1.Write a C++ program that demonstrates exception handling using custom exception classes without inheriting from std::exception.

1. Create a base exception class MathException (without inheriting from std::exception).

2. Derive two exception classes:

    - DivideByZeroException for division by zero.

    - NegativeSquareRootException for taking the square root of a negative number.

3. Implement two functions:

    - divide(int a, int b): Throws DivideByZeroException if b == 0.

    - safeSqrt(double x): Throws NegativeSquareRootException if x < 0.

4. Handle exceptions in main() and ensure the program continues execution.

2.Write a C++ program that simulates a bank account system using custom exception classes without inheriting from std::exception.

1. Create a base exception class BankException (without inheriting from std::exception).

2. Derive two exception classes:

    - InsufficientFundsException: Thrown when attempting to withdraw more than the available balance.

    - NegativeDepositException: Thrown when attempting to deposit a negative amount.

3. Implement a BankAccount class with:

    - deposit(double amount): Throws NegativeDepositException if amount < 0.

- withdraw(double amount): Throws InsufficientFundsException if amount > balance.

4. Handle exceptions in main() and ensure the program continues execution.

3.Write a C++ program that manages student grades using exception handling.

1. Create a base exception class (GradeException) inheriting from std::exception.
2. Derive two exception classes:
   - InvalidGradeException: Thrown when a grade is outside the valid range (0-100).
   - FailingGradeException: Thrown when a student's grade is below passing (e.g., < 40).
3. Implement a Student class with:
   - setGrade(int grade): Throws InvalidGradeException if grade < 0 or grade > 100.
   - checkPass(): Throws FailingGradeException if grade < 40.
4. Handle exceptions in main() and ensure the program continues execution.

4.Write a C++ program that simulates an ATM transaction system with custom exception handling.

1. Create a base exception class (ATMException) inheriting from std::exception.
2. Derive two exception classes:
   - InvalidPINException: Thrown when an incorrect PIN is entered more than 3 times.
   - InsufficientBalanceException: Thrown when trying to withdraw more than the account balance.

3. Implement an ATM class with:

    - validatePIN(int enteredPIN): Throws InvalidPINException after 3 incorrect attempts.

    - withdraw(double amount): Throws InsufficientBalanceException if amount > balance.

4. Handle exceptions in main() and ensure the program continues execution.