Computer Science

Class XII (As per CBSE Board)

New syllabus 2023-24

Chapter 3

Exception handling

Visit : python.mykvs.in for regular updates

Exception Handling

Error in Python can be of two types i.e. Syntax errors and Exceptions. Errors are problems in a program due to which will stop the program from execution. On the other hand, exceptions are raised when some internal events occur due to limitation of hardware or software part, which change the normal flow of the program.

Different types of exceptions in python:

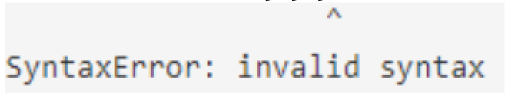
- SyntaxError: This exception is raised when the interpreter encounters a syntax error in the code, such as a misspelled keyword, a missing colon, or an unbalanced parenthesis.
- TypeError: This exception is raised when an operation or function is applied to an object of the wrong type, such as adding a string to an integer.
- NameError: This exception is raised when a variable or function name is not found in the current scope.
- IndexError: This exception is raised when an index is out of range for a list, tuple, or other sequence types.
- KeyError: This exception is raised when a key is not found in a dictionary.
- ValueError: This exception is raised when a function or method is called with an invalid argument or input, such as trying to convert a string to an integer when the string does not represent a valid integer.
- AttributeError: This exception is raised when an attribute or method is not found on an object, such as trying to access a non-existent attribute of a class instance.
- IOError: This exception is raised when an I/O operation, such as reading or writing a file, fails due to an input/output error.
- ZeroDivisionError: This exception is raised when an attempt is made to divide a number by zero.
- ImportError: This exception is raised when an import statement fails to find or load a module.

Difference between Syntax Error and Exceptions

Syntax Error: This error is caused by the wrong syntax in the code.

```
if(amount > 999)
        print("amount more than 1000")
```

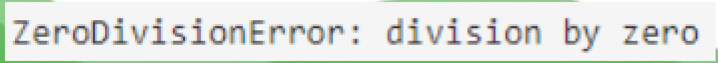
```
if(amount > 999)
```



```
                  ^
SyntaxError: invalid syntax
```

Exceptions: Exceptions are raised when the program is syntactically correct, but the code results in an error. This error does not stop the execution of the program, however, it changes the normal flow of the program.

```
a = 10 / 0
print(a)
```



```
ZeroDivisionError: division by zero
```

handling exceptions using try-except-finally blocks

```
try:
    # Some Code....which may have runtime error

except:
    # optional block
    # Handling of exception (if required)

else:
    # execute if no exception

finally:
    # Some code .....(always executed)
```

```
try:
    k = 9//0  # raises divide by zero exception.
    print(k)

# handles zerodivision exception
except ZeroDivisionError:
    print("Can't divide by zero")

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

Advantages of Exception Handling:

- Improved program reliability
- Simplified error handling
- Cleaner code
- Easier debugging

Disadvantages of Exception Handling:

- Performance overhead
- Increased code complexity
- Possible security risks