



Temasek Junior College
JC H2 Computing
Problem Solving & Algorithm Design 12
Program Testing

Example 1 An array should contain eight positive integers. An algorithm is designed to locate the first incidence of a particular integer in the array.

Test data is required to test the algorithm as fully as possible.

The table below shows two suitable examples of test conditions and test data:

Condition being tested	Array data	Search integer
Search integer not present	5 8 5 9 2 3 9 4	7
Array contains an incorrect amount of integers	2 3 5 6 4 8 1	9

Copy the above table headings and write down two more conditions which should be tested, providing suitable test data in each case. [4]

Solution: Any 2 of this type of example:

(Marking: 1 for condition and 1 for both parts of test data x 2)

Condition being tested	Array data	Search integer
Search integer occurs once	5 8 5 7 2 3 6 4	7
Search integer occurs more than once	5 8 5 7 2 3 7 4	7
Array data contains all same integer (= or \neq Search)	2 2 2 2 2 2 2 2	2
Array empty		2
Search integer does not exist	5 8 5 9 2 3 9 4	7

Other possibilities can be credited

Example 2 Below is an algorithm [WJEC-S12-1101-01 S. 10 + 4 (2013)]

```

Algorithm
X is integer
Y is integer
Z is integer
I is integer
startmainprog
    input X
    input Y
    set Z = 1
    for I = 1 to Y
        set Z = Z * X
    next I
    output Z
endmainprog

```

- (a) Complete the table below to show how each variable changes when the algorithm is performed on the test data given [4]

Test data: $X = 2$ and $Y = 4$

i	X	Y	Z

- (b) Briefly describe the purpose of this algorithm [1]

Solution:

(a)

I	X	Y	Z
	2	4	1
1	2	4	2
2	2	4	4
3	2	4	8
4	2	4	16

NOTE – deduct one mark if any additional rows are completed

- (b) The purpose of this algorithm is to raise X to the power Y (calculate the power of a number) or multiplies X by itself Y times

1 Program Testing

Understand the types of program errors that are possible and which types can be detected by test cases.

There are three main types of program errors in 'Syllabus 9596':

1. **Syntax errors** ***
 2. Semantic errors
 3. **Run-time errors** ***
 4. **Logic errors** ***
 5. Linking [Compiler based language]
 6. Arithmetic errors
 - a. Rounding
 - b. Truncation
- Following are the types of error, and for each describe:
 - a **situation** where the error might occur.
 - **how** the error is detected.
 - **strategies** that would reduce the chance of this error.

2 Errors

2.1 Syntax*** An error that occurs when a command does not follow the expected syntax of the language

Example 3 IF without THEN

```
>>> prin("Hello")
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    prin("Hello")
NameError: name 'prin' is not defined
>>>
>>> a = 10
>>> if a == 10
SyntaxError: invalid syntax
>>>
>>> print ('TJC')
SyntaxError: EOL while scanning string literal
```

Situations

missing punctuation key or reserved word spelt incorrectly

NOTE: example can come from a specific language but it must be clear that it is a syntax error by showing what the correct word should be for example prin should be print, msgbos should be msgbox

If just 'spelling mistake' then must have example like above

How detected: Compiler or interpreter will often detect and highlight

Strategies: programmers can employ to reduce the chance of this error:

- ✓ Consistent naming conventions
- ✓ Use meaningful variable names
- ✓ Consistent use of letter case (CamelCase, lowercase, etc...)
- ✓ Thorough design before coding

2.2 Semantic Program, it will run successfully in the sense that the computer will not generate any error messages. However, your program will not do the right thing.

Example 4 Forgetting to divide by 100 when printing a percentage amount.

```
>>> price = 50
>>> Disc_PerCent = 10
>>> print('Discounted price = ', 50 * (100-Disc_PerCent))
Discounted price = 4500
>>> print('Discounted price = ', 50 * (100-Dics_PerCent)/100)
Discounted price = 45.0
```

Situations This will produce the wrong answer because the programmer implemented the solution incorrectly

How detected Programmer requires to work backward by looking at the output of the program and trying to figure out what it is doing.

Strategies programmer must fully understand the problem so the he can tell if his program properly solves it.

2.3 Runtime/execution***

An error that only occurs when the program is running and is difficult to foresee before a program is compiled and run

Example 5 division by zero
reading past the end of file
stack overflow / request more memory than available
overflow of data type (for example, integer too big)
trying to access out of range array

```
>>> age = 100
>>> age/0
Traceback (most recent call last):
  File "<pyshell#6>", line 1, in <module>
    age/0
ZeroDivisionError: division by zero

>>> 500.0**1000
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    500.0**1000
OverflowError: (34, 'Result too large')

>>> array = [0] * 10
>>> array[10]
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    array[10]
IndexError: list index out of range
```

How detected Program crashes

Strategies programmers can employ to reduce the chance of this error:

- ✓ Dry run algorithms with realistic test data before coding (not twice)
- ✓ Thorough design of algorithm before coding

2.4 Logical*** An error that causes a program to output an incorrect answer. A logic error occurs when the programmer makes a mistake in their logic for some part of the program.

Example 6 `count = count – 1` should be `count = count + 1`
 branch to the wrong statement
 call the wrong sub-routine
 looping too many times

How detected Testing with test data that has predicted output and comparing with actual output (Not twice)

Strategies programmers can employ to reduce the chance of this error:

- ✓ Dry run algorithms with realistic test data before coding (not twice)
- ✓ Thorough testing of code with a test plan

2.5 Linking An error that occurs when a programmer calls a function within a program and the correct library has not been linked to that program [when using compiler]

Example 7 When the square root function is used but the library that calculates the square root has not been linked to the program

How detected Compiler or interpreter will often detect and highlight
 Program will crash when function call is attempted to execute

Strategies programmers can employ to reduce the chance of this error:

- ✓ Test individual chunks of code that contain any reference to external libraries
- ✓ Thorough testing of code with a test plan

2.6 Rounding Rounding is when a number is approximated to nearest whole number/tenth/hundredth, etc.

Example 8 79.4 rounded to nearest whole number is 79, an error of +0.4

How detected Testing with test data that has accurate predicted output and comparing with actual output (Not twice)

Strategies programmers can employ to reduce the chance of this error:

- ✓ Use integer data type wherever possible
- ✓ Using data types that will store larger/more accurate numbers than will ever be used in the program

2.7 Truncation Truncating is when a number is approximated to a whole number/tenth/hundredth, etc. nearer zero

Example 9 22.8 truncated to whole number is 22, an error of 0.8

How detected Testing with test data that has accurate predicted output and comparing with actual output (Not twice)

Strategies programmers can employ to reduce the chance of this error:

- ✓ Use integer data type wherever possible
- ✓ Using data types that will store larger/more accurate numbers than will ever be used in the program

3 Debugging features of the programming environment: Breakpoint and Stepping

- **Breakpoint:** It is a signal that tells the debugger to temporarily suspend execution of your program at a certain point. A point where the program can be halted to see if the program works at this point.
- **Stepping:** When your program is in break mode – that is, the program has paused at a breakpoint – you can control how execution continues or Executes one statement at a time and then pauses to see the effect of each statement.

Example 10 Debug in Python

```

Python 3.6.4 (v3.6.4:d48ecec, Dec 19 2019, 06:54:40) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license()" for more information.
>>>
[DEBUG ON]
>>>
RESTART: D:\Computing\0. 9569 Computi
from 2019\Python codes\debug_test.py
Twinkle, twinkle, little star,
How I wonder what you are.
something
[DEBUG ON]
>>>

```

Go Step Over Out Quit ☒ Stack ☒ Source ☒ Locals ☒ Globals

```

bdb'.run(), line 431: exec(cmd, globals, locals)
'_main_'.<module>(), line 3: print ("something")
> 'idlelib.run'.write(), line 362: return self.shell.write(s, self.tags)

```

Locals	
s	'\n'
self	<idlelib.run.PseudoOutputFi...bject at 0x000002275C50C860>

Globals	
Executive	<class 'idlelib.run.Executive'>
LOCALHOST	'127.0.0.1'
MyHandler	<class 'idlelib.run.MyHandler'>
MyRPCServer	<class 'idlelib.run.MyRPCServer'>

```

import io
import linecache
import queue
import sys
import time
import traceback
import _thread as thread
import threading
import warnings

import tkinter # Tcl, deletions, messag

from idlelib import autocomplete # Auto
from idlelib import calltips # CallTips
from idlelib import debugger_r # start_
from idlelib import debugobj_r # remote
from idlelib import iomenu # encoding
from idlelib import rpc # multiple obje
from idlelib import stackviewer # Stack
import __main__

for mod in ('simplifiedialog', 'messagebox'
            'dialog', 'filedialog', 'com
            'ttk'):
    delattr(tkinter, mod)
    del sys.modules['tkinter.' + mod]

```

```

import pdb; pdb.set_trace()
help
debugger-breakpoint

```

Example 11 Computer programs sometimes contain errors.

Name three different types of error that could occur in a computer program. Give an example of each type of error.

Solution:

[6]

Error	Suitable Example
Syntax	Incorrect: IF A ADN B Then Correct: IF A AND B Then
Semantic	Attempting to assign incorrect data type integer = real
Runtime or Execution	Division by zero $A \leftarrow 5$ $B \leftarrow 0$ PRINT ('A / B = ', A / B) Reading past end of file or out of memory
Logical	Count \leftarrow Count – 1 should be Count \leftarrow Count + 1
Linking	When the Square Root function is used and the library that calculates the Square Root has not been linked to the program.
Rounding	34.5 rounded to nearest whole number is 35, an error of +0.5.
Truncation	34.9 truncated to whole number is 34, an error of -0.9.

1 mark for naming each error

1 mark for suitable example (there are many suitable examples)

