

Syllabus Content:

12.3 Program testing and maintenance:

Candidates should be able to:

Show understanding of ways of exposing and avoiding faults in programs

Notes and guidance

Locate and identify the different types of errors

- syntax errors
- logic errors
- run-time errors

Show understanding of the methods of testing available and select appropriate data for a given method Including:

- dry run, walkthrough, white-box, black-box, integration, alpha, beta, acceptance, stub
- Show understanding of the need for a test strategy and test plan and their likely contents
- Choose appropriate test data for a test plan Including normal, abnormal and extreme/boundary
- Show understanding of the need for continuing maintenance of a system and the differences between each type of maintenance including
 - o perfective, adaptive, corrective
- Analyse an existing program and make amendments to enhance functionality

Testing strategies

Finding syntax errors is easy. The compiler/ interpreter will find them for you and usually gives you a hint as to what is wrong.

Syntax Error:

A **syntax error** is a 'grammatical' error, in which a program statement does not follow the rules of the high-level language constructs.

Syntax error: an error in which a program statement does not follow the rules of the language



```
(Declarati
(General)
⊢Module Module1
                                       When Syntax error occurs, VB IDE underlines
                                     wrong syntax, and gives options for corrections.
      Sub Main()
          Dim number
                      As Intege
          Consolee.writeelinee
          Console.WriteLine("
                              Hello Dear, Enter your lucky number")
          Console.WriteLine
                             "If your Lucky numbner is correct, you will win a car ")
          number = Console ReadLine()
          While number <> -1
              Console WriteLine("Sorry Dear, You lost Enter your lucky number again ")
              number = Console.ReadLine()
              Console.WriteLine("HURRAAYYY, You've just won a HONDA CIVIC")
              Console.WriteLine("The Lucky number you've entered is " & number)
              Console.ReadKey()
      End Sub
  End Module
```

Much more difficult to find are logic errors and run-time errors.

A run -time error occurs when program execution comes to an unexpected ha It or 'crash' or it goes into an infinite loop and 'freezes'.

Logical Error:

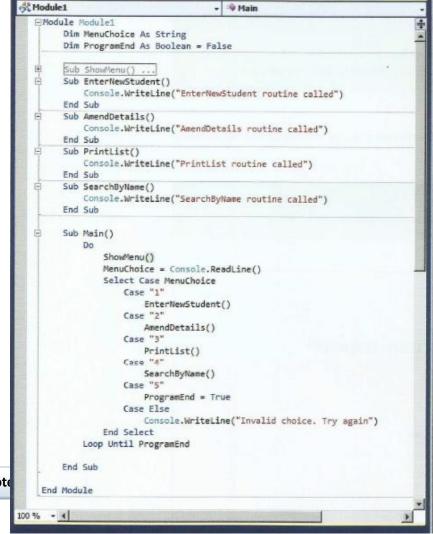
Logic error: an error in the logic of the solution that causes it not to behave as intended

Run-time Error:

Run-time error: an error that causes program execution to crash or freeze

Testing

Both of these types of errors can only be found by careful testing. The danger of such errors is that they may only manifest themselves under certain circumstances. If a program crashes every time it is executed, it is obvious there is an error.



www.majidtahir.com

For all subjects' note



If the program is used frequently and appears to work until a certain set of data causes a malfunction.

That is much more difficult to discover without perhaps serious consequences.

Stub testing

When you develop a user interface, you may wish to test it before you have implemented all the facilities.

You can write a **'stub'** for each procedure (see Figure).

The procedure body only contains an output statement to acknowledge that the call was made. Each option the user chooses in the main program will call the relevant procedure.

Black-box Testing:

As the programmer, you can see your program code and your testing will involve knowledge of the code (see the next section, about white-box testing).

As part of thorough testing, a program should also be tested by other people, who do not see the program code and don't know how the solution was coded.

Such program testers will look at the program specification to see what the program is meant to do, devise test data and work out expected results.

Test data usually consists of normal data values, boundary data values and erroneous data values.

The tester then runs the program with the test data and records their results. This method of testing is called black-box testing because the tester can't see inside the program code: the program is a 'black box'.

Where the actual results don't match the expected results, a problem exists. This needs further investigation by the programmer to find the reason for this discrepancy and correct the program.

Once black-box testing has established that there is an error, other methods have to be employed to find the lines of code that need correcting.

KEY TERMS

Test data: carefully chosen values that will test a program

Black-box testing: comparing expected results with actual results when a program is run

White-box Testing:

How can we check that code works correctly? We choose suitable test data that checks every path through the code





KEYYERMS

White-box testing: testing every path through the program code

Dry-running an algorithm:

A good way of checking that an algorithm works as intended is to **dry-run** the algorithm using a **trace table** and different test data.

The idea is to write down the current contents of all variables and conditional values at each step of the algorithm.

KEYTERMS

Dry-run: the process of checking the execution of an algorithm or program by recording variable values in a trace table

Trace table: a table with a column for each variable that records their changing values



Sec 12.3: Program testing & maintenance

CS 9618 notes & online classes with Majid Tahir at https://fiqar.org

```
WORKED EXAMPLE 15.02
Tracing an algorithm
Here is the algorithm of the number-guessing game:
SecretNumber + 34
INPUT "Guess a number: " Guess
NumberOfGuesses - 1
REPEAT
  IF Guess = SecretNumber
         OUTPUT "You took" NumberOfGuesses "quesses"
      ELSE
         IF Guess > SecretNumber
            THEN
               INPUT "Guess a smaller number: " Guess
               INPUT "Guess a larger number: " Guess
         ENDIF
         NumberOfGuesses + NumberOfGuesses + 1
   ENDIF
```

To test the algorithm, construct a trace table (Table 15.02) with one column for each variable used in the algorithm and also for the condition Guess > SecretNumber

Now carefully look at each step of the algorithm and record what happens. Note that we do not tend to write down values that don't change. Here secretNumber does not change after the initial assignment, so the column is left blank in subsequent rows.

SecretNumber	Guess	NumberOfGuesses	Guess > SecretNumber	Message
34	5	1	FALSE	larger
	55	2	TRUE	smaller
	30	3	FALSE	larger
	42	4	TRUE	smaller
	36	5	TRUE	smaller
	33	6	FALSE	larger
	34	7		7 guesses

Table 15.02 Trace table for number-guessing game

UNTIL Guess = SecretNumber

References:

- © Cambridge International AS & A level Computer Science Course book by Sylvia Langfield and Dave Duddell
- Visual Basics Console Mode Editor Window from notes of Sir Majid Tahir