EXCEPTION HANDLING

AIM:

To demonstrate the exception handling in c++ using the try,catch.

ALGORITHM:

· Create a BankAccount class and initialize balance to zero.

· Accept user choice using a menu-driven approach.

· For deposit, check for invalid amount and throw an exception.

· Catch the exception and display an error message.

· For withdrawal, check for insufficient balance and throw an exception.

· Catch the exception and display an error message.

· Display the current balance when requested.

· Repeat the process until exit option is selected.

PROGRAM:

```
/*
*  Program to demonstrate Exception Handling in C++
*  Author   : MUTHUGANESH S
*  Date     : 28/1/2026
*  Filename: ExceptionHandling.cpp
*  retval   : void
*/

#include <iostream>
using namespace std;

class BankAccount {
    double Balance;
public:
// Constructor to initialize balance
    BankAccount() : Balance(0.0) {}

    // Function to deposit money
    void Deposit(double Amount) {
        // Exception handling for invalid deposit amount
        try{
            if(Amount <= 0)
                throw Amount;
            Balance += Amount;
        }
```

```
        catch(double Amount){
            cout<< "Exception: Invalid deposit amount: " << Amount << std::endl;
        }

    }

    // Function to withdraw money
    void Withdraw(double Amount) {
    // Exception handling for insufficient funds
        try{
            if(Amount>Balance)
                throw Amount;
            Balance -= Amount;
        }

        catch(double Amount){
            cout<< "Exception: Insufficient funds for withdrawal of: " <<
Amount <<endl;
        }

    }

    // Function to get current balance
    double GetBalance() {
        return Balance;
    }
};

int main(){

    BankAccount Account;
    int Choice;
    double Amount;
    cout << "1. Deposit\n2. Withdraw\n3. Balance\n4. Exit\n\n";

    // Menu-driven interface
    do {
        cout << "Enter your choice: ";
        cin >> Choice;

        switch (Choice) {
            case 1:
                cout << "Enter the amount to deposit: ";
                cin >> Amount;
                Account.Deposit(Amount);
                break;

            case 2:
                cout << "Enter the amount to withdraw: ";
                cin >> Amount;
                Account.Withdraw(Amount);
                break;

            case 3:
```

```
                cout << "Current balance: " << Account.GetBalance() << endl;
                break;

            case 4:
                cout << "Exiting..." << endl;
                break;

            default:
                cout << "Invalid choice. Please try again." << endl;
        }
    } while (Choice != 4);

    return 0;
}
```

OUTPUT: