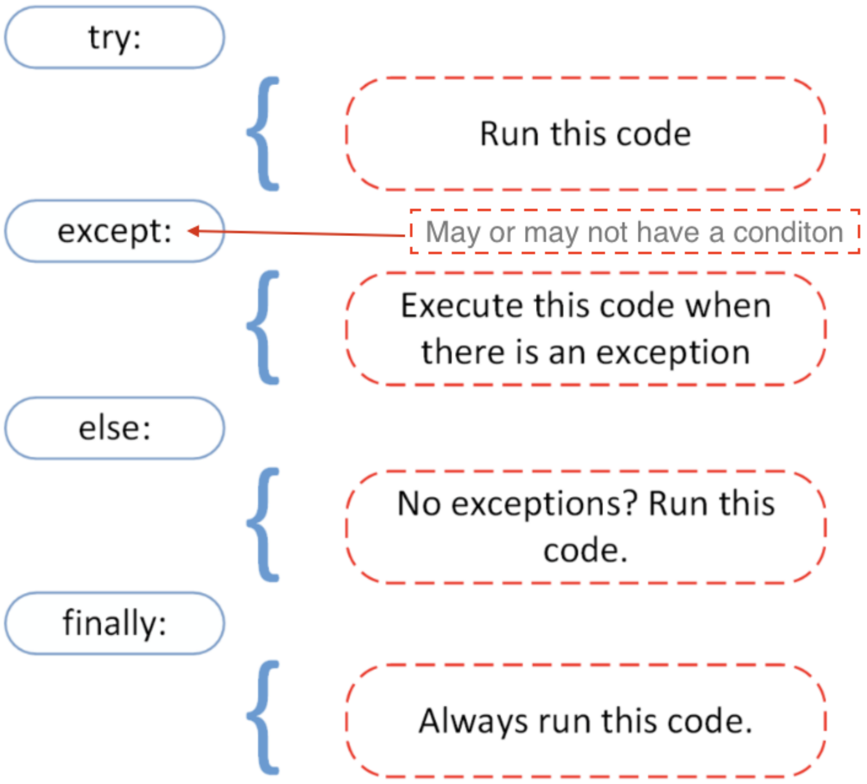
Day 2 :- 04/02/2025

- **OOPS:-**
  In object Oriented Programming we group all behaviour/properties in object
  The four pillars of oops are :-
  - Encapsulation: this allows us to bundle data and functions with in a class. This also helps to maintain data integrity(The process of ensuring that data is accurate throughout its life cycle) and secure code.
  - Inheritance:- In inheritance one child class aquire/inherit some properties of parent class.
  - Abstraction:- It hides implementation and expos only essential functionality to user(Example of atm machine)
  - Polymorphism:
- **Errors and exceptions:-**
  There are two types of errors:
  - Syntax: This error occur when there is mistake in code such as syntax of for loop is wrong etc.
  - Exception: Error detected during execution . This occur even if there is no syntax error (Example: when we try to concatenate string and int we get exception error).
- **Exception handling:-.**
  Exception handling is a programming technique that allows a program to continue running when it encounters unexpected conditions.


try:
Run this code
except:
May or may not have a conditon
Execute this code when there is an exception
else:
No exceptions? Run this code.
finally:
Always run this code.

The try statement specifies **exception handlers(A Code that say what to do when a program will do an anomalous or any exception event occurs which disturbs normal flow of program)**.
Try statement have more than one except clause.

The try statement works as follows.

- First, the try clause (the statement(s) between the try and except keywords) is executed.
- If no exception occurs, the except clause is skipped and execution of the try statement is finished.
- If an exception occurs during execution of the try clause, the rest of the clause is skipped. Then, if its type matches the exception named after the except keyword, the except clause is executed, and then execution continues after the try/except block.
- If an exception occurs which does not match the exception named in the except clause, it is passed on to outer try statements; if no handler is found, it is an unhandled exception and execution stops with an error message

Syntax:

```
try:
x = int(input("Please enter a number: "))
     break
  except ValueError:
     print("Oops!  That was not valid number.  Try again...")
```

else clause:-
The optional else clause is executed **if the control flow leaves the try suite, no exception was raised, and no return, continue, or break statement was executed**. Exceptions in the else clause are not handled by the preceding except clauses.

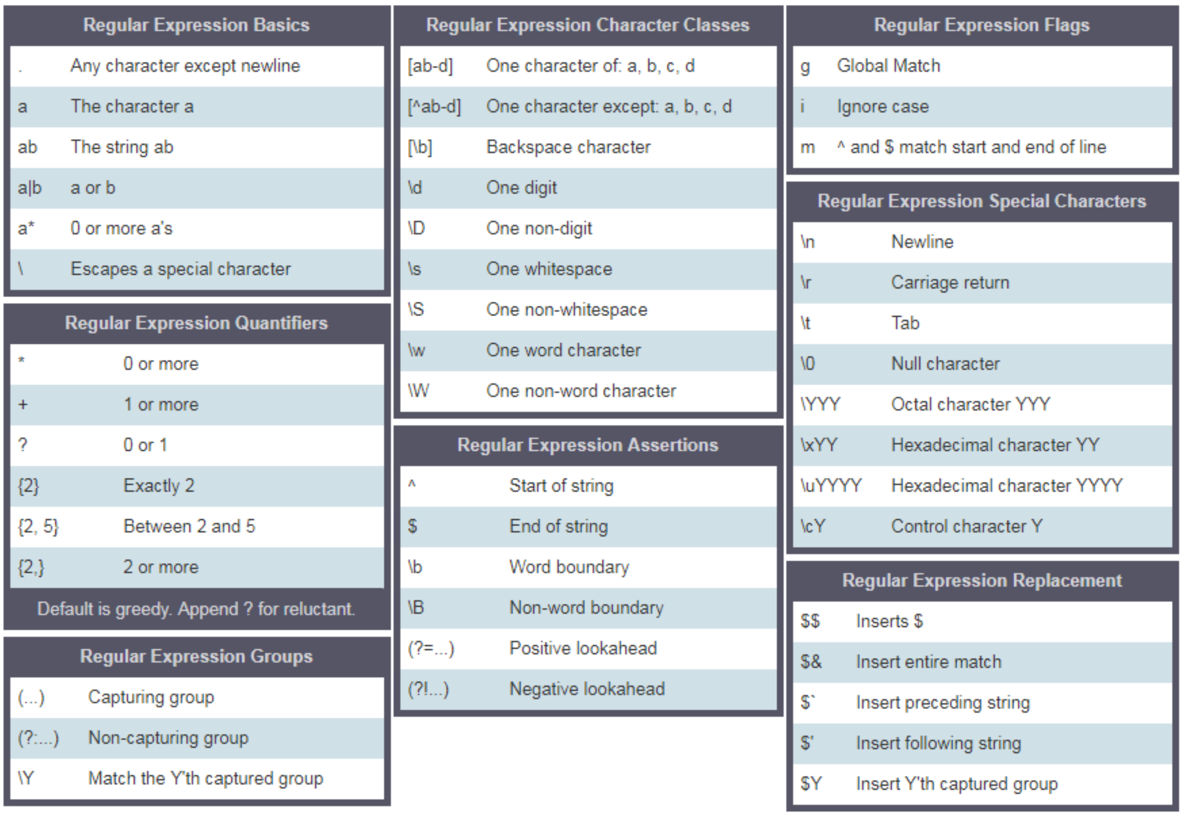
finally clause
If finally is present, it specifies a 'cleanup' handler. The try clause is executed, including any except and else clauses. If an exception occurs in any of the clauses and is not handled, the exception is temporarily saved. The finally clause is executed. If there is a saved exception it is re-raised at the end of the finally clause. If the finally clause raises another exception, the saved exception is set as the context of the new exception. If the finally clause executes a return, break or continue statement, the saved exception is discarded:

Raising Exception:
The raise statement allows us to force a specified exception to occure.

- **Regex:-**
  A regular expression or RegEx is a special text string that helps to find patterns in data. A RegEx can be used to check if some pattern exists in a different data type. **To use RegEx in python first we should import the RegEx module which is called *re*.**
  Methods in *re* Module
  To find a pattern we use different set of *re* character sets that allows to search for a match in a string.
- *re.match()*: **searches only in the beginning of the first line of the string** and returns matched objects if found, else returns None.

- *re.search*: Returns a match object if there is one anywhere in the string, including multiline strings.
- *re.findall*: Returns a list containing all matches
- *re.split*: Takes a string, splits it at the match points, returns a list
- *re.sub*: Replaces one or many matches within a string

Writing RegEx Patterns:
To declare a string variable we use a single or double quote. To declare RegEx variable *r''*. The following pattern only identifies apple with lowercase, to make it case insensitive either we should rewrite our pattern or we should add a flag



Regular Expression Basics	
.	Any character except newline
a	The character a
ab	The string ab
a|b	a or b
a*	0 or more a's
\	Escapes a special character

Regular Expression Quantifiers	
*	0 or more
+	1 or more
?	0 or 1
{2}	Exactly 2
{2, 5}	Between 2 and 5
{2,}	2 or more
Default is greedy. Append ? for reluctant.	

Regular Expression Groups	
(...)	Capturing group
(?:...)	Non-capturing group
\Y	Match the Y'th captured group

Regular Expression Character Classes	
[ab-d]	One character of: a, b, c, d
[^ab-d]	One character except: a, b, c, d
[\b]	Backspace character
\d	One digit
\D	One non-digit
\s	One whitespace
\S	One non-whitespace
\w	One word character
\W	One non-word character

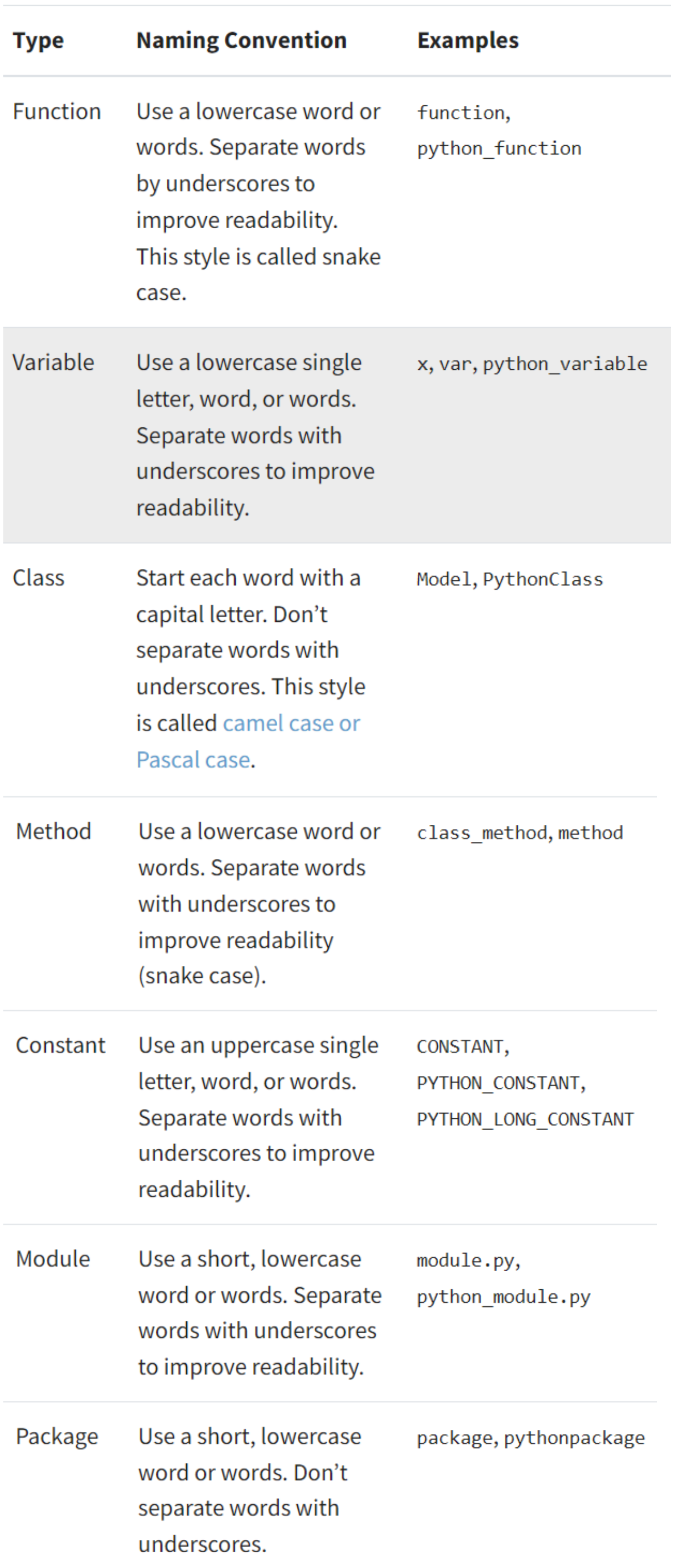
Regular Expression Assertions	
^	Start of string
$	End of string
\b	Word boundary
\B	Non-word boundary
(?=...)	Positive lookahead
(?!...)	Negative lookahead

Regular Expression Flags	
g	Global Match
i	Ignore case
m	^ and $ match start and end of line

Regular Expression Special Characters	
\n	Newline
\r	Carriage return
\t	Tab
\0	Null character
\YYY	Octal character YYY
\xYY	Hexadecimal character YY
\uYYYY	Hexadecimal character YYYY
\cY	Control character Y

Regular Expression Replacement	
$$	Inserts $
$&	Insert entire match
$`	Insert preceding string
$'	Insert following string
$Y	Insert Y'th captured group

- **Virtual Enviroment:-**
  Contains packages modules that are only required for particular projects.
- **PEP8:-**
  It is a guideline how to write a code which improve readability of code.
  Naming Conventions:

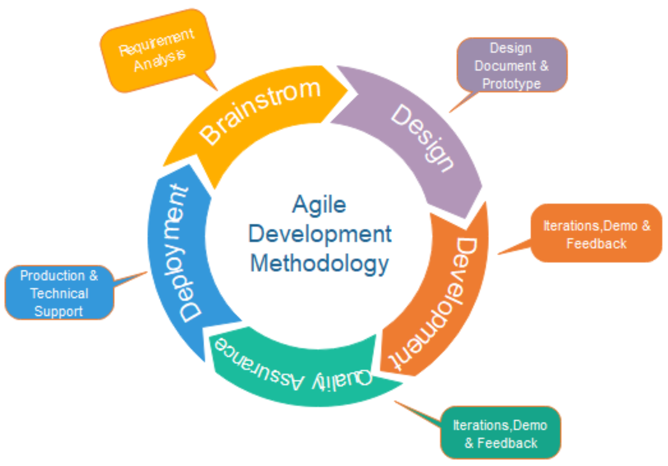


Type	Naming Convention	Examples
Function	Use a lowercase word or words. Separate words by underscores to improve readability. This style is called snake case.	`function`, `python_function`
Variable	Use a lowercase single letter, word, or words. Separate words with underscores to improve readability.	`x`, `var`, `python_variable`
Class	Start each word with a capital letter. Don't separate words with underscores. This style is called camel case or Pascal case.	`Model`, `PythonClass`
Method	Use a lowercase word or words. Separate words with underscores to improve readability (snake case).	`class_method`, `method`
Constant	Use an uppercase single letter, word, or words. Separate words with underscores to improve readability.	`CONSTANT`, `PYTHON_CONSTANT`, `PYTHON_LONG_CONSTANT`
Module	Use a short, lowercase word or words. Separate words with underscores to improve readability.	`module.py`, `python_module.py`
Package	Use a short, lowercase word or words. Don't separate words with underscores.	`package`, `pythonpackage`

We cannot remember so many rules so there are many software or tools such as linters and auto formatters which follows PEP8. Linters are programs that analyse code. Some good linters are RUFF , flask8 , pycodestyle. Where as auto formatters are programs that refactor your code with rules of pep 8 automatically.

- **Unit Testing:-**
  Unit Testing is a software testing technique in which individual units or components of a software application are tested in isolation. These units are the smallest pieces of code, typically functions or methods, ensuring they perform as expected.
  There are two types of unit testing:
  1) Manual:- Manual testing is like checking each part of a project by hand, without using any special tools. People, like developers, do each step of the testing themselves.
  2) Automated: Automated unit testing is a way of checking if software works correctly without needing lots of human effort

  Tools Used for unit testing are: LambdaTest, Junit, Mocha etc.
  Unit Testing Techniques
  There are 3 types of Unit Testing Techniques. They are follows
  - **Black Box Testing:** This testing technique is used in covering the unit tests for input, user interface, and output parts.
  - **White Box Testing:** This technique is used in testing the functional behavior of the system by giving the input and checking the functionality output including the internal design structure and code of the modules.
  - **Gray Box Testing:** This technique is used in executing the relevant test cases, test methods, and test functions, and analyzing the code performance for the modules.
- **Agile:-**
  It is a software development methodology. It breaks task into smaller iterations , each small iteration is consider as "frame".
  Phases of Agile Model:



Requirement Analysis
Brainstrom
Design Document & Prototype
Design
Agile Development Methodology
Iterations,Demo & Feedback
Development
Deployment
Production & Technical Support
Quality Assurance
Iterations,Demo & Feedback

Fig. Agile Model

Following are the phases in the Agile model are as follows:

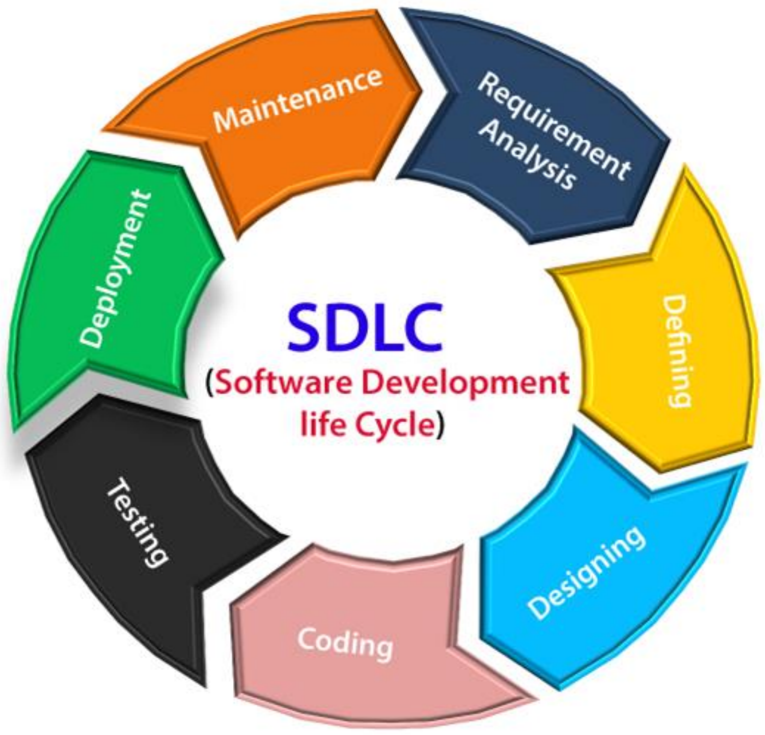
1. Requirements gathering: In this phase, you must define the requirements. You should explain business opportunities and plan the time and effort needed to build the project. Based on this information, you can evaluate technical and economic feasibility.
2. Design the requirements: When you have identified the project, work with stakeholders to define requirements. You can use the user flow diagram or the high-level UML diagram to show the work of new features and show how it will apply to your existing system.

3. Construction/ iteration: hen the team defines the requirements, the work begins. Designers and developers start working on their project, which aims to deploy a working product. The product will undergo various stages of improvement, so it includes simple, minimal functionality.
4. Testing/ Quality assurance: In this phase, the Quality Assurance team examines the product's performance and looks for the bug.
5. Deployment: In this phase, the team issues a product for the user's work environment.
6. Feedback:

Agile Testing Methods:

- Scrum
- Crystal
- Dynamic Software Development Method(DSDM)
- Feature Driven Development(FDD)
- Lean Software Development
- eXtreme Programming(XP)

- **SDLC:-**
  SDLC Cycle represents the process of developing software. SDLC framework includes the following steps:



The stages of SDLC are as follows:
Stage1: Planning and requirement analysis
Requirement Analysis is the most important and necessary stage in SDLC.

The senior members of the team perform it with inputs from all the stakeholders and domain experts or SMEs in the industry.
Planning for the quality assurance requirements and identifications of the risks associated with the projects is also done at this stage.

Stage2: Defining Requirements

Once the requirement analysis is done, the next stage is to certainly represent and document the software requirements and get them accepted from the project stakeholders.
This is accomplished through "SRS"- Software Requirement Specification document which contains all the product requirements to be constructed and developed during the project life cycle.

Stage3: Designing the Software

The next phase is about to bring down all the knowledge of requirements, analysis, and design of the software project. This phase is the product of the last two, like inputs from the customer and requirement gathering.

Stage4: Developing the project

In this phase of SDLC, the actual development begins, and the programming is built. The implementation of design begins concerning writing code. Developers have to follow the coding guidelines described by their management and programming tools like compilers, interpreters, debuggers, etc. are used to develop and implement the code.

Stage5: Testing

After the code is generated, it is tested against the requirements to make sure that the products are solving the needs addressed and gathered during the requirements stage.
During this stage, unit testing, integration testing, system testing, acceptance testing are done.

Stage6: Deployment

Once the software is certified, and no bugs or errors are stated, then it is deployed.
Then based on the assessment, the software may be released as it is or with suggested enhancement in the object segment.
After the software is deployed, then its maintenance begins.

Stage7: Maintenance

Once when the client starts using the developed systems, then the real issues come up and requirements to be solved from time to time.
This procedure where the care is taken for the developed product is known as maintenance.