Catching exceptions: try code and catch/except errors

Try a bit of code, if error happens, catch used to do alternate code (like logging error)

Can be used to log important information(like location, cause, type)

Useful for large code or when need to run code regardless of an error

Divide and conquer:

If an error has unknown origin, snippets of code can be run independently, if code returns error, that is where the problem(or different problem) has occurred

Commenting out the rest of the code is viable, but new default test values need to be given if code requires

Most useful with spaghetti code

Backtracking:

errors in code can be traced back through previous run code(e.g. where method called from) to find any problems with code that could have caused problem.

Useful for shorter/simpler code where call locations are known/easily found

Deduction:

Ignore any code that is not likely to cause problems in an error(e.g code not calling or used by function)

Try to locate the error based on likeliness of where it has occurred

Testing or deducing that the location is not errored should go to next likely location

Useful if no information can be gained about an error