|  |  |  |
| --- | --- | --- |
|  | | |
| logo1 | **PESIT Bangalore South Campus**  Hosur road, 1km before Electronic City, Bengaluru -100  **Department of Computer Science and Engineering** |  |

**INTERNAL ASSESSMENT TEST 2**

**Scheme of Evaluation**

|  |  |
| --- | --- |
| **Date :** 23/09/19 | **Max Marks: 60** |
| **Subject & Code:** Intr. To Software Testing (17CS552) | **Section:** A,B and C |
| **Name of Faculty:** D.sudaroli Vijayakumar | **Time: 11**:30 AM-1:00 PM |

**Note: *Answer FIVE full questions. Selecting One question from each part.***

|  |  |  |
| --- | --- | --- |
|  | **Part I** |  |
| 1a | **What is structural testing? Why to use white box testing when black box testing is used to test conformance of requirements?**  **Answer:**  **Definition of Structural Testing: (2)**  Code based testing or glass box testing or white box testing  Internal structure/program code of the application is considered for generating test cases.  **Reasons for betterness of white box testing: (4)**   1. Dead code is identified 2. Maximum failures can be eliminated since every line of code is inspected while generating test cases. 3. Confidence to the developers 4. All syntactical and faults can be easily identified. | **(6)** |
| 1b | Consider the following program segment:  /\* Sort takes an integer array and sorts it in ascending order \*/   1. Void sort(int a[], int n){ 2. int i,j; 3. for(i=0;i<n;i++) 4. for(j=i+1;j<n;j++) 5. if(a[i]>a[j]) 6. { 7. temp=a[i]; 8. a[i]=a[j]; 9. a[j]=temp; 10. } 11. } 12. Draw the program graph for this program segment. 13. Determine the cyclomatic complexity for this program. 14. How is the cyclomatic complexity metric useful?   **Answer:**  **Program graph(3)**  **A close up of a fence  Description automatically generated**  (b) **Cyclometic Complexity: (2)**  V(G)=e-n+2P  V(G)=4  **(c)Usefulness (1)**  To identify the independent paths in a graph. | **6** |
|  | OR |  |
| 2 | Consider a program to input two numbers and print them in ascending order given below. Find all du-paths and identify those du-paths that are definition clear. Also find all du-paths, all-uses and all-definitions and generate test cases for these paths.   1. Void main() 2. { 3. int a,b,t; 4. clrscr(); 5. printf(“Enter first number:”); 6. scanf(“%d”,&a); 7. printf(“Enter Second number:”); 8. scanf(“%d”,&b); 9. if(a<b){ 10. t=a; 11. a=b; 12. b=t; 13. } 14. Printf(“%d%d”,a,b); 15. getch(); 16. }   **Answer:**  **Program graph(2)**  **Idenficiation of defining and used nodes(3)**   |  |  |  |  | | --- | --- | --- | --- | | S:no | Variable | Defining node | Usage node | | 1 | a | 6 | 9,10,14 | | 2 | b | 8 | 9,11,12,14 | | 3 | t | 10 | 10,12 |   **Du-Paths Table(3)**   |  |  |  | | --- | --- | --- | | S:No | Variable | Du-paths | | 1 | a | 6-9 | | 2 | a | 6-10 | | 3 | a | 6-14 | | 4 | B | 8-9 | | 5 | B | 8-11 | | 6 | B | 8-12 | | 7 | B | 8-14 | | 8 | t | 10-10 | | 9 | t | 10-12 |   **Du-paths that are definition clear (1)**  **Test cases(3)**   |  |  |  |  | | --- | --- | --- | --- | | **S:No** | **a** | **b** | **EO** | | **1** | **8** | **2** | **invalid** | | **2** | **8** | **2** | **invalid** | | **3** | **2** | **3** | **invalid** | | **4** | **2** | **3** | **invalid** | | **5** | **1** | **6** | **invalid** | | **6** | **8** | **9** | **invalid** | | **7** | **1** | **8** | **8** | | **12** |
|  | **Part II** |  |
| 3a | What is a typical test execution phase? Explain the steps involved in test execution phase.  **Answer:**  1.Build  2.Sutiable Environment  3.Team Size  4.TRR  5.Smoke/Sanity Testing  6.Exploratory testing  Explanation with example for the above mentioned points . | **8** |
| 3b | Write short notes on procedure call testing.  In some programming languages (FORTRAN, for example), a single procedure may have multiple entry points, and one would want to test invocation through each of  the entry points. More common are procedures with multiple exit points.  Exercising all the entry points of a procedure is not the same as exercising all the calls. For example, procedure A may call procedure C from two distinct points, and procedure B may also call procedure C. In this case, coverage of calls of C means exercising calls at all three points. If the component under test has been constructed D call coverage in a bottom-up manner, as is common, then unit testing of A and B may already have exercised calls of C. In that case, even statement coverage of A and B would ensure coverage of the calls relation | **4** |
|  | OR |  |
| 4 | Consider the program to input two numbers and print them in ascending order given in question no.2. Consider all variables and generate possible program slices. Design at least one test case from every slice.  **Answer:**  Definition of Slice S(V,n) (1)  Rules for creating slices(2)  Three variables and the number of slices and test cases for each slice(9)  S(t,16)🡪 {1-13,16}  S(6,14)🡪{1-4,7-9,11-14,16}  S(a,14)🡪{1-11,13,14,16}  S(a,6)🡪{1-6,16} | **12** |
|  | **Part III** |  |
| 5a | What is test oracle? What are its advantages and disadvantages over human oracle?  **Answer:**   * Software that applies a pass/fail criterion to a program execution is called a test oracle/ oracle * A test oracle may apply a pass/fail criterion that reflects only a part of the actual program specification, or is an approximation, and therefore passes some program executions it ought to fail * The best oracle we can obtain is an oracle that detects deviations from expectation that may or may not be actual failure * Relying on human intervention to judge test outcomes is not merely expensive, but also unreliable. * person cannot maintain the level of attention required to identify one failure in a hundred program executions, little more one or ten thousand. That is a job for a computer. | **6** |
| 5b | Explain fault-based adequacy criteria.  **Answer:**  Definition of mutation score along with example | **6** |
|  | OR |  |
| 6 | Define Scaffolding and its purpose with reference to test execution. Differentiate generic versus specific Scaffolding.  Answer:  **Scaffolding definition (2)**  **Components (6)**  **Differentiation (4)**   * Code produced to support development activities (especially testing)   + Not part of the product as seen by the end user   + May be temporary (like scaffolding in construction of buildings * Includes – Test harnesses, drivers, and stubs   Scaffoldings may include  1)Test drivers (substituting for a main or calling population)  2)Test harness (substituting for parts of the deployment environment)  3)Stubs (substituting for functionally called or used by the software under test.   * At least some level of generic scaffolding support can be used across a fairly wide class of applications * For large test suites writing each test case and a scaffold for each is impractical. * Fully generic scaffolding may suffice for small numbers of hand-written test cases . The simplest form of scaffolding is a driver program that runs a single, specific test case . | **12** |
|  | **Part IV** |  |
| 7 | Explain the following: i) Test case ii) Test case specification iii) Test suite iv) Adequacy criteria  v) Test obligation.  Answer:  Only Definitions  **Test case:**  Step by step instructions or inputs that has to be applied to the application for validation (1)  **Test case specification:**  Instructions on arriving at the test cases (2)  **Test Suite:**  Subset of all test cases (3)  **Adequacy Criteria (3)**  Mutation Score  **Test obligation (3)** | **12** |
|  | OR |  |
| 8 | Consider the greatest of three numbers code and explain the difference between statement coverage, branch coverage, path coverage and condition coverage.  **Answer:**  Source code for greatest of three numbers (4)  Statement coverage definition and no of test case identification (2)  Branch coverage definition and no of test case identification (2)  Path Coverage definition and no of test case identification (2)  Condition Coverage definition and no of test case identification (2) | **12** |
|  | **Part V** |  |
| 9 | Explain the terminologies involved in mutation testing and the purpose of mutation score with suitable example.  **Answer:**   1. Mutant: 2. Original program: 3. Alternate Program 4. Fault Seeding 5. Mutation Score   Definition of all the above terminologies (6)  Code considered as example (3)  Mutation Analysis (3) | **12** |
|  | OR |  |
| 10 | Explain Mccabe’s basis path methods with suitable example to identify the paths as well as comment on the violations in structured programming.  **Answer:**  Definition (2)  Example graph (4)  Explanation on graph (2)  Violations of structured programming  Branching in,out of the loop with diagram(4) | **12** |