Practical No 8

Aim: Write a program to define and raise custom exceptions in Python using user-defined exception classes.

Theory:

A **user-defined exception** (also called a **custom exception**) is an error type that a programmer creates to handle application-specific error conditions.
It is defined as a **class that inherits from Python's built-in `Exception` class** (or its subclasses).

These exceptions allow you to:

- Give **meaningful names** to errors (e.g., `InvalidAgeError`, `InsufficientFundsError`).
- Provide **custom error messages** or extra information.
- Organize errors into an **exception hierarchy** for better handling.

Syntax

```
class CustomException(Exception):
    """Description of the custom exception."""
    pass
```

Simple Example

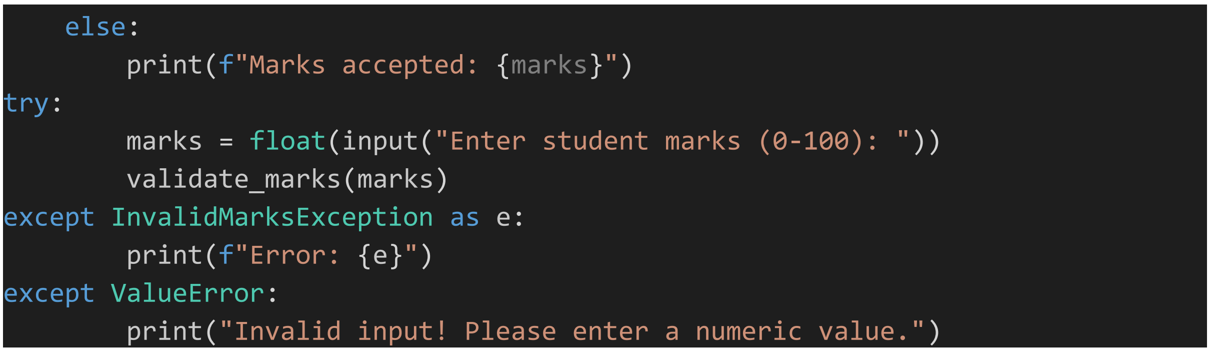
```
class InvalidAgeError(Exception):
    """Raised when the entered age is not valid."""
    pass

# Using the custom exception
age = 15
if age < 18:
    raise InvalidAgeError("Age must be at least 18 to vote.")
```

Problem Statemens:

1. Write a Python program to validate student marks using a **user-defined exception**. Create a custom exception class InvalidMarksError that is raised if marks are **less than 0 or greater than 100**; otherwise, accept and display the marks.

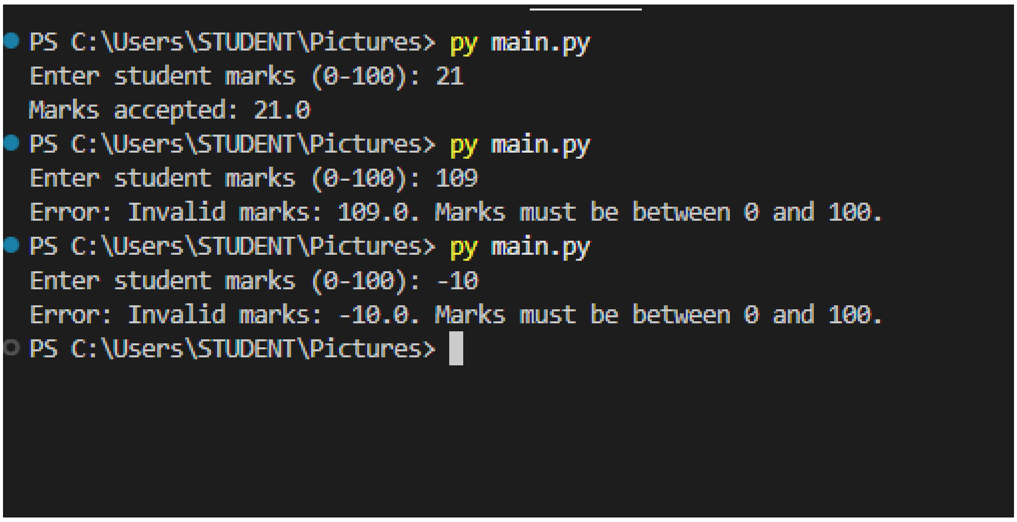
Program:

```
class InvalidMarksException(Exception):
    pass
def validate_marks(marks):
    if marks < 0 or marks > 100:
        raise InvalidMarksException(f"Invalid marks: {marks}. Marks must be between 0 and 100.")
```



```
    else:
        print(f"Marks accepted: {marks}")
try:
        marks = float(input("Enter student marks (0-100): "))
        validate_marks(marks)
except InvalidMarksException as e:
        print(f"Error: {e}")
except ValueError:
        print("Invalid input! Please enter a numeric value.")
```

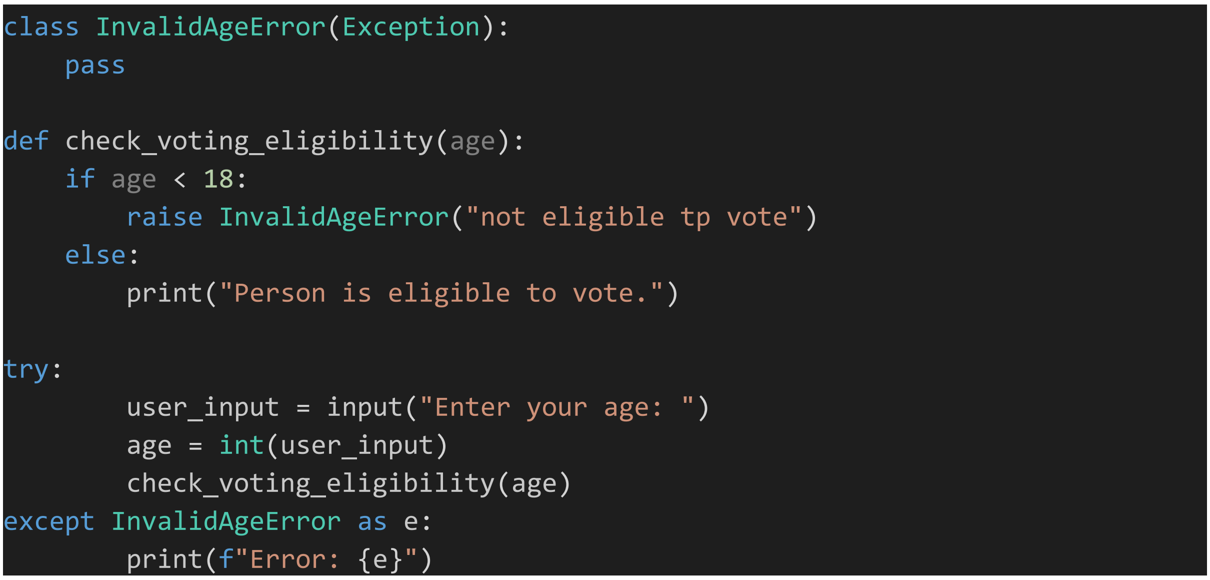
Output:



```
PS C:\Users\STUDENT\Pictures> py main.py
Enter student marks (0-100): 21
Marks accepted: 21.0
PS C:\Users\STUDENT\Pictures> py main.py
Enter student marks (0-100): 109
Error: Invalid marks: 109.0. Marks must be between 0 and 100.
PS C:\Users\STUDENT\Pictures> py main.py
Enter student marks (0-100): -10
Error: Invalid marks: -10.0. Marks must be between 0 and 100.
PS C:\Users\STUDENT\Pictures> 
```

2. Write a Python program to demonstrate **user-defined exceptions** by creating a custom exception class InvalidAgeError. Raise this exception if the entered age is **less than 18**, otherwise display that the person is eligible to vote.

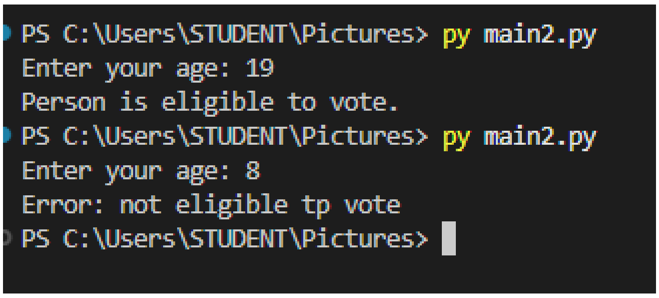
Program:



```
class InvalidAgeError(Exception):
    pass

def check_voting_eligibility(age):
    if age < 18:
        raise InvalidAgeError("not eligible tp vote")
    else:
        print("Person is eligible to vote.")

try:
        user_input = input("Enter your age: ")
        age = int(user_input)
        check_voting_eligibility(age)
except InvalidAgeError as e:
        print(f"Error: {e}")
```

```
except ValueError:
        print("Invalid input! Please enter a valid integer age.")
```

Output:



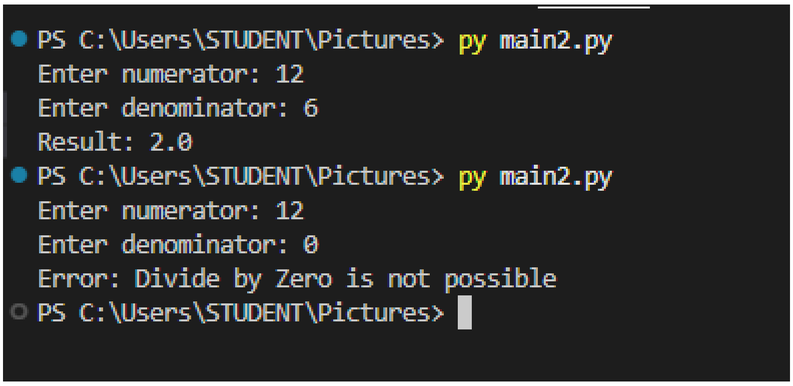
```
PS C:\Users\STUDENT\Pictures> py main2.py
Enter your age: 19
Person is eligible to vote.
PS C:\Users\STUDENT\Pictures> py main2.py
Enter your age: 8
Error: not eligible tp vote
PS C:\Users\STUDENT\Pictures>
```

3. Write a Python program to demonstrate **user-defined exceptions** by creating a custom exception class DivisionByZeroError. Raise this exception when the denominator is **zero**, since division by zero is not allowed.

Program:

```
class DivisionByZeroError(Exception):
    pass

def divide(a, b):
    if b == 0:
        raise DivisionByZeroError("Divide by Zero is not possible")
    return a / b

try:
        num = float(input("Enter numerator: "))
        denom = float(input("Enter denominator: "))
        result = divide(num, denom)
        print(f"Result: {result}")
except DivisionByZeroError as e:
        print(f"Error: {e}")
except ValueError:
        print("Invalid input! Please enter numeric values.")
```

Output:



```
PS C:\Users\STUDENT\Pictures> py main2.py
Enter numerator: 12
Enter denominator: 6
Result: 2.0
PS C:\Users\STUDENT\Pictures> py main2.py
Enter numerator: 12
Enter denominator: 0
Error: Divide by Zero is not possible
PS C:\Users\STUDENT\Pictures>
```

Conclusion:

The above code is executed successfully.