

**Shri Vile Parle Kelavani Mandal's  
Narsee Monjee College of Commerce & Economics  
(Autonomous)**

**Bachelor of Science (IT) Programme 2022-2023**

**TYBSC(IT) Semester VI Division A**

**Course Name: Software Testing and Quality Assurance (STQA) Mini  
Project**

**Topic : (Practical 4) Design test cases for testing BMI program with the  
black-box strategy**

**Submitted by:**

<b>Roll No</b>	<b>Student's Name</b>	<b>SAP ID</b>
A016	Abdeali Harianawala	45207200003
A017	Tanisha Jhaveri	45207200061
A018	Ronak Joshi	45207200018
A019	Harshal Kanchan	45207200033
A020	Anjana Kizhiyapat	45207200022
A052	Avinash Joshi	45207200058

**Date of Submission: 20/01/2023**

**CONTENTS****I. INTRODUCTION TO BLACK BOX TESTING****II. CODE****III. TECHNIQUES**

<b>1</b>	<b>BOUNDARY VALUE ANALYSIS .....</b>	<b>4</b>
1.1	Testing Program Using BVA.....	4
<b>2</b>	<b>GRAPH BASED TESTING .....</b>	<b>5</b>
2.1	Steps In Graph-Based Testing .....	6
2.2	Situations Where This Graph Is Useful.....	6
2.3	Testing Program Using Graph Based Testing .....	6
<b>3</b>	<b>EQUIVALENCE PARTITIONING METHOD .....</b>	<b>8</b>
3.1	Guidelines For Equivalence Partitioning.....	8
3.2	Testing Program Using Equivalence Partitioning Method .....	9
<b>4</b>	<b>STATE TRANSITION TECHNIQUE.....</b>	<b>9</b>
4.1	When To Use State Transition? .....	9
4.2	When to Not Rely On State Transition? .....	9
4.3	Four Parts Of State Transition Diagram .....	10
4.4	State Transition Diagram and State Transition Table .....	10
4.5	Testing Program Using State Transition Technique .....	10
4.6	Advantages and disadvantages .....	12
4.7	Summary .....	12
<b>5</b>	<b>DECISION TABLE TESTING.....</b>	<b>12</b>
5.1	Testing Program Using Decision Table Testing .....	13
5.1.1	EXPLANATION.....	13
5.1.2	TEST CASES.....	14
5.2	Conclusion.....	14
<b>6</b>	<b>ERROR GUESSING .....</b>	<b>14</b>
6.1	Error Guessing In The Bmi Calculator Program.....	15
6.2	Summary .....	15

# Black Box Testing

Black Box Testing is a software testing technique where the tester does not have any knowledge of the internal structure or implementation of the system or program being tested. The tester only knows the inputs and expected outputs of the system or program.

The goal of black box testing is to ensure that the system or program behaves as expected and meets the requirements, regardless of how it is implemented. This technique is also known as functional testing, behavioural testing, or closed-box testing. It is used to test the functionality, usability, and user interface of the system or program.

By conducting black box testing, the tester can identify defects and bugs in the system or program and ensure that it is working correctly before it is released to the end-users.

**CODE:** The following is the code for calculating the BMI written in C Language:

```
1 // STQA Practical 4
2 #include <stdio.h>
3 #include <conio.h>
4
5 void main()
6 {
7     //---- ht : height, wt : weight
8     float ht=0, wt=0, bmi=0;
9
10    //---- height
11    printf("Enter height (meters): \n");
12    scanf("%f", &ht);
13
14    //---- criteria to check if the entered value (ht) is
15    invalid
16    while(ht < 0.3 || ht > 2.4){
17        printf("Enter the valid height in meters.");
18        scanf("%f", &ht);
19    }
20
21    //---- weight
22    printf("Enter weight (kg): \n");
23    scanf("%f", &wt);
24
25    //---- criteria (wt)
26    while(wt < 2.0 || wt > 350){
27        printf("Enter the valid weight in kilograms.");
28        scanf("%f", &wt);
29    }
30
31    //---- calculating the body mass index (bmi)
32    bmi = wt / (ht * ht);
33    printf("\nBMI = %f kg/sq.m", bmi);
34    getch();
35 }
```

**SOME TECHNIQUES IN BLACK BOX TESTING:**

1. Boundary Value Analysis (BVA)
2. Graph Based Testing
3. Partitioning Equivalence Testing
4. State Transition Testing
5. Decision Table Testing
6. Error Guessing

## **1 BOUNDARY VALUE ANALYSIS**

---

Boundary Value Analysis is a black-box testing technique that focuses on testing the program with inputs that are on the edge of the input domain.

This includes the minimum and maximum values that the program can handle, as well as values just above and below these limits. The idea is that these values are more likely to cause the program to fail or behave differently than values that are farther away from these limits. This technique is based on the principle that "boundary values are more likely to be incorrect" and hence by testing the system using the boundary values, it is more likely to discover errors.

Boundary Value Analysis is a black-box technique because the tester does not have any knowledge of the internal structure or implementation of the system or program being tested. The tester only knows the inputs and expected outputs of the system or program. The tester does not need to know how the program is implemented to test it using this technique, only the input domain and the expected output is needed to perform the test.

This technique is useful for testing the robustness of the system and helps in identifying the errors that might occur due to incorrect handling of edge cases. It's a powerful technique that helps to identify the errors that might occur at the boundaries of the input domain, which is not always obvious.

### **1.1 TESTING PROGRAM USING BVA**

## Boundary Values

Boundary Value Analysis (ht):

0.29	0.3	0.31
Invalid	Valid Boundary	Valid
2.39	2.40	2.41
Valid	Valid Boundary	Invalid

Boundary Value Analysis (wt):

1.99	2.0	2.01
Invalid	Valid Boundary	Valid
349.99	350	350.01

## Test Cases

Test Case No.	Height (m)	Weight (Kg)	Expected Output
1	0.3	2	BMI=22.22 kg/sq.m
2	2.4	350	BMI=145.83 kg/sq.m
3	1.86	82	BMI=23.70 kg/sq.m
4	1.7	1.99	Enter Valid Weight
5	0.29	75	Enter Valid Height
6	2.75	1	Enter Valid Height

## 2 GRAPH BASED TESTING

One of the most frequently utilized structures for presumption is the graph. Because of its clearly defined framework, testing and case studies are made simpler. It is regarded as an efficient method for choosing a system's systematic tests. The desired information that needs to be acquired or collected is represented by this model. The information that has to be provided is then presented and is simple enough for the testers to understand.

- 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 modelling – nodes represent user observable states of the software and links represent transitions between states.
- vi. Data flow modelling – nodes are data objects and links are transformations from one data object to another.
- vii. Timing modelling – nodes are program objects and links are sequential connections between these objects, link weights are required execution times.

For testing that uses graphs, the tester must first gather the data for the graph model before covering every aspect of a specific graph. In this testing procedure, the task of building a graph is given to the tester first, and then the remaining phases. These are then discussed.

## **2.1 STEPS IN GRAPH-BASED TESTING**

1. Construct the graph model.
2. Determine the test/important requirements.
3. Choose the route that will satisfy those demands.
4. Pick the information to be entered.

Software applications are, as we all know, composed of several components. The graph is created when these objects are identified.

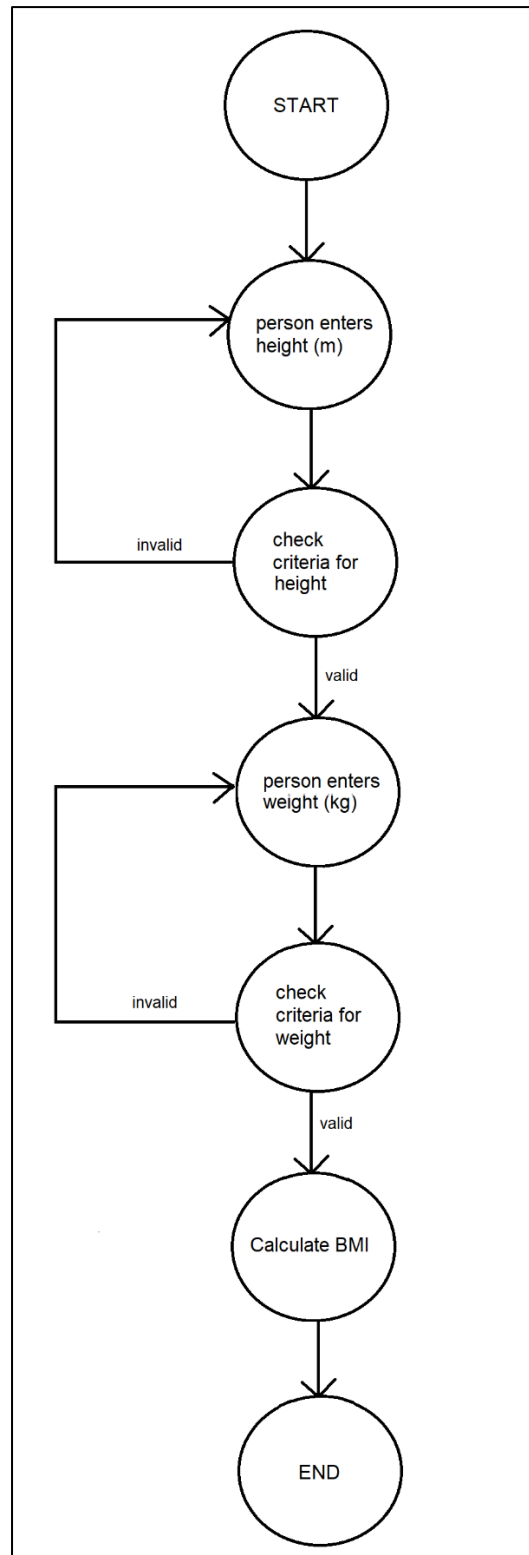
Originally used for hardware testing, it is now being utilized for software testing. Static vs. dynamic testing, the box approach, white box, black box, specification-based testing, visual testing, and Grey box testing are among the various testing techniques used in software testing. Black box testing includes graph-based testing, which includes cause and effect testing. One might select a test case related to cause and effect in this kind of testing.

While impact here refers to the conditions at the system's output or its transformation, cause here stands for the input condition that provides information about the internal change in the system.

## **2.2 SITUATIONS WHERE THIS GRAPH IS USEFUL**

- I. to identify the current issue so that a choice can be made quickly.
- II. to compare the system-affecting elements.
- III. to determine a problem's primary cause.

## **2.3 TESTING PROGRAM USING GRAPH BASED TESTING**



Everything in the development world of today is becoming visual. Nobody wants to go through mountains of tedious text before analysing or testing the software. This method has made it easy for non-technical people to comprehend the idea. As just the most significant texts are read under the graph, this technique saves time.

### 3 EQUIVALENCE PARTITIONING METHOD

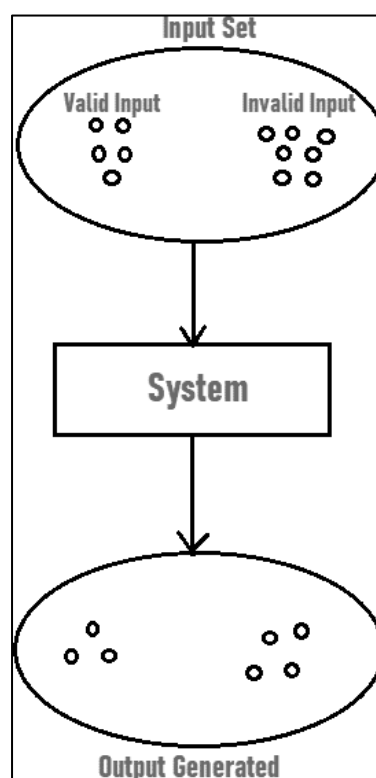
---

Equivalence class partitioning (ECP) is another name for the Equivalence Partitioning Method. It is a method of software testing known as "black-box testing" that separates the input domain into classes of data from which test cases can be derived. An ideal test case detects a sort of error that might necessitate the execution of numerous arbitrary test cases before general error is noticed.

Equivalence classes are assessed for predetermined input circumstances in equivalence partitioning. Every time an input is provided, the kind of input condition is verified, and for these input conditions, the Equivalence class reflects or specifies a collection of valid or invalid states.

#### 3.1 GUIDELINES FOR EQUIVALENCE PARTITIONING

- 1) If the range condition is given as an input, then one valid and two invalid equivalence classes are defined.
- 2) If a specific value is given as input, then one valid and two invalid equivalence classes are defined.
- 3) If a member of set is given as an input, then one valid and one invalid equivalence class is defined.
- 4) If Boolean no. is given as an input condition, then one valid and one invalid equivalence class is defined.





### 3.2 TESTING PROGRAM USING EQUIVALENCE PARTITIONING METHOD

Ht (in meters)	Wt (in kgs)	Output
T	T	T
T	F	F
F	T	F
F	F	F

Ht (in meters)	Wt (in kgs)	Output
$0.3 < ht < 2.4$	$2.0 < wt < 350$	T
$0.3 < ht < 2.4$	$2.0 > wt > 350$	F
$0.3 > ht > 2.4$	$2.0 < wt < 350$	F
$0.3 > ht > 2.4$	$2.0 > wt > 350$	F

If the height lies between 0.3 and 2.4 and if the weight lies between 2.0 and 350, then the output is calculated.

## 4 STATE TRANSITION TECHNIQUE

State Transition Testing is a black box testing technique in which changes made in input conditions cause state changes or output changes in the Application Under Test(AUT). State transition testing helps to analyze behaviour of an application for different input conditions. Testers can provide positive and negative input test values and record the system behavior.

It is the model on which the system and the tests are based. Any system where you get a different output for the same input, depending on what has happened before, is a finite state system.

State Transition Testing Technique is helpful where you need to test different system transitions.

### 4.1 WHEN TO USE STATE TRANSITION?

- This can be used when a tester is testing the application for a finite set of input values.
- When the tester is trying to test sequence of events that occur in the application under test. I.e., this will allow the tester to test the application behavior for a sequence of input values.
- When the system under test has a dependency on the events/values in the past.

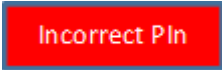
### 4.2 WHEN TO NOT RELY ON STATE TRANSITION?

- When the testing is not done for sequential input combinations.
- If the testing is to be done for different functionalities like exploratory testing

### 4.3 FOUR PARTS OF STATE TRANSITION DIAGRAM


There are 4 main components of the State Transition Model as below:

- 1) **States** that the software might get
- 2) **Transition** from one state to another (represented as arrows)
- 3) **Events** that origin a transition like closing a file or withdrawing money



Incorrect Pin

- 4) **Actions** that result from a transition (an error message or being given the cash.)



Access  
Granted

In our example the Event would be entering height and weight and the Action would be the invalid message or the final answer of BMI.

### 4.4 STATE TRANSITION DIAGRAM AND STATE TRANSITION TABLE

There are two main ways to represent or design state transition, State transition diagram, and state transition table.

In state transition diagram the states are shown in boxed texts, and the transition is represented by arrows. It is also called State Chart or Graph. It is useful in identifying valid transitions.

In state transition table all the states are listed on the left side, and the events are described on the top. Each cell in the table represents the state of the system after the event has occurred. It is also called State Table. It is useful in identifying invalid transitions.

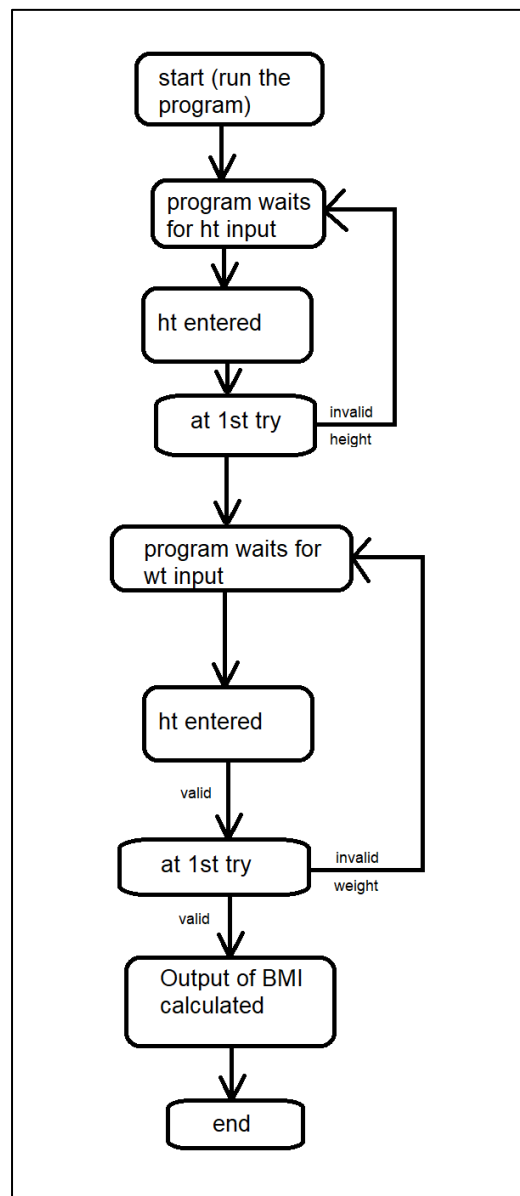
Let's see in detail and get an understanding about the test cases used in BMI Calculator program as per the State Transition Technique.

### 4.5 TESTING PROGRAM USING STATE TRANSITION TECHNIQUE

**State Transition Table :**

TESTS	TEST1		TEST2		TEST3	
	Height	Weight	Height	Weight	Height	Weight
START STATE	0	0	0	0	0	0
INPUT	0.57	85	0.1	350	2.0	366
VALID?	Yes	Yes	No	Yes	Yes	No
OUTPUT	261.61 kg/sq.m		Enter Valid Height		Enter Valid Weight	
END STATE	exit		exit		exit	

### State Transition Diagram :



## 4.6 ADVANTAGES AND DISADVANTAGES

Advantages	Disadvantages
This testing technique will provide a pictorial or tabular representation of system behavior which will make the tester to cover and understand the system behavior effectively.	The main disadvantage of this testing technique is that we can't rely in this technique every time. For example, if the system is not a finite system (not in sequential order), this technique cannot be used.
By using this testing, technique tester can verify that all the conditions are covered, and the results are captured	Another disadvantage is that you have to define all the possible states of a system. While this is all right for small systems, it soon breaks down into larger systems as there is an exponential progression in the number of states.

## 4.7 SUMMARY

- State Transition testing is defined as the testing technique in which changes in input conditions cause's state changes in the Application under Test.
- In Software Engineering, State Transition Testing Technique is helpful where you need to test different system transitions.
- Two main ways to represent or design state transition, State transition diagram, and State transition table.
- In state transition diagram the states are shown in boxed texts, and the transition is represented by arrows.
- In state transition table all the states are listed on the left side, and the events are described on the top.
- This main advantage of this testing technique is that it will provide a pictorial or tabular representation of system behavior which will make the tester to cover and understand the system behavior efficiently.
- The main disadvantage of this testing technique is that we can't rely in this technique every time.

## 5 DECISION TABLE TESTING

One of the common case design methods for black box testing is the decision table method. In this methodical technique, several input configurations and the corresponding system behaviour are recorded in a table-like structure.

Because of this, it is sometimes referred to as a cause-effect table. This method of selecting test cases is intended to speed up testing and provide thorough coverage of the software application's testing domain.

For the functions where there is a logical connection between two or more inputs, decision table techniques are appropriate. This method predicts the outcome of numerous input combinations and is related to the right combination of inputs. We must consider conditions as input and actions as output when designing the test cases using the decision table technique.

With the help of a decision table, testers can immediately spot conditions that were missed when checking all potential conditions combinations. Both True(T) and False(F) values are used to denote the conditions.

## 5.1 TESTING PROGRAM USING DECISION TABLE TESTING

### 5.1.1 EXPLANATION

Considering the example of Body Mass Index (BMI), the height and weight of a person is taken and calculated to get the BMI.

The condition is simple – The calculated answer will be displayed when the value of the height and weight fits the proper criteria.

According to the code given above, the height is considered valid only if it is greater than or equal to 0.3 or less than or equal to 2.4. Similarly, the value of weight is considered valid only if it is greater than or equal to 2.0 or less than or equal to 350.

If the values of height and weight do not match the criteria that is mentioned, then a statement would be printed to display the valid inputs.

<b>Conditions</b>	[1] Weight (wt)	T	T	F	F
	[2] Height (ht)	T	F	T	F
<b>Actions</b>	Expected Result	Display Answer	Enter the valid height in meters.	Enter the valid weight in kilograms.	Enter the valid height in meters.

### Legend

- T – True (The input matches the criteria)
- F – False (The input does not match the criteria)

### Interpretation

- Case 1 –
  - Both the variables match the criteria and displays the calculated BMI value

- Case 2 –
  - The weight variable matches the criteria, but the height variable does not. Hence the message “Enter the valid height in meters” would be displayed.
- Case 3 –
  - The height variable matches the criteria, but the weight variable does not. Hence the message “Enter the valid weight in kilograms” would be displayed.
- Case 4 –
  - Both the variables do not match the criteria. Since the height variable code was mentioned first, the message “Enter the valid height in meters” would be displayed.

### 5.1.2 TEST CASES

ht	wt	Expected output
0.30	2	BMI = 22.222221 kg/sq.m
3.34	220.1	Enter the valid height in meters.
1.5	1.99	Enter the valid weight in kilograms.
3.34	350.44	Enter the valid height in meters.

### 5.2 CONCLUSION

The program was tested with the above test cases and worked well.

## 6 ERROR GUESSING

---

Error guessing in black box testing is a technique used to identify potential errors or defects in a software system by using knowledge and experience of the system or similar systems. It is a way to explore the system without having access to its internal structure or design.

The tester uses their knowledge of the system's requirements, functionality, and common pitfalls to make educated guesses about where errors may occur. They can then design test cases to verify their guesses. This technique can be useful in uncovering errors that may not be identified through other testing methods, such as functional testing or boundary value analysis.

Error guessing is particularly useful when dealing with complex systems, as it can help to identify potential problem areas that may not be obvious. It is also a good technique for testing systems with poorly or incompletely specified requirements.

It's important to note that Error guessing is not a replacement for other testing methods, but rather a supplement to them. It should be used in conjunction with other testing techniques to ensure that the system is thoroughly tested, and all potential errors are identified.

## **6.1 ERROR GUESSING IN THE BMI CALCULATOR PROGRAM**

The use of error guessing in the case of the program that calculates the body weight index, here are a few specific examples of how it can be applied:

1. The tester can use their knowledge and experience of similar programs to check if the program properly handles decimal input values by providing weight and height with decimal values as inputs, and check if the program is able to calculate the BMI correctly.
2. The tester can use error guessing to identify potential problem areas in the program such as input validation, by providing inputs that are below the minimum valid input value or above the maximum valid input value and check if the program is able to handle these scenarios properly and produces the expected output or an error message.
3. The tester can use error guessing to identify potential new test cases that may not have been previously considered, such as testing the program with a weight that is zero or negative, to check if the program produces the expected output or an error message.
4. Tester can also use error guessing to identify if the program is able to handle corner cases such as very large or very small inputs, and check if the program produces the expected output or an error message.

Overall, error guessing can be used to identify potential errors or defects in the program that may not be identified through other testing methods, as well as to validate assumptions about the program's functionality. By using error guessing in conjunction with other testing techniques, the tester can ensure that the program is thoroughly tested, and all potential errors are identified, which will result in a more robust and reliable program.

## **6.2 SUMMARY**

In summary, Error guessing is a black box testing technique used to identify potential errors or defects in a software system by using knowledge and experience of the system or similar systems. In the case of a program that calculates the body weight index (BMI) using a person's weight and height as input, error guessing can be used to identify potential errors or defects in the program, to validate assumptions about the program's functionality, and to identify new test cases that may not have been previously considered. It can be used in conjunction with other testing techniques such as functional testing, boundary value analysis, and stress testing to ensure that the program is thoroughly tested and all potential errors are identified.