Debugging Programs

Based on CBSE Curriculum

Class -11

Chapter-6

Introduction

- When we develop a program for some task, generation of errors is so obvious.
- A Programmer needs to find these errors and try to fix them until the program become error free. This process is called debugging.
- If you are familiar with an error or exception, it will be easier for you to debug the program.
- In this chapter, we will discuss python errors, exception, debugging techniques and python debugger pdb.

What is Debugging?

- Debugging means the process of finding errors, finding reasons of their occurrence and techniques of their fixation.

ERRORS and EXCEPTION

- Errors and Exception, both interrupts the Program Execution.
- Errors and Exception are not same.
  - Errors, usually finds out at the time of translation.
  - Whereas, Exception generates during program run, due to an input.

Error

- An error, also known as a bug, is a programming code that prevents a program from its successful interpretation.
- Some errors are less harmful in nature whereas some are very harmful. Some errors are very difficult to find.
- Errors are of three types –
  - Compile Time Error
  - Run Time Error
  - Logical Error

Compile Time Error

- These errors are basically of 2 types –
  - Syntax Error : Violation of formal rules of a programming language results in syntax error.

    For ex-

    x=(x*y)

  - Semantics Error: Semantics refers to the set of rules which sets the meaning of statements. A meaningless statement results in semantics error.

    For ex-

    x * y = z

Logical Error

- If a program is not showing any compile time error or run time error but not producing desired output, it may be possible that program is having a logical error. This error generates because of wrong analysis by programmer.

For ex-

- To use a variable without an initial value.
- To provide wrong parameters to a function.

These errors are most difficult to find out and handle.

Run Time Error

- These errors generates during a program execution and known as run time errors.
- These errors are difficult to find out.
- Some run time errors prevents a program from execution which is known as program crash or abnormal termination of program.
- Most run time errors like infinite loop or wrong value are easy to detect.
- Python is having provision of checkpoints to handle these errors.
- It is important to prevent a program from crashing in case of run time error. A program should be robust.

Exception

- Exception is a state when a program terminates abnormally and it can't be controlled.

For ex-

- a=10
- b=0
- c=a/b (Mathematical Exception)

Another Ex-

- Inputting wrong account number in an ATM is an error.
- But receiving less amount than expected is an Exception.

An exception cause termination of a program.

Exception Handling in Python

- In Python, try and except clauses are used to handle an exception. The block which has the probability of occurring an exception, that block is to be written under try clause and the code to handle the exception is to be written under except clause. Syntax is as under-

```
try:
    # code with probability of exception will be written here.
except:
    #code to handle exception will be written here.
```

```
try:
    print("result of 10/5 = ",(10/5))
    print("result of 10/0 = ",(10/0))
except:
    print("Devide by zero error! Denominator must not be zero!")
```

OUTPUT

```
result of 10/5 =  2.0
Devide by zero error! Denominator must not be zero!
>>>
```

Available Exceptions in Python

Exception Name	Description
IOError	This exception generates due to problem in input or output.
NameError	This exception generates due to unavailability of an identifier.
IndexError	This exception generates when subscript of a sequence is out of range.
ImportError	This exception generates due to failing of import statement.
TypeError	This exception generates due to wrong type used with an operator or a function.
ValueError	This exception generates due to wrong argument passed to a function.
ZeroDivisionError	This exception generates when divisor comes to zero.
OverFlowError	This exception generates when result of a mathematical calculation exceeds the limit.
KeyError	This exception generates due to non-availability of key in mapping of dictionary.
FOFError	This exception generates when end-of-file condition comes without reading input of a built in function.

How to debug a program?

- An error can also be debugged by correcting the code.
- Compile time error can be debugged before program run.
- Logical error can be solved by Testing.
- Testing is to be used with some sample values to check the correct behavior of the program.
- There are following techniques of debugging-
  - To identify the place of Error generation.
  - By printing step wise value of Variable.
  - By tracing the Program Code.

Detection of origin of an Error

- Look at following code carefully-

```
a=int(input("Enter a number"))
b="a"
for i in range(4):
    print(b**i)
```

- Following error generates during its translation-

```
TypeError: unsupported operand type(s) for ** or pow(): 'str' and 'int'
```

- We need to change the statement to remove this error

```
a=int(input("Enter a number"))
b=a
for i in range(4):
    print(b**i)
```

OUTPUT

```
Enter a number12
1
12
144
1728
```

Printing of stepwise values of Variable

- See following code carefully -

```
a=0
b=1
print(a)
print(b)
for i in range(5):
    c=a+b
    print((c), end=" ")
    b=c
    a=b
```

- Following wrong output generated on its run.

```
0
1
1 2 4 8 16
```

- We need to print value of each variable at every step to detect the error.

```
a=0
b=1
print(a)
print(b)
for i in range(5):
    c=a+b
    print((c), end=" ")
    b=c
    a=b
    print("a = ",a,end = " ")
    print("b = ",b)
```

OUTPUT

```
0
1
1 a =  1 b =  1
2 a =  2 b =  2
4 a =  4 b =  4
8 a =  8 b =  8
16 a =  16 b =  16
```

We can find out the problem by watching these values and then can change the code accordingly.

Printing of stepwise values of Variable

- Now we can change the code to get the corrected result.

a = b
b = c

change

OUTPUT

corrected code

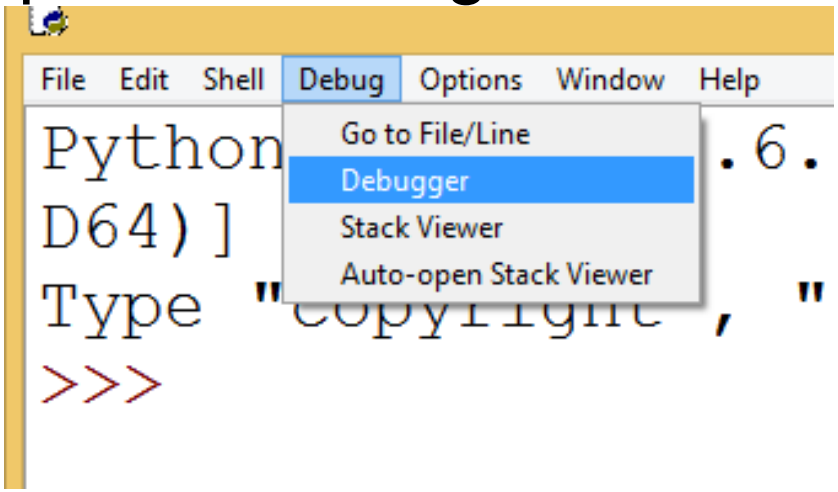
```
a=0
b=1
print(a)
print(b)
for i in range(5):
    c=a+b
    print((c), end=" ")
    a=b
    b=c
```

```
0
1
1 2 3 5 8
```

Therefore, we can say that debugging means to find the error and to correct it until program run successfully.

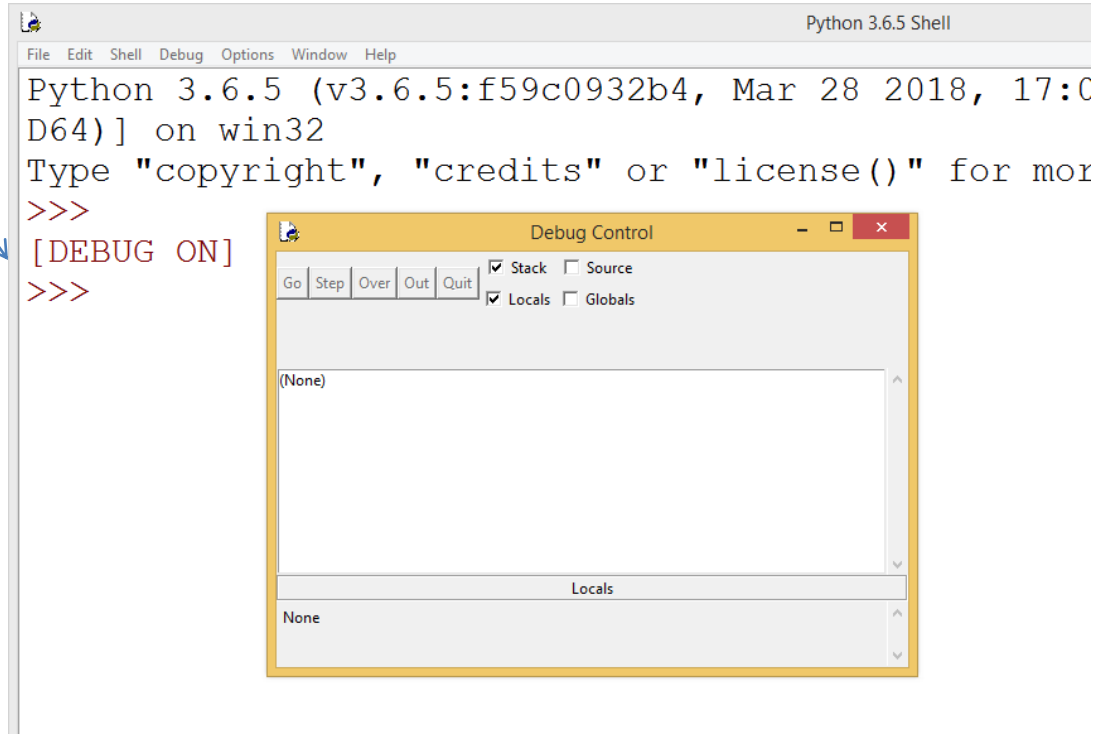
Tracing of code

- Another technique for removing an error is code tracing. In this technique, lines are to be executed one by one and their effect on variables is to be observed. Debugging tool or **debugger tool** is provided in Python for this.
- In Python3.6.5, to make debugger tool available, click on debugger option in debug menu.



Tracing of code

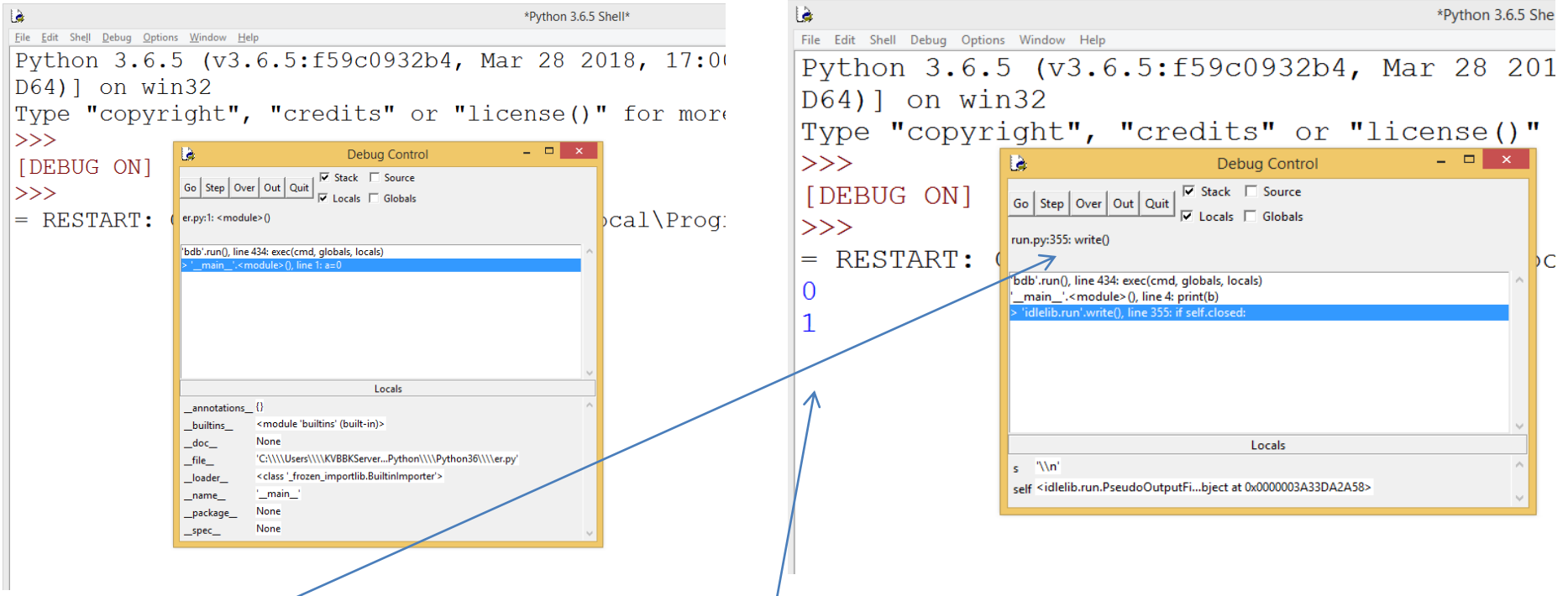
- Then, a box will be opened and a message will come saying DEBUG ON



- Then, we will open our program from file menu and will run it.

Tracing of code

- Then, program environment will be shown like this in debugger.



- Now, click on STEP button. All statements will be executed one by one and result will be display in output. When we will get wrong value, we can stop the program there and can correct the code. This process will be continued until program results in correct desired output.