Python Programming - 2301CS404

Lab - 10

Name: Jadeja Rudrarajsinh

Enrollment No:23010101411

Roll No:487

Exception Handling

01) WAP to handle following exceptions:

1. ZeroDivisionError
2. ValueError
3. TypeError

Note: handle them using separate except blocks and also using single except block too.

In [2]:
```
try:
    x=int(input("enter no : "))
    y=int(input("enter no : "))

    result=x/y

    print("result",result)
except ZeroDivisionError:
    print("cannot be divide wd 0")
except ValueError:
    print("invalid input")
except TypeError:
    print("type mistake")
```

```
result 1.0909090909090908
```

02) WAP to handle following exceptions:

1. IndexError
2. KeyError

In [12]:
```
list=[1,2,3]
dic={"name":"JP","roll":7}
try:
    print(list[2])
    print(dic["no"])

except IndexError:
    print("index error")

except KeyError:
    print("key error")
```

```
3
key error
```

03) WAP to handle following exceptions:

1. FileNotFoundError
2. ModuleNotFoundError

In [16]:
```
try:
 fp = open("abc1.txt","r")
except FileNotFoundError as err:
 print(err)
else:
 print(fp.read())
 fp.close()

try:
    import non_existent_module
except ModuleNotFoundError as a:
    print("enter : ",str(a))
```

```
Jay mataji from Jadeja Rudrarajsinh!!!!!!
enter :  No module named 'non_existent_module'
```

04) WAP that catches all type of exceptions in a single except block.

In [18]:
```
try:
    a=int(input("1st no :"))
    b=int(input("2nd no :"))
    c=a/b
    print("result :",c)
except Exception as e:
    print(f"Error:{e}")
```

```
result : 0.5
```

05) WAP to demonstrate else and finally block.

In [22]:
```
try:
    a=int(input("1st no :"))
    b=int(input("2nd no :"))
    c=a/b
except Exception as e:
    print("Error:{e}")
else:
    print("result:",result)
finally:
    print("done")
```

```
result: 1.0909090909090908
done
```

06) Create a short program that prompts the user for a list of grades separated by commas.

Split the string into individual grades and use a list comprehension to convert each string to an integer.

You should use a try statement to inform the user when the values they entered cannot be converted.

In [36]:
```
grade_input = input("enter grade")
try:
    grades=[int(grade) for grade in grade_input.split(',')]
    print("grades",grades)
except ValueError:
    print("error: plz enter valid grades separated by commas,with no non-numeric
```

```
grades [2]
```

07) WAP to create an udf divide(a,b) that handles ZeroDivisionError.

In [24]:
```
def divide(a,b):
    try:
        result = a/b
        return result
    except ZeroDivisionError:
        print("error: cannot divide by 0")
        return None
print(divide(10,2))
print(divide(10,0))
```

```
5.0
error: cannot divide by 0
None
```

08) WAP that gets an age of a person form the user and raises ValueError with error message: "Enter Valid Age" :

If the age is less than 18.

otherwise print the age.

In [26]:
```
try:
    age = int(input("Enter your age: "))
    if age < 18:
        raise ValueError("Enter Valid Age")
    print("Your age is:", age)
except ValueError as e:
    print("Error:", e)
```

```
Your age is: 20
```

09) WAP to raise your custom Exception named InvalidUsernameError with the error message : "Username must be between 5 and 15 characters long":

if the given name is having characters less than 5 or greater than 15.

otherwise print the given username.

In [30]:
```
try:
    un = input("Enter your username: ")

    if len(un) < 5 or len(un) > 15:
        raise Exception("Username must be between 5 and 15 characters long")
    print("Your username is:", un)
except Exception as e:
    print("Error:", e)
```

```
Your username is: rudrarajsinh_9
```

10) WAP to raise your custom Exception named NegativeNumberError with the error message : "Cannot calculate the square root of a negative number" :

if the given number is negative.

otherwise print the square root of the given number.

In [32]:
```
import math
try:
    num = float(input("Enter a number: "))
    if num < 0:
        raise Exception("Cannot calculate the square root of a negative number")
    print("Square root:", math.sqrt(num))
except Exception as e:
    print("Error:", e)
except ValueError:
    print("Error: Please enter a valid number")
```

```
Square root: 3.0
```