Debugging and Testing

Introduction to Python Debugger

- In programs, errors are common, and one cannot avoid them.
- It is almost impossible to write a program with zero errors.
- The errors are related to the syntax of the programming language and sometimes related to the logic.
- The removal of software errors is called debugging.
- Removing errors is one of the toughest challenges posed by programmers and one of the toughest skills that need to be acquired.
- Python provides native non-GUI debuggers.

Methods of removing bugs

1. Code tracing
2. Ducky Debugging
3. Print statements

Python Debugger(pdb)

Python provides a native debugger called Python debugger (Pdb).

It is one of the default debuggers that come bundled with Python.

Pdb through Python command line interpreter

- Pdb can use the command-line interface (CLI) so that a Python program can be run from the terminal. The way to invoke pdb through CLI is to use the command.

python-m pdb test_debug.py

Running through Pdb

Python program of Listing1 is imported for the pdb environment. First, it is necessary to know the lines of the Python code.

So, list commands are useful in a pdb environment to list the lines.

Differences between step and Next

Step	Next
When a function call is encountered, it moves to that function's first line and tells what is happening.	Next, executes all the lines of the function in a single go. Then Pause at the next function call.
The step helps to step through the lines (steps into a function)	If a function call is encountered, the function cell is executed along with the function and continues to the next lines of the current function.

Set up of Break Points

Break Points	Description of the Breakpoint
breakpoint() function.	The following code illustrates the example of using a breakpoint as `breakpoint ()` The major advantage is that many Python debuggers recognize this and, therefore, it is preferable to use this.
Breakpoint by function name.	`(Pdb) break sample.model` Here, the model is the function present in the sample.py program.
Breakpoints by condition.	Here, the sample is the program name, and the model is the function that needs to be debugged. `(Pdb) break sample.py, line, condition variable.`
Breakpoints by line number.	`(Pdb) break 5` Set a breakpoint in line 5 of the program.

Manipulation of Break Points

Break Points	Description of the Breakpoint
Break (Pdb) break 8	**List all breaking points** If one wants to know all the breakpoints of a program, then this command lists all the breakpoints of the program. The above command removes the breakpoint and is declared in line 8.
clear	(Pdb) clear One can remove all declared breakpoints using a clear command. One can remove a specific breakpoint, for example, 8, then it can be done. (Pdb) clear 8
enable 3 disable 2	Enabling and disabling the breakpoints. One can disable the breakpoint as (Pdb) disable 8 To enable the breakpoint, one can use the command below. (Pdb) enable 8

Pdb through primitive GUI and IDLE

- Pdb through CLI is difficult. Modern Python IDE provides a lot of support by offering GUI for debugging purposes. IDLE support a primitive GUI facility for debugging.
- For doing that, one has to set the debug options ON in IDLE.
- IDLE responds like this!

Debugging GUI Buttons

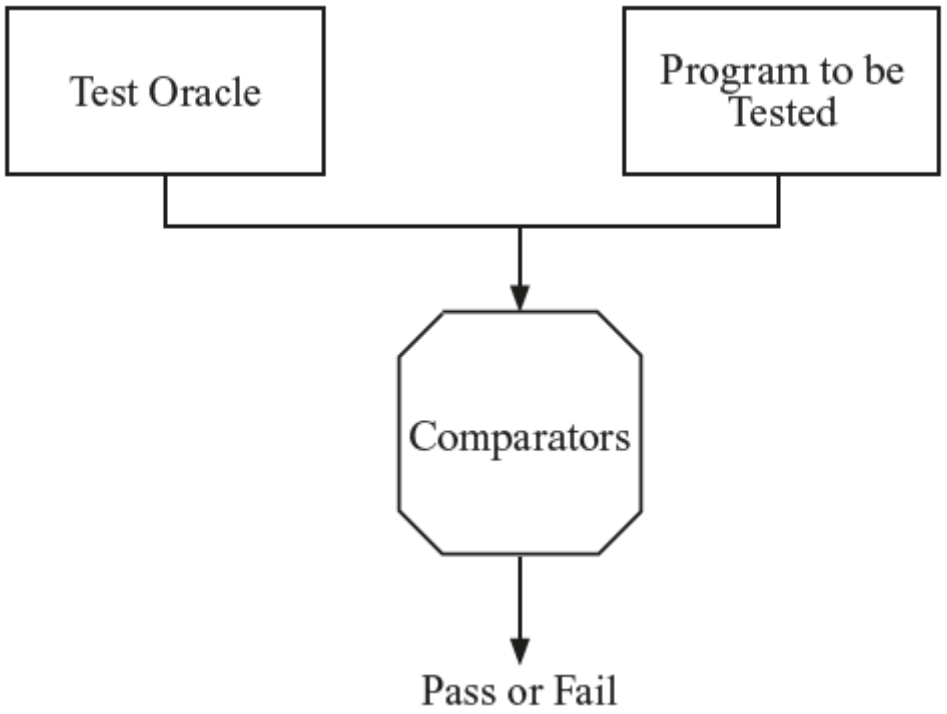
Buttons	Descriptions
Go	Go along with breakpoints. Executes till break point. Right click on the line makes it yellow.
Step	Helps to walk through.
Over	A combination of step + go, it steps over functions.
Out	Helps to execute the rest of the code in the function, if one is inside the function.
Quit	Turns off the debugger.

Testing in Python

- Software testing is executing a program to check, if any errors are present. Testing is a necessary activity to check whether the program behaves in an expected manner. All the individual components of a program are called a unit.
- A unit can be an object, function, or module. Every unit can be tested independently.
- One can automate the execution of tests. This is called test automation, and there are various tools available.
- There are various advantages to automated testing.

1. Manual testing is difficult.
2. Automated tests are effective.
3. Manual testing is a boring job. Automated testing reduces the effort.
4. Automated tests can detect errors easily.
5. Many tools are available for unit testing.

Testing Process



- A testing bed or test oracle is a mechanism that provides the correct result. Test cases reveal errors.
- An ideal test case uncovers the errors that are present in the program. The characteristics of an ideal
- test case are.

1) Reliability

2) Validity

- If many test cases satisfy criteria that detect the same error is called reliability. A criterion is valid if criteria, when satisfied by an error, will be revealed. Some strategies that are used are given below.

1) Black-box testing

2) White-box testing

Black Box testing

Test Case Design in Black-Box Testing

One way to design a test case is to go for partitioning. Suppose, if a program tests for input, for example, from 0 <= x <= max, then according to the partitioning equivalence testing, the test cases would be designed as

$$\text{Test case} = \begin{cases} < 0 \text{ for } x < 0 \\ x \text{ in range 0-max} \\ x > \text{max} \end{cases}$$

Often, a program fails at the boundary conditions. With boundary value analysis, the boundary will be considered for a test case analysis. Suppose, if the range of x is

$$0 <= x <= 5$$

Then, the test case would be to check the value of x as 0, 5, -1.0, and 5.1. {-1 for $x < 0$, 0 and 5 for boundaries and 5.1 for $x > 5$}

White Box testing

- In the white-box testing, the implementation and non-functionality are tested.
- It is also known as structural, and control-flow testing.

Test Case Design in White-Box Testing

In a sample case, the control flow can be tested as shown below in Fig. 17.4.

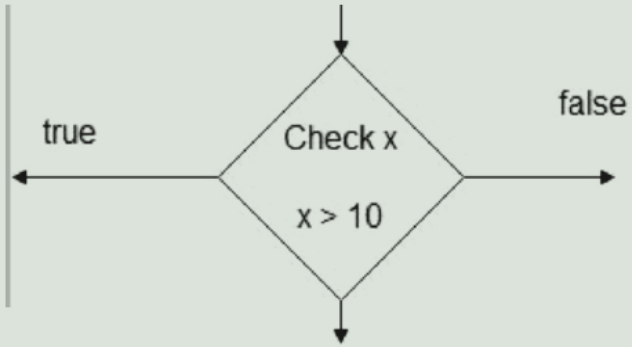


Fig. 17.4. Structural Testing

For example, a test case can be designed in such a way that both parts of the condition are tested through a proper test case. A test case would be $x > 6$ and $x < 6$. It would be interesting to check how the program functions when $x = 6$.

Doctest

- Two testing libraries are provided by Python. They are listed below.

1) Doctest

2) Unit test

Doctest or doctest is one of the simplest ways of testing a Python program.

Doctest is useful for

1) Check if the docstring is updated.

2) To write documentation.

3) To test modules.

Unit test

- Unit test is another testing library. Doctest is primitive. When a Python program is lengthy and has various functions, one can use the unittest. Pytest is originally called a pyunit. It is modelled on Java’s Junit to provide comprehensive testing.
- It can be observed that a unit test is a series of test cases where the actual and expected values are checked whether there is a match. If there is a match, the module is considered a 'pass'. Otherwise, the test is considered a 'fail'.

Pytest

An advanced unit test framework is called the Pytest. Pytest offers more functionality than unittest and is also actively supported by the Python user community.

Pytest Debugging

The Python environment does not provide the native pytest.

It should be installed. pytest can be installed as below in Command line using pip.

pip install pytest

pytest expects python programs to have test either as prefix or suffix. Then pytest can be run as follows:

```
python –m pytest test_sample.py
or
pytest test_sample.py
```

This can be used in a verbose mode:

```
python –m pytest –v test_sample.py
or
py.test test_sample.py –v
```

The difference between pytest and doctests is given in Table 17.8.

Table 17.8: Differences between Doctest and Pytest

Doctest	Pytest
Together	Separate code and test Develop an extensive suite

Table 17.9: Differences between Pytest and Unittest

Pytest	Unittest
Supports all function-oriented and object-oriented tests. Only one assert statement.	Conceptually based on object-oriented programming, an object inheritance. Various types of assert statements.

Introduction to Profiling

- The requirement of goof programming is to create efficient programs.
- The characteristic of efficiency is time.
- A program is supported to run faster.
- There are two ways of finding complexity.

1. One way is to use asymptotic analysis.
2. Another way is to use empirical analysis.

Python timeit()

- At various times, one needs to measure the execution time. Python Module timeit is used to measure the execution time.
- The syntax of timeit is given below.

- Instead of timing the statements, functions, and modules, Python has two profiling tools for profiling the entire code. The popular profilers are.
- 1) profile
- 2) cProfile
- Both tools are used to analyse CPU cycles that are required to run the Python script. Both profiling modules hold a common interface.