Debugging

- Debugging and PDB
- pdb Commands

Debugging

- Python comes with the debugger **pdb.**
- Invoking the debugger:

  python –m pdb <name of script>

- Invoking the above command starts the python debugger on the specified script

trainer.cpp@gmail.com

pdb Commands

- s/step: step to next executable command
- n/next: jump to next statement in current function
- r/return: return from current function
- l: List code
- ll: long list the code
- p: evaluate the expression after p and print result

trainer.cpp@gmail.com

pdb Commands

- pp: pretty print the result
- c/continue: continue execution till a breakpoint
- b: breakpoint [list or insert a breakpoint]
- cl: clear a breakpoint

trainer.cpp@gmail.com

Logging Module

- Python Logging Levels
- Creating a simple Stream Logger
- Adding Formatter
- Logger Hierarchy
- Logger File handers

Python Logging Levels

LEVEL	WHEN TO USE
DEBUG	Detailed information, typically of interest only when diagnosing problems.
INFO	Confirmation that things are working as expected.
WARNING	An indication that something unexpected happened, or indicative of some problem in the near future (e.g. 'disk space low'). The software is still working as expected.
ERROR	Due to a more serious problem, the software has not been able to perform some function.
CRITICAL	A serious error, indicating that the program itself may be unable to continue running.

trainer.cpp@gmail.com

Simple Logger with stream handler

- Handlers allow sending the log messages to multiple different destination.
- Ex: Console (StreamHandler), Files (FileHandler), Sockets (SocketHandler)
- h = logging.StreamHandler()
  <logger_object>.addHandler(h)

Formatter

- Use **logging.Formatter** class
- Ex:

  fmt = logging.Formatter('%(asctime)s %(levelname)s %(message)s')
- Asctime, levelname, message, lineno, levelname, levelno, processname, process …..

Getting Logger Instance

- logging.getLogger()
- Returns root logger instance without any arguments
- Returns a new logger object with the corresponding name.

- Use the following when using in modules

  Logging.getLogger(__name__)

trainer.cpp@gmail.com

Handlers

- <logger object>.addHandler(<handler object>)

- StreamHandler
- FileHandler(filename, mode = ‘a’)
- RotatingFileHandler(filename, mode=‘a’, maxBytes=0, backupCount=0)

trainer.cpp@gmail.com