#include <iostream>

#include <string>

#include <stdexcept>

using namespace std;

class Publication {

protected:

string title;

float price;

public:

Publication() : title(""), price(0.0) {}

virtual void getData() {

cout << "Enter title: ";

cin.ignore();

getline(cin, title);

cout << "Enter price: ";

cin >> price;

}

virtual void putData() const {

cout << "Title: " << title << endl;

cout << "Price: " << price << endl;

}

};

class Book : public Publication {

private:

int page\_count;

public:

Book() : page\_count(0) {}

void getData() override {

Publication::getData();

cout << "Enter page count: ";

cin >> page\_count;

if (cin.fail()) throw invalid\_argument("Invalid input for page count.");

}

void putData() const override {

Publication::putData();

cout << "Page Count: " << page\_count << endl;

}

};

class Tape : public Publication {

private:

float playing\_time;

public:

Tape() : playing\_time(0.0) {}

void getData() override {

Publication::getData();

cout << "Enter playing time (in minutes): ";

cin >> playing\_time;

if (cin.fail()) throw invalid\_argument("Invalid input for playing time.");

}

void putData() const override {

Publication::putData();

cout << "Playing Time (minutes): " << playing\_time << endl;

}

};

int main() {

Book book;

Tape tape;

try {

cout << "Enter book details:" << endl;

book.getData();

} catch (const exception& e) {

cerr << "Exception: " << e.what() << endl;

book = Book(); // Reset to default values

}

try {

cout << "Enter tape details:" << endl;

tape.getData();

} catch (const exception& e) {

cerr << "Exception: " << e.what() << endl;

tape = Tape(); // Reset to default values

}

cout << "\nBook Details:" << endl;

book.putData();

cout << "\nTape Details:" << endl;

tape.putData();

return 0; }