# Ch 11 project

Goals:

1. Understand and use delegates as function pointers
2. Understand and implement events
3. Understand Actions, Function and Lambdas

## Overview

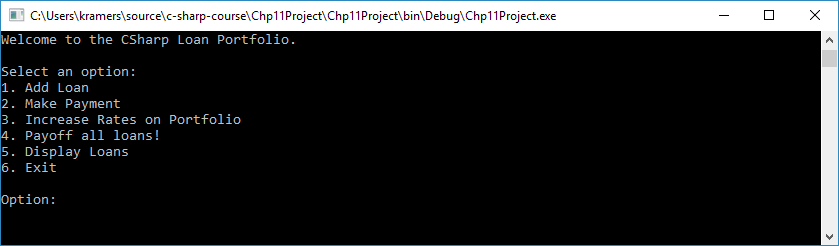
To exercise all these concepts, you’ll create the following classes/delegates:

A screenshot of a social media post

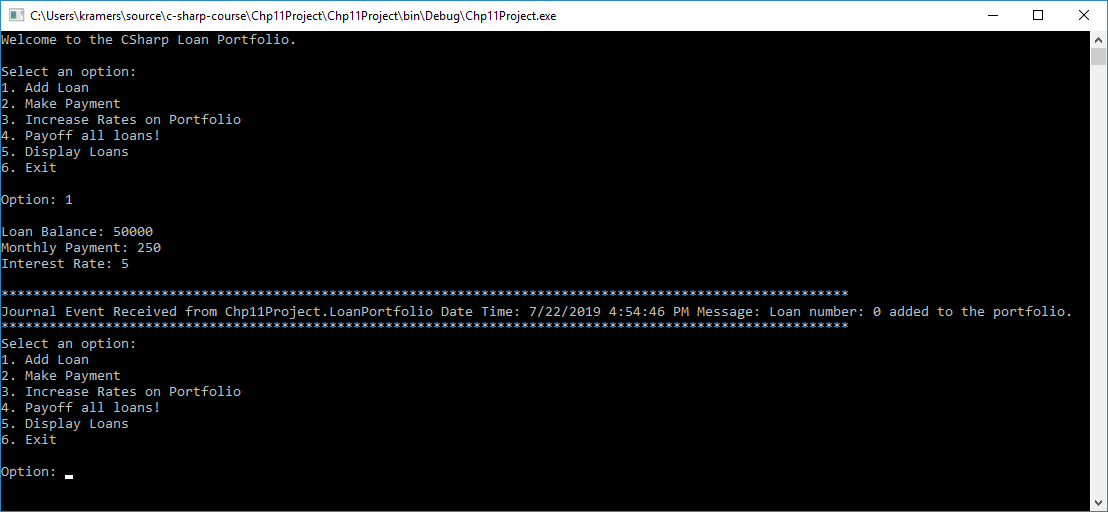
Description generated with very high confidence

The LoanPortfolio class maintains a generic collection of the Loan class. You’ll write a console application (i.e. the Program class in the diagram above) that will add loans, make payments, do bulk actions to every loan in the portfolio and log all events to the console.

Here’s a sample run of the application:

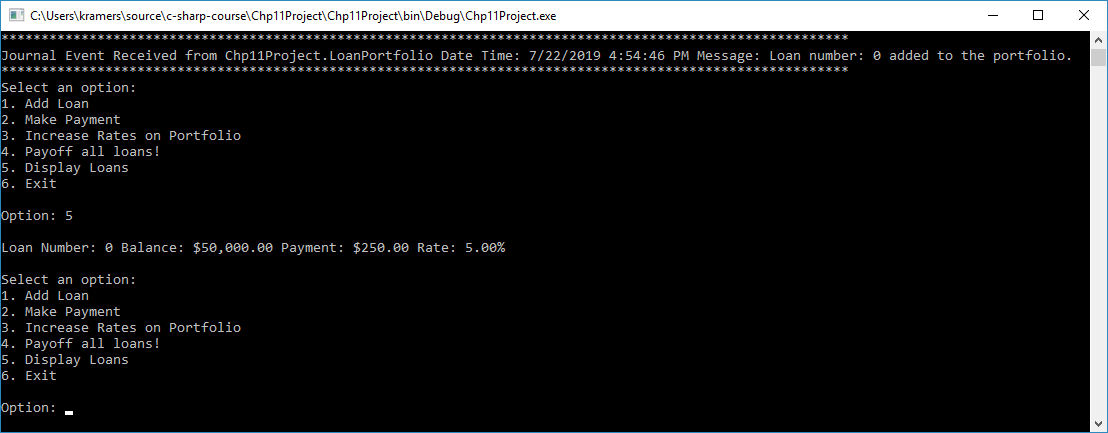


Enter option 1 Add Loan:

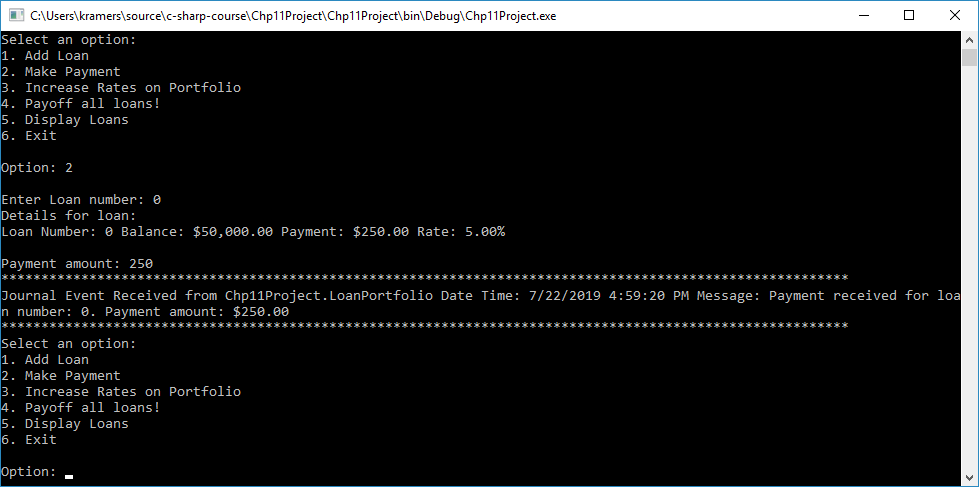


**Note: journal event is printed when a loan is added to the portfolio.**

Display all loans:



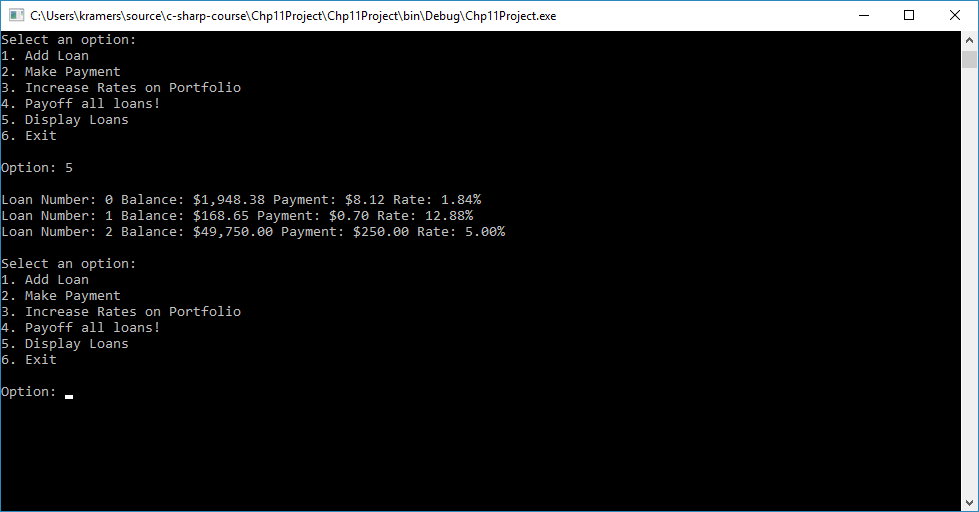
Make a payment:



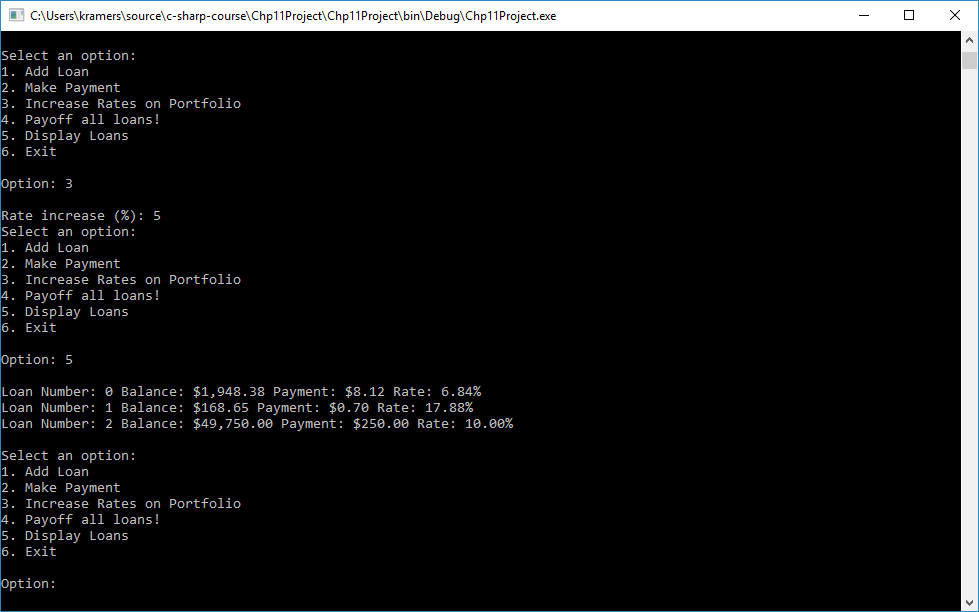
**Note: the journal event printed when a payment is made on a loan.**

I added a few more loans to show the increase rates on portfolio and payoff all loans functionality.

Display all loans:

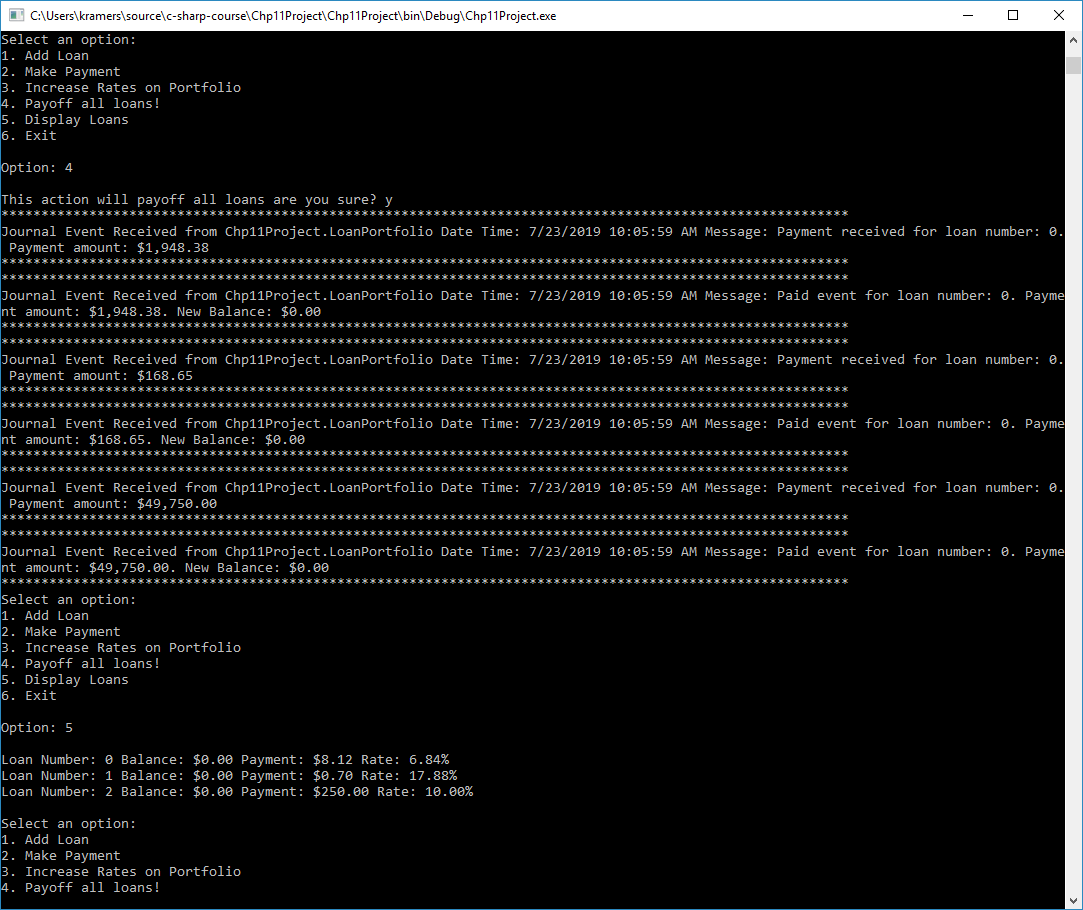


Increase Rates on Portfolio:



The rate on all loans in the portfolio has been increased by 5%.

Payoff all loans!



**Note: journal event printed when a loan is paid out.**

## Class overview

LoanPortfolio class:   
Contains a generic collection of the Loan class.

**AddLoan method:**Add the loan to the portfolio and call the EventJournal delegate to journal the loan number of the loan added. Register for PaidOut and ReceivedPayment events from the Loan class. When those events are received journal the loan number, payment amount, and new balance.

**FindLoans method:**Takes a Func<Loan, bool>. You will pass a lambda into this function to retrieve Loans by the criteria you need to find the loan by. For example to find all loans with a balance greater then 10000 you call FindLoans as follows:

IEnumerable<Loan> loans = portfolio.FindLoans((l) => l.Balance > 10000);

**PortfolioAction method:**Takes an Action<Loan> parameter. Runs the action for every loan in the portfolio.

**Journal Property:**Delegate property of type EventJournal.

Loan class:  
Contains the Rate, LoanNumber, Balance, and PaymentAmount for a loan.

**ReceivedPayment event:**This event is fired whenever a payment is made to the loan. Create a PaymentEventArgs with appropriate data and fire the event.

**PaidOut event:**This event is fired whenever the balance is less than or equal to 0 on a loan. Create a PaymentEventArgs with appropriate data and fire the event.

**ToString method:**Override the ToString method to return a string with Loan Number, Balance, Payment, and Rate. This is a good place for a StringBuilder.

Program class:  
This driver program will provide the menu prompt and accept input to add loans, make payments, increase rates on all loans and payoff all loans. In addition, the Program will provide the delegate function handler for the LoanPortfolio classes EventJournal delegate.

## Additional Instructions

All logging to the console takes place in the Program.cs file. No other class in the object model will do Console.WriteLine except the function in Program.cs that the EventJournal delegate points to.