OVERVIEW

- An event which causes the termination of the program.
- To handle unexpected termination of the program, exception handling is done.
- To handle the exception, try and except block is used.

Types

1. Default except block.
2. Specific except block.
3. Generic except block.
4. Multiple except block.

Default except block

Syntax :

```
try:
    statements
except:
    statements
```

Multiple except block

- A single try block can have multiple except blocks.

Syntax :

```
try :
    statements
except exception1:
    statements
except exception2 :
    Statements
```

Generic except block

- Handles all types of exceptions in a single except block.

Syntax :

```
try :
    statements
except Exception/BaseException :
    Statements
```

Nested try and except block

Syntax :

```
try:
    statements
    try:
        statements
    except:
        statements
except:
    nested try – except block
```

"as" keyword

- Used to give alias name for the exception names written in the except block.
- **Syntax :** except <exception name> as alias_name:

Raise keyword

- Used to raise a specific exception whenever the condition is matched.
- Once an exception is raised, it searches for the specific except block and handles th exception.

Syntax : raise error_name("message")

Finally block

- It is a block which will get executed even when the exception is raised or not.
- We can add try and except block inside finally.

Syntax :

```
try :
    statements
except:
    statements
finally :
    Statements
```

Else block

- It is a block which will get executed even when the exception is not raised.

Syntax :

```
try :
    statements
except:
    statements
else:
    Statements
```

User defined exceptions or custom exception

- Custom exceptions can be created by Inheriting Exception class.
- **Syntax :**

```
class user_exception_name(Exception):

    pass
```