-pass, test-to-fail, 10. Schedule of testing 11. The main purpose of test planning is to show whether software is correct as per requirement.	
	(b)	<b>What do you mean by Software Metrics? List any three types of metrics.</b>	<b>4M</b>
	<b>Ans:</b>	<p>Software metrics: Metrics are necessary to provide measurements of such qualities. Metrics can also be used to gauge the size and complexity of software and hence are employed in project management and cost estimation.</p> <p style="text-align: center;"><b>OR</b></p> <p>A Metric is a quantitative measure of the degree to which a system, system component, or process possesses a given attribute. Metrics can be defined as “STANDARDS OF MEASUREMENT”. Software Metrics are used to measure the quality of the project. Simply, Metric is a unit used for describing an attribute. Metric is a scale for measurement.</p> <p>Types of software metrics are:</p> <ol style="list-style-type: none"><li>1. Project metrics</li><li>2. Progress metrics</li><li>3. Productivity metrics</li></ol>	<b>(Software Metrics:3 marks, listing:1 mark)</b>
	(c)	<b>What are different techniques for finding defects? Explain in detail.</b>	<b>4M</b>
	<b>Ans:</b>	<p><b>Static Techniques:</b> Static techniques of quality control define checking the software product and related artifacts without executing them. It is also termed ‘desk checking/verification /white box testing’. It may include reviews, walkthroughs, inspection, and audits. Here the work product is reviewed by the reviewer with the help of a checklist, standards, any other artifact, knowledge and experience, in order to locate the defect with respect to the established criteria. Static technique is so named because it involves no execution of code, product, documentation, etc. This technique helps in establishing conformance to requirements view.</p> <p><b>Dynamic Techniques:</b> Dynamic testing is a validation technique which includes dummy or actual execution of work products to evaluate it with expected behavior. It includes black box testing methodology such as system testing and unit testing. The testing methods evaluate the product with respect to requirements defined, designs created and mark it as ‘pass’ or ‘fail’. This technique establishes ‘fitness for use’ view.</p> <p><b>Operational Techniques:</b> Operational techniques typically include auditing work products and projects to understand whether the processes defined for development /testing are being followed correctly or not, and also whether they are effective or not. It also includes</p>	<b>(Different techniques listing:1 mark, Explanation : 3 marks)</b>



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

revisiting the defects before and after fixing and analysis. Operational technique may include smoke testing and sanity testing of a work product.

OR

Different techniques to find the defects are :

- a) Quick Attacks:
- b) Equivalence and Boundary Conditions
- c) Common Failure Modes
- d) State-Transition Diagrams
- e) Use Cases
- f) Code-Based Coverage Models
- g) Regression and High-Volume Test Techniques

**a) Quick Attacks:**

- The quick-attacks technique allows you to perform a cursory analysis of a system in a very compressed timeframe.
- Even without a specification, you know a little bit about the software, so the time spent is also time invested in developing expertise.

**b) Equivalence and Boundary Conditions:**

- Boundaries and equivalence classes give us a technique to reduce an infinite test set into something manageable.
- They also provide a mechanism for us to show that the requirements are "covered".

**c) Common Failure Modes:**

- The heart of this method is to figure out what failures are common for the platform, the project, or the team; then try that test again on this build.
- If your team is new, or you haven't previously tracked bugs, you can still write down defects that "feel" recurring as they occur—and start checking for them.
- The more your team stretches itself (using a new database, new programming language, new team members, etc.), riskier the project will be—and, at the same time, the less valuable this technique will be.

**d) State-Transition Diagrams:**

- Mapping out the application provides a list of immediate, powerful test ideas.
- Model can be improved by collaborating with the whole team to find "hidden" states—transitions that might be known only by the original programmer or specification author.
- Once you have the map, you can have other people draw their own diagrams, and then compare theirs to yours.
- The differences in those maps can indicate gaps in the requirements, defects in the software, or at least different expectations among team members.
- The map you draw doesn't actually reflect how the software will operate; in other words, "the map is not the territory."



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

- Drawing a diagram won't find these differences,
- Like just about every other technique on this list, a state-transition diagram can be helpful, but it's not sufficient by itself to test an entire application.

**e) Use Cases:**

Use cases and scenarios focus on software in its role to enable a human being to do something. Use cases and scenarios tend to resonate with business customers, and if done as part of the requirement process, they sort of magically generate test cases from the requirements.

**f) Code-Based Coverage Models:**

Imagine that you have a black-box recorder that writes down every single line of code as it executes. Programmers prefer code coverage. It allows them to attach a number— an actual, hard, real number, such as 75%—to the performance of their unit tests, and they can challenge themselves to improve the score.

- Customer-level coverage tools are expensive, programmer-level tools that tend to assume the team is doing automated unit testing and has a continuous-integration server and a fair bit of discipline.
- After installing the tool, most people tend to focus on statement coverage—the least powerful of the measures.

**g) Regression and High-Volume Test Techniques:**

- People spend a lot of money on regression testing, taking the old test ideas described above and rerunning them over and over.
- This is generally done with either expensive users or very expensive programmers spending a lot of time writing and later maintaining those automated tests.

**ii. Weaknesses**

- Building a record/playback/capture rig for a GUI can be extremely expensive, and it might be difficult to tell whether the application hasn't broken, but has changed in a minor way.

**(d) Describe the factors considered to decide test strategy or test approach.**

**4M**

**Ans: Factors considered to decide test strategy or test approach :**

(1) Analytical Approach: This test approach is based on an analysis of some factor, which strongly effects the testing environment, e.g., Requirements might be analyzed in order to design a test approach such that the most important requirements get tested first and test cases for other requirements get designed / executed later. Another exercise commonly performed is Risk Analysis, where tests are designed and prioritized such that the critical risks get eliminated at the earliest possible time. This approach is an example of preventive test approach, since we are analyzing and prioritizing test cases early on, based on an analysis of some factor contributing to the testing environment.

(2) Model-based approach: the tests are designed based on some mathematical or stochastic (statistical) model of the object functionality. As an example, if a model predicts the failure

**(Any 4 factors considered to decide test strategy or test approach:4 marks, 1 mark each)**



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

rates of a particular system (or software) under some conditions, and the failure rate of our product is as stipulated by the model under the specified conditions, then our product is assumed to be working fine. This approach pays emphasis on identification and selection of the appropriate model during the early stages of SDLC and is a preventive test approach.

(3) Methodical Approach: This test approach depends heavily upon following a pre-determined method to perform testing. The method used can vary widely, ranging from adherence to certain checklists to error guessing and experience-based approaches. Here, tests are designed, executed and implemented in accordance with the selected method. Actual testing effort may get started early on or later during the SDLC when following this testing approach.

(4) Process-or Standard-compliant Approach: As the name suggests, this testing approach recommends designing and creation of test assets based on some externally developed industry standard, e.g., Tests could be designed based on IEEE 829 standards. As an alternative, one of the agile methodologies, such as Extreme Programming (XP), might be adopted. Again, actual testing effort might get started early on or later during the SDLC when this approach is selected.

(5) Dynamic (Heuristic) Approach: This test approach involves performing heuristic testing. Exploratory testing is a good example of the type of testing performed, when this approach is followed. Here, the tests are designed and executed simultaneously, and hence, it is a reactive test approach.

(6) Regression-averse Approach: This approach recommends designing tests such that regression defects get detected at the earliest. This may involve extensive automation of functional regression tests as well as re-use of existing test material.

**(B) Attempt any ONE of the following:**

**6 Marks**

**(a) Differentiate between verification and validation.**

**6M**

**Ans:**

Verification	Validation
Are we building the system right?	Are we building the right system?
Verification is the process of evaluating products of a development phase to find out whether they meet the specified requirements.	Validation is the process of evaluating software at the end of the development process to determine whether software meets the customer expectations and requirements.
The objective of Verification is to make	The objective of Validation is to make sure

**(Any valid 6 differences: 6 marks, 1 mark each)**



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

		<p>sure that the product being develop is as per the requirements and design specifications.</p> <p>Following activities are involved in Verification: Reviews, Meetings and Inspections.</p> <p>Verification is carried out by QA team to check whether implementation software is as per specification document or not.</p> <p>Execution of code comes under Verification.</p> <p>Verification process explains whether the outputs are according to inputs or not.</p> <p>Verification is carried out before the Validation</p> <p>Following items are evaluated during Verification: Plans, Requirement Specifications, Design Specifications, Code, Test Cases etc</p> <p>Cost of errors caught in Verification is less than errors found in Validation.</p>	<p>that the product actually meet up the user's requirements, and check whether the specifications were correct in the first place.</p> <p>Following activities are involved in Validation: Testing like black box testing, white box testing, gray box testing etc.</p> <p>Validation is carried out by testing team.</p> <p>Execution of code does not comes under Validation.</p> <p>Validation process describes whether the software is accepted by the user or not.</p> <p>Validation activity is carried out just after the Verification.</p> <p>Following item is evaluated during Validation: Actual product or Software under test.</p> <p>Cost of errors caught in Validation is more than errors found in Verification.</p>	
	<b>(b)</b>	<b>Distinguish between white box testing and block box testing.(any six)</b>		<b>6M</b>
	<b>Ans:</b>	<p><b>White box testing</b></p> <p>This needs the knowledge of software in detail.</p> <p>It is also called as transparent box or glass box testing.</p> <p>This can be performed by only developers and professional testers.</p>	<p><b>Black box testing</b></p> <p>This does not need the knowledge of software in detail</p> <p>It is also called as opaque box, dark box testing</p> <p>This can be performed by end users or anyone.</p>	<b>(White box testing and black box testing any 6 differences :6 marks, 1 mark each)</b>



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

		<p>The testing is proper here with respect to the domain ,data etc.</p> <p>It is suited for algorithm testing.</p> <p>It is exhaustive and time consuming.</p> <p>It is a structural testing of a system.</p>	<p>The testing is only by trial and error methods.</p> <p>It is not suited for algorithm testing.</p> <p>It is least exhaustive and least time consuming.</p> <p>It is a behavioral testing of a system.</p>	
<b>5.</b>		<b>Attempt any TWO of the following:</b>		<b>16Marks</b>
	<b>(a)</b>	<b>Describe test reporting in detail. How to prepare a summary report?</b>		<b>8M</b>
	<b>Ans:</b>	<p>Test reporting is a means of achieving communication through the testing cycle. There are 3 types of test reporting.</p> <p><b>1. Test incident report:</b></p> <p>A test incident report is communication that happens through the testing cycle as and when defects are encountered .A test incident report is an entry made in the defect repository each defect has a unique id to identify incident .The high impact test incident are highlighted in the test summary report.</p> <p><b>2. Test cycle report:</b></p> <p>A test cycle entails planning and running certain test in cycle , each cycle using a different build of the product .As the product progresses through the various cycles it is expected to stabilize.Test cycle report gives</p> <ol style="list-style-type: none"> <li>1. A summary of the activities carried out during that cycle.</li> <li>2. Defects that are uncovered during that cycle based on severity and impact</li> <li>3. Progress from the previous cycle to the current cycle in terms of defect fixed</li> <li>4. Outstanding defects that not yet to be fixed in cycle</li> <li>5. Any variation observed in effort or schedule</li> </ol> <p><b>3 Test summary report:</b></p> <p>The final step in a test cycle is to recommend the suitability of a product for release. A report that summarizes the result of a test cycle is the test summary report.</p> <p>There are two types of test summary report:</p> <ol style="list-style-type: none"> <li>1. Phase wise test summary, which is produced at the end of every phase.</li> <li>2. Final test summary report.</li> </ol>		<b>(Explanation: 4 marks, Summary report:4 marks)</b>



**SUMMER – 18 EXAMINATION**

**Subject Name: Software Testing**

**Model Answer**

Subject Code:

**17624**

A Summary report should present

1. Test Summary report Identifier

2 Description: Identify the test items being reported in this report with test id

3 Variances: Mention any deviation from test plans, test procedures, if any.

4 Summary of results: All the results are mentioned here with the resolved incidents and their solutions.

5 Comprehensive assessment and recommendation for release should include: Fit for release assessment and recommendation of release.

**OR**

- A test report is any descriptions, explanation or justification the status of a test project.
- A comprehensive test report is all of those things together.
- Test reporting is a means of achieving this communication.

Types of reports that might be generated for a classical waterfall based project include:

- summary report for the phase;
- technical review report;
- walkthrough report;
- audit report.
- test report

**Summary Report:**

- The phase may be between five and ten pages in length.
- It would cover the major points drawn out from the testing.
- It is suggested that the report be structured with a single highlight sheet at the front based on a small commentary and a table that will show key areas where successful testing has been completed and areas where concern must be documented.
- The report should also address any issues that arose from external dependencies that may have affected the schedule.

**(b) With the help of an example, explain Graph-based testing.**

**8M**

**Ans: {{\*\*Note:- Any other relevant example shall be considered\*\* }}**

**(Explanation:  
4 marks,**





SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

- i. Graph Based Testing is also called as State Based Testing
- ii. It provides a framework for model based testing.
- iii. Black-box methods based on the nature of the relationships (links) among the program objects (nodes), test cases are designed to traverse the entire graph.
- iv. Transaction flow testing – nodes represent steps in some transaction and links represent logical connections between steps that need to be validated.
- v. Finite state modeling – nodes represent user observable states of the software and links represent transitions between states.
- vi. Data flow modeling – nodes are data objects and links are transformations from one data object to another.
- vii. Timing modeling – nodes are program objects and links are sequential connections between these objects, link weights are required execution times.

Steps in graph testing:

- i. Build a graph model.
- ii. Identify the test requirements.
- iii. Select test paths to cover those requirements.

In order to design test design cases following steps are used:

- i. Understanding the system
- ii. Identifying states, inputs and guards.
- iii. Create a state graph model of the application.
- iv. Verify whether State graph that we modeled is correct in all details.
- v. Generate sequence of test actions.

Derive test data so that those test paths can be executed.

**Example:**

**Example: 4 marks)**





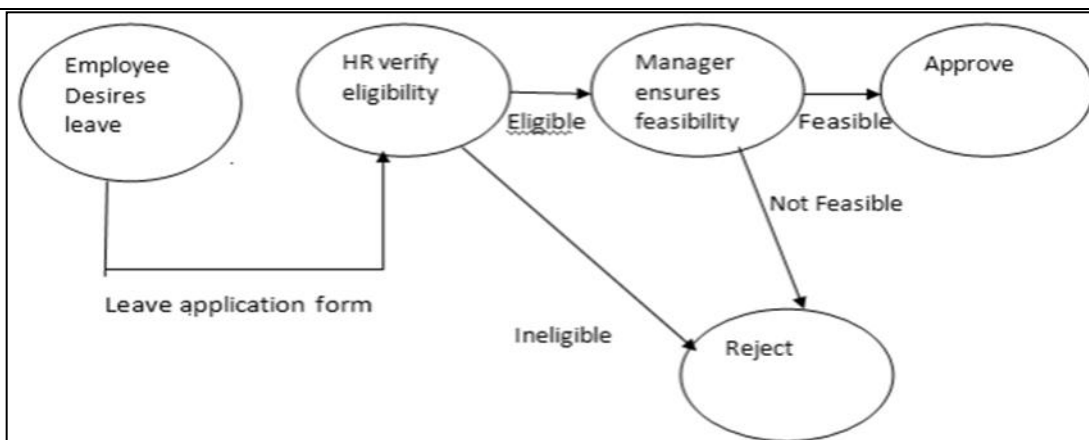
SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624



(c) Explain following concepts related to Integration testing with neat & labeled diagram:

8M

- (i) Top-down testing.  
(ii) Bottom-up testing.

Ans:

- Integration is process by which components are aggregated to create larger components.
- Testing the data flow or interface between two features is known as integration testing.
- It mainly focuses on I/O protocols, parameters passing between different unit's modules and/or system etc.

**Types of Integration testing:**

**1. Top-down Testing:**

In this approach testing is conducted from main module to sub module. If the sub module is not developed a temporary program called STUB is used for simulate the sub module.

**Advantages:**

- Advantageous if major flaws occur toward the top of the program.
- Once the I/O functions are added, representation of test cases is easier.
- Early skeletal Program allows demonstrations and boosts morale.

**Disadvantages:**

- Stub modules must be produced
- Stub Modules are often more complicated than they first appear to be.

(Top down Explanation :3 marks, Any similar diagram: 1 mark, Bottom up Explanation :3 marks, Any similar diagram 1mark)



SUMMER – 18 EXAMINATION

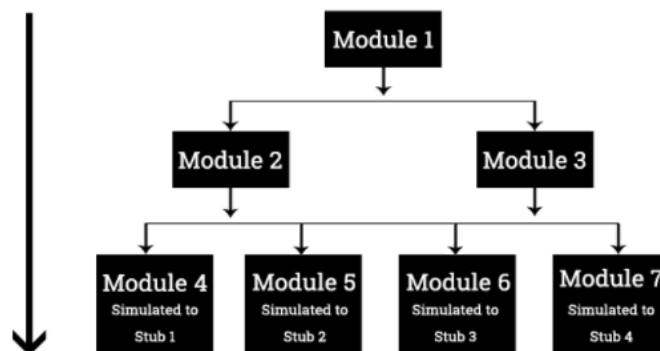
Subject Name: Software Testing

Model Answer

Subject Code:

17624

- Before the I/O functions are added, representation of test cases in stubs can be difficult.
- Test conditions may be impossible, or very difficult, to create.
- Observation of test output is more difficult.
- Allows one to think that design and testing can be overlapped.
- Induces one to defer completion of the testing of certain modules.



**2. Bottom-up testing:**

In this approach testing is conducted from sub module to main module, if the main module is not developed a temporary program called DRIVERS is used to simulate the main module.

**Advantages:**

- Advantageous if major flaws occur toward the bottom of the program.
- Test conditions are easier to create.
- Observation of test results is easier.
- Driver Modules must be produced.
- The program as an entity does not exist until the last module is added.



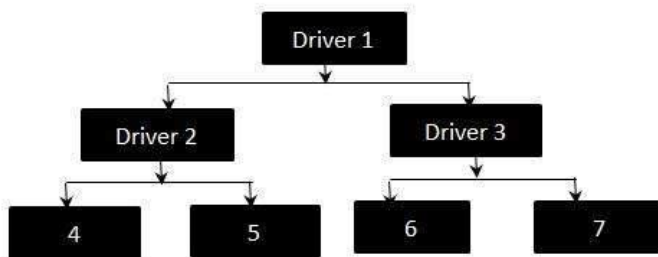
SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624



6.	Attempt any FOUR of the following:	16Marks
(a)	Give any two root causes of defects. Also give any two effects of defects.	4M
Ans:	<p><b>Root of defects:</b></p> <ol style="list-style-type: none"> <li>Miscommunication of requirements introduces error in code.</li> <li>Lack of design Experience.</li> <li>Lack of coding practice.</li> <li>Unrealistic time schedule for development.</li> <li>Multiple changes in the requirements.</li> </ol> <p><b>Effects of defects</b></p> <ol style="list-style-type: none"> <li>A defect is an error in coding or logic that causes a program to fail or to produce incorrect /unexpected results.</li> <li>It is commonly refers to troubles with the software products, with its external behavior or with its internal features.</li> <li>Increased complexity of software.</li> <li>Programming Errors.</li> </ol>	(Any two relevant root causes :2 marks, Any two relevant effects: 2 marks)
(b)	What is Software testing? Give any three objectives of testing.	4M
Ans:	<p><b>Software testing:</b> Software testing is defined as performing Verification and Validation of the Software Product for its correctness and accuracy of working. Software Testing is the process of executing a program with the intent of finding errors. A successful test is one that uncovers an as-yet-undiscovered error. Testing can show the presence of bugs but never their absence. Testing is a support function that helps developers look good by finding their mistakes before anyone else does. Execution of a work product with intent to find a defect.</p>	(Definition :1 mark, Any three objectives :3 marks)



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

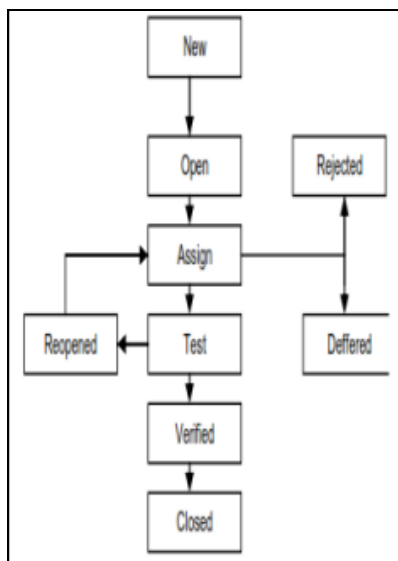
**Objectives of software testing are as follows:**

1. Finding defects which may get created by the programmer while developing the software.
2. Gaining confidence in and providing information about the level of quality.
3. To prevent defects.
4. To make sure that the end result meets the business and user requirements.
5. To ensure that it satisfies the BRS that is Business Requirement Specification and SRS that is System Requirement Specifications.
6. To gain the confidence of the customers by providing them a quality product.

**(c) Explain Defect Life Cycle with diagram.**

**4M**

**Ans:**



**OR**

**(Description of defect life cycle Stages: 2 marks, Proper Labeled Diagram: 2 marks)**



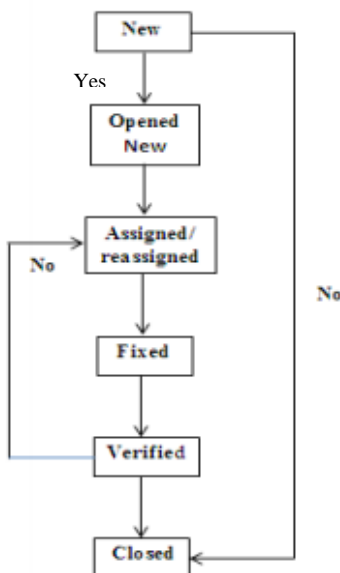
SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624



- **Open:** After a tester has posted a bug, the lead of the tester approves that the bug is genuine and he changes the state as “OPEN”.
- **Assign:** Once the lead changes the state as “OPEN”, he assigns the bug to corresponding developer or developer team. The state of the bug now is changed to “ASSIGN”.
- **Test/Retest:** Once the developer fixes the bug, he has to assign the bug to the testing team for next round of testing. Before he releases the software with bug fixed, he changes the state of bug to “TEST”. It specifies that the bug has been fixed and is released to testing team.// At this stage the tester do the retesting of the changed code which developer has given to him to check whether the defect got fixed or not.
- **Deferred:** The bug, changed to deferred state means the bug is expected to be fixed in next releases. The reasons for changing the bug to this state have many factors. Some of them are priority of the bug may be low, lack of time for the release or the bug may not have major effect on the software.
- **Rejected:** If the developer feels that the bug is not genuine, he rejects the bug. Then the state of the bug is changed to “REJECTED”.
- **Verified:** Once the bug is fixed and the status is changed to “TEST”, the tester tests the bug. If the bug is not present in the software, he approves that the bug is fixed and changes the status to “VERIFIED”.
- **Reopened:** If the bug still exists even after the bug is fixed by the developer, the tester changes the status to “REOPENED”. The bug traverses the life cycle once again.



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

- **Closed:** Once the bug is fixed, it is tested by the tester. If the tester feels that the bug no longer exists in the software, he changes the status of the bug to “CLOSED”. This state means that the bug is fixed, tested and approved.
- **Fixed:** When developer makes necessary code changes and verifies the changes then he/she can make bug status as „Fixed“ and the bug is passed to testing team.
- **Pending retest:** After fixing the defect the developer has given that particular code for retesting to the tester. Here the testing is pending on the testers end. Hence its status is pending retest.

(d) What is Test plan? List test planning activities.

4M

- Ans:**
- i. Test planning is the first activity of test team.
  - ii. If a tester does not plan for testing then it leads to failure.
  - iii. Test plans are defined in framework created by test strategy and established by test policy like any project, the testing also should be driven by a plan.
  - iv. The test plan acts as the anchor for the execution, tracking and reporting of the entire testing project.
  - v. The test plan acts as the anchor for the execution, tracking and reporting of the entire testing project and covers.

(Test plan  
Explanation:  
2 marks, Test  
activities 2  
marks)

**The following are the test plan activities:**

1. To determine the scope and the risks that need to be tested and that are not to be tested.
2. Determine the strategy.
3. Making sure that the testing activities have been included.
4. Deciding Entry exit criteria.
5. Evaluating the test estimate.
6. Planning when and how to test and deciding how the test results will be evaluated and defining
7. The test artifacts delivered as part of test execution.
8. Defining the management information, including the metrics required and defect resolution and risks issues.



SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

9. Ensuring that the test documentation generates repeatable test assets.

(e) Describe compatibility testing with an example.

4M

- Ans:**
- Compatibility Testing is a type of Software testing to check whether your software is capable of running on different hardware, operating systems, applications, network environments.
  - It is a type of the Non-functional testing.
  - There are two types of compatibility testing:
    - Forward Compatibility.
    - Backward Compatibility.

(Compatibility Explanation: 1 mark, Types Explanation : 2 marks, any similar Example/diagram : 1 mark)

**Forward Compatible:**

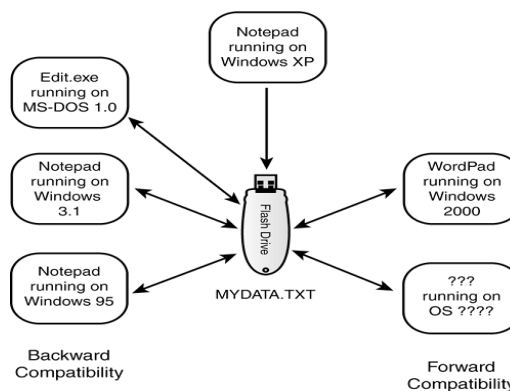
If something is forward compatible, it will work with future versions of the software.

**Backward Compatible:**

If something is backward compatible, it will work with previous versions of the software.

**E.g.**

- Cutting text from a web page and pasting it into document opened in your word processor.
- Saving accounting data from one spreadsheet program and then loading into a completely different spreadsheet program.



(f) Explain Client-Server testing

4 M

SUMMER – 18 EXAMINATION

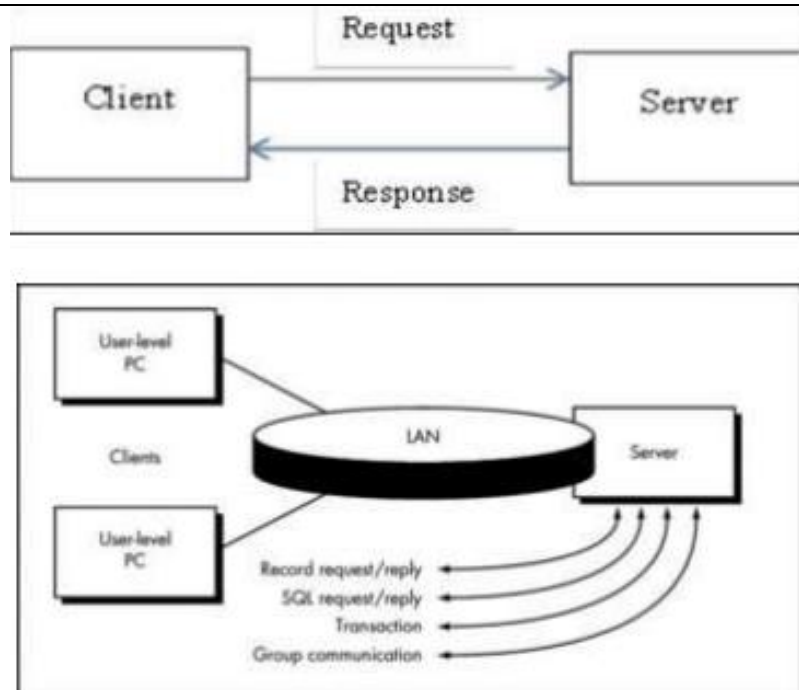
Subject Name: Software Testing

Model Answer

Subject Code:

17624

Ans:



(Diagram :2 marks,  
Explanation : 2 marks)

In Client-server testing there are several clients communicating with the server.

1. Multiple users can access the system at a time and they can communicate with the server.
2. Configuration of client is known to the server with certainty.
3. Client and server are connected by real connection.
4. Testing approaches of client server system:
  1. **Component Testing:** One need to define the approach and test plan for testing client and server individually. When server is tested there is need of a client simulator, whereas testing client a server simulator, and to test network both simulators are used at a time.
  2. **Integration testing:** After successful testing of server, client and network, they are brought together to form system testing.
  3. **Performance testing:** System performance is tested when number of clients is communicating with server at a time. Volume testing and stress testing may be used for





SUMMER – 18 EXAMINATION

Subject Name: Software Testing

Model Answer

Subject Code:

17624

testing, to test under maximum load as well as normal load expected. Various interactions may be used for stress testing.

4. **Concurrency Testing:** It is very important testing for client-server architecture. It may be possible that multiple users may be accessing same record at a time, and concurrency testing is required to understand the behavior of a system in this situation.
5. **Disaster Recovery Business continuity testing:** When the client server are communicating with each other, there exists a possibility of breaking of the communication due to various reasons or failure of either client or server or link connecting them. The requirement specifications must describe the possible expectations in case of any failure.
6. **Testing for extended periods:** In case of client server applications generally server is never shutdown unless there is some agreed Service Level Agreement (SLA) where server may be shut down for maintenance. It may be expected that server is running 24X7 for extended period. One needs to conduct testing over an extended period to understand if service level of network and server deteriorates over time due to some reasons like memory leakage.
7. **Compatibility Testing:** Client server may be put in different environments when the users are using them in production. Servers may be in different hardware, software, or operating system environment than the recommended. Other testing such as security testing and compliance testing may be involved if needed, as per testing and type of system.



Subject Name: Software Testing

SUMMER – 18 EXAMINATION  
Model Answer

Subject Code:

17624