Handling Exceptions

```
try:

  fun(3)
# note that braces () are necessary here for
# multiple exceptions

except ConnectionError :
   raise RuntimeError('Failed to open database')

except ZeroDivisionError:
  print("ZeroDivisionError Occurred and Handled")

except NameError:
  print("NameError Occurred and Handled")

except OSError as err:
   print("OS error:", err)

finally:
  # this block is always executed
  # regardless of exception generation.
  print('This is always executed')
```

Warning control

```
import warnings
 # adding a single entry into warnings filter
warnings.simplefilter('error', UserWarning)
 # displaying the warning
warnings.warn('This is a warning message')

**********************************************

Traceback (most recent call last):
  File "main.py", line 8, in
     warnings.warn('This is a warning message')
UserWarning: This is a warning message
```

```
In the above program, a single entry is added to the warnings filter using
warnings.simplefilter('error', UserWarning) in which the action is "error"
and category is UserWrning and then the warning is displayed using the
warn() method.
```

use C code in Python

you can write C code that can be imported into Python as a module. Python calls these **extension modules**. You can invoke it from Python directly, an example from the

Python Code

```
import example

result = example.do_something()
```

C Code

```


static PyObject * example(PyObject *self)
{
   // do something
   return Py_BuildValue("i", result);
}
```

```
import sys
class ostream:
   def __lshift__(self, a):
       if a == endl:
           sys.stdout.write("\n")
           sys.stdout.flush()
       else:
           sys.stdout.write(str(a))
       return self


cout, endl = ostream(), object()


cout<<5<<endl
```