Financial Engineering Mathematics

財務工程數學

NTUST/First Semester, 2019

昀騰金融科技

Wintom Financial Technology

技術長 CTO

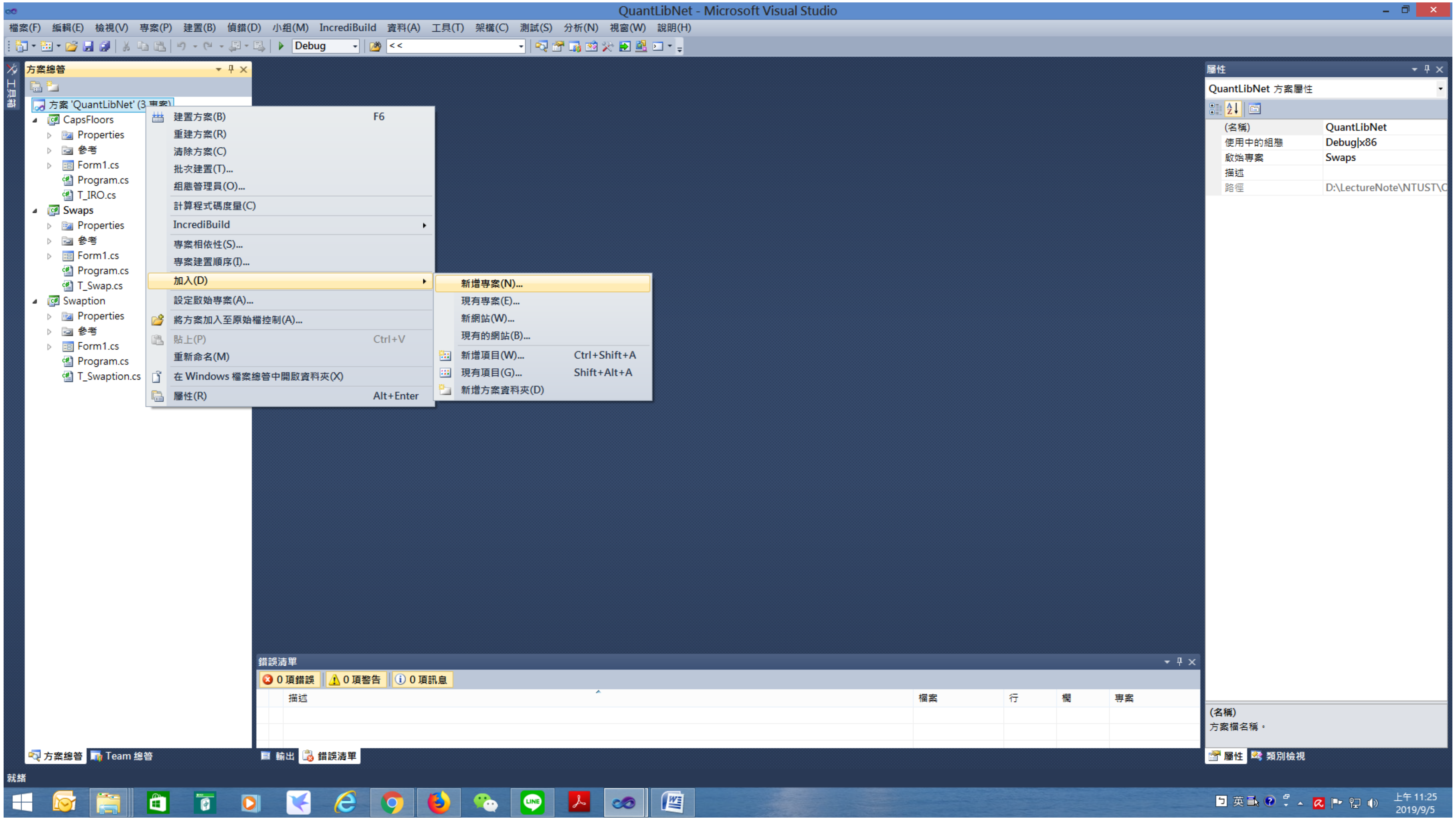
董夢雲 博士 Dr. Andy Dong

dongmy@ms5.hinet.net

Contents

3.1 Swaps Calculation

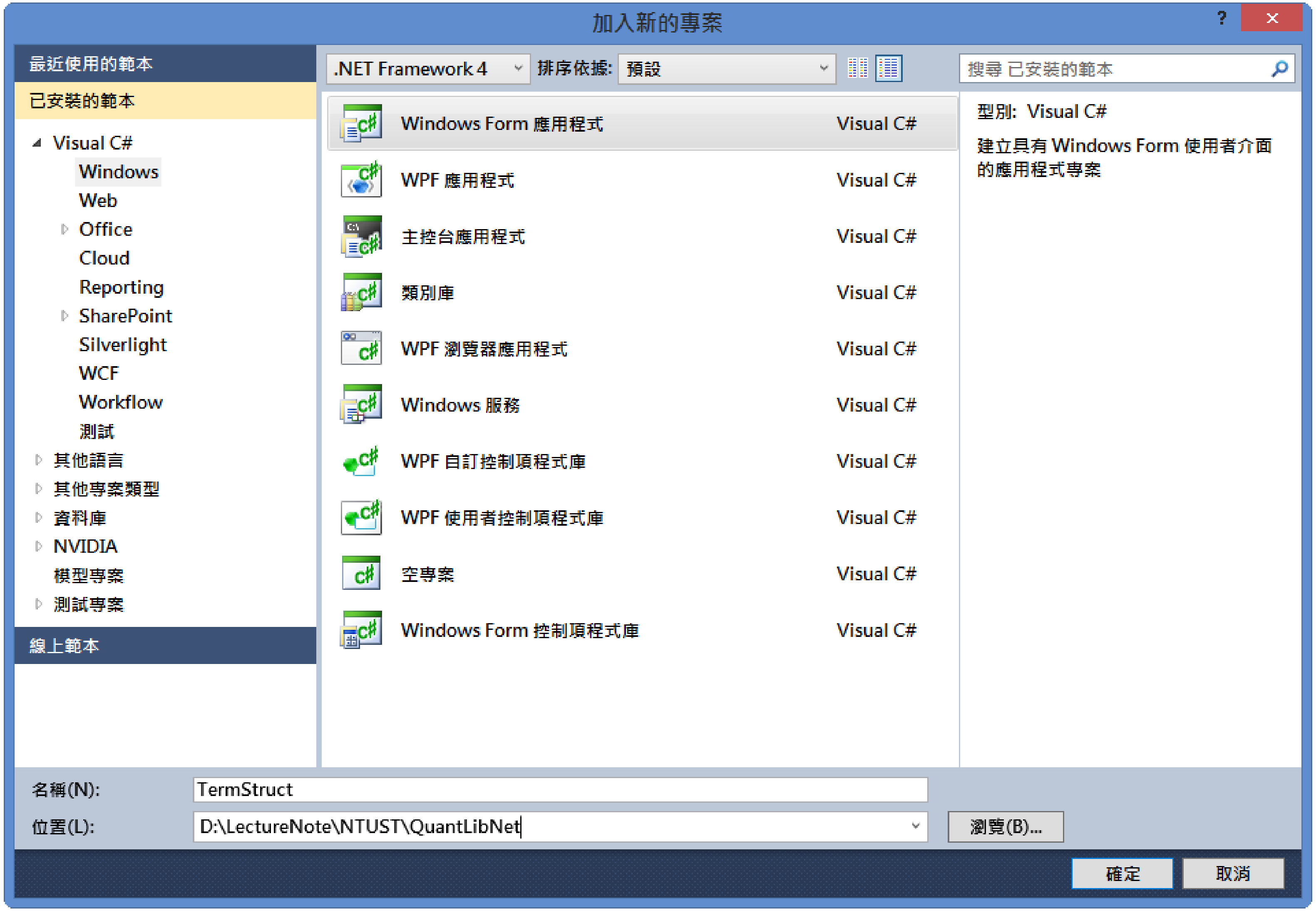
◆ Add New Project



(名稱)	QuantLibNet
使用中的組態	Debug|x86
啟始專案	Swaps
描述	
路徑	D:\LectureNote\NTUST\C

(名稱)
方案檔名稱。

◆ Name：Swaps，Windows Form Application。



Windows Form 應用程式	Visual C#
WPF 應用程式	Visual C#
主控台應用程式	Visual C#
類別庫	Visual C#
WPF 瀏覽器應用程式	Visual C#
Windows 服務	Visual C#
WPF 自訂控制項程式庫	Visual C#
WPF 使用者控制項程式庫	Visual C#
空專案	Visual C#
Windows Form 控制項程式庫	Visual C#

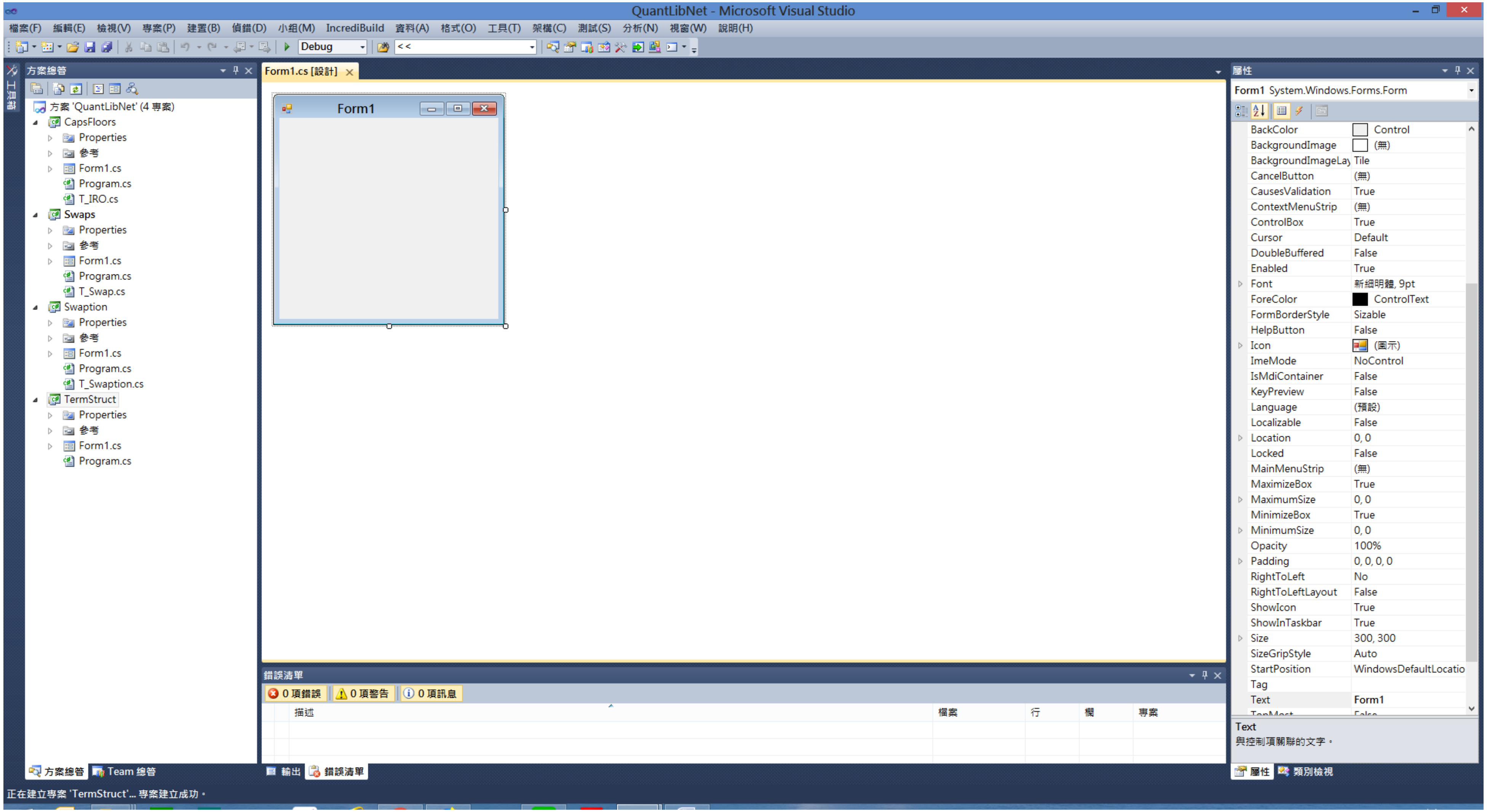
型別: Visual C#

建立具有 Windows Form 使用者介面的應用程式專案

名稱(N): TermStruct

位置(L): D:\LectureNote\NTUST\QuantLibNet

◆ Create New Form


QuantLibNet - Microsoft Visual Studio
檔案(F) 編輯(E) 檢視(V) 專案(P) 建置(B) 偵錯(D) 小組(M) IncrediBuild 資料(A) 格式(O) 工具(T) 架構(C) 測試(S) 分析(N) 視窗(W) 說明(H)
Debug
方案總管
Form1.cs [設計]
Form1
屬性
錯誤清單
0 項錯誤
0 項警告
0 項訊息
描述
檔案
行
欄
專案
方案總管
Team 總管
輸出
錯誤清單
屬性
類別檢視

- 方案 'QuantLibNet' (4 專案)
  - CapsFloors
    - Properties
    - 參考
    - Form1.cs
    - Program.cs
    - T_IRO.cs
  - Swaps
    - Properties
    - 參考
    - Form1.cs
    - Program.cs
    - T_Swap.cs
  - Swaption
    - Properties
    - 參考
    - Form1.cs
    - Program.cs
    - T_Swaption.cs
  - TermStruct
    - Properties
    - 參考
    - Form1.cs
    - Program.cs

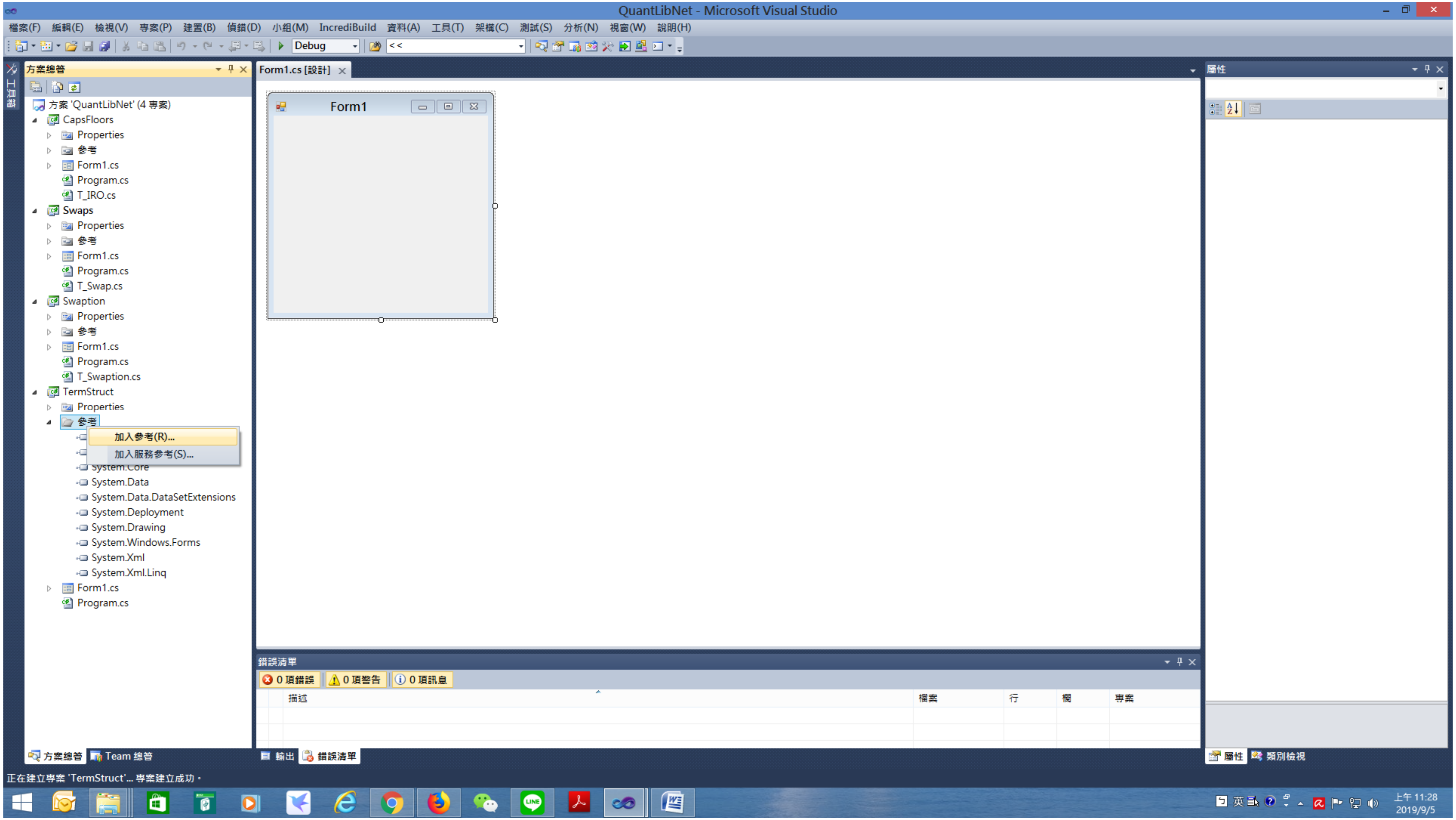
Form1 System.Windows.Forms.Form

BackColor	Control
BackgroundImage	(無)
BackgroundImageLay	Tile
CancelButton	(無)
CausesValidation	True
ContextMenuStrip	(無)
ControlBox	True
Cursor	Default
DoubleBuffered	False
Enabled	True
Font	新細明體, 9pt
ForeColor	ControlText
FormBorderStyle	Sizable
HelpButton	False
Icon	(圖示)
ImeMode	NoControl
IsMdiContainer	False
KeyPreview	False
Language	(預設)
Localizable	False
Location	0, 0
Locked	False
MainMenuStrip	(無)
MaximizeBox	True
MaximumSize	0, 0
MinimizeBox	True
MinimumSize	0, 0
Opacity	100%
Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	300, 300
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocatio
Tag	
Text	Form1

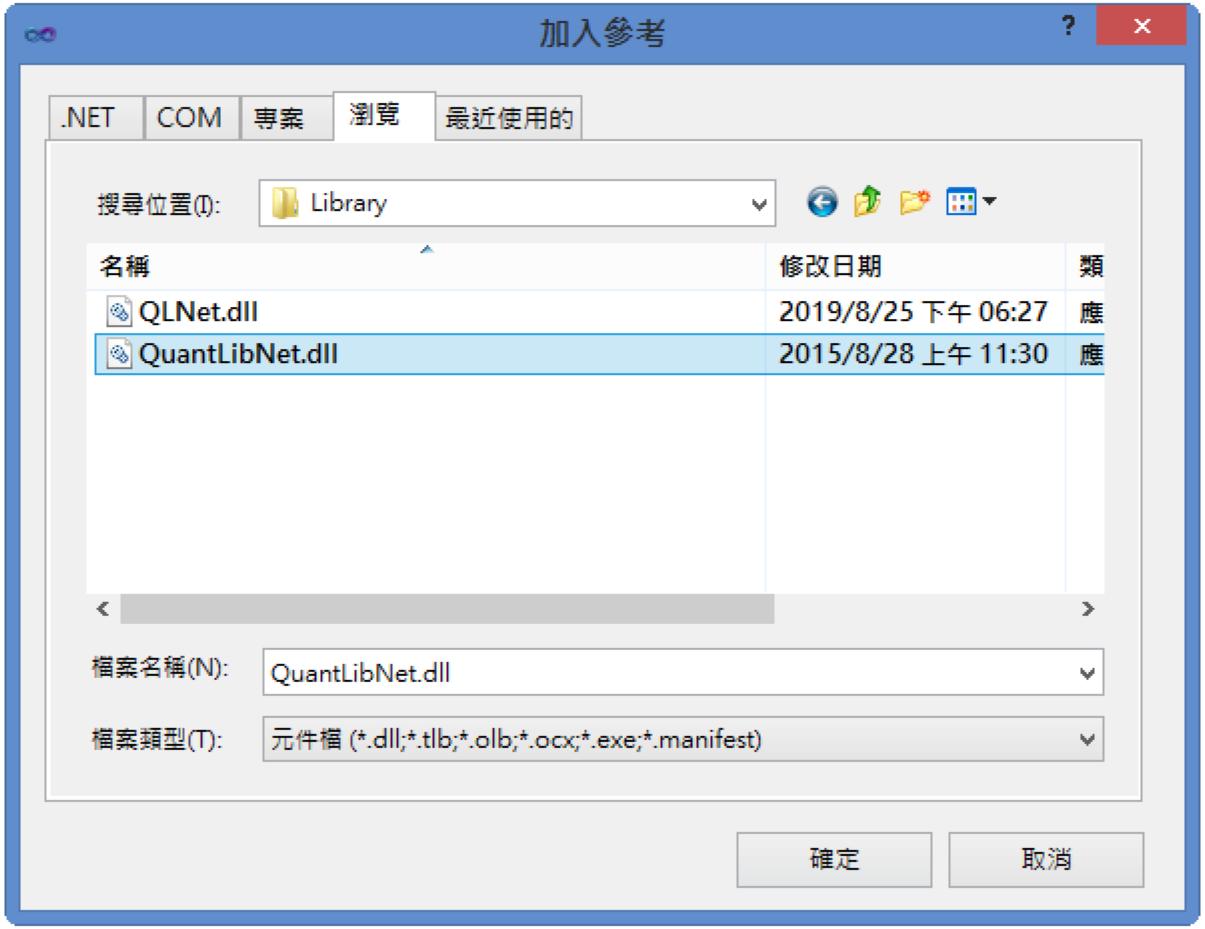
Text
與控制項關聯的文字。

正在建立專案 'TermStruct'... 專案建立成功。

➢ Add Reference QuantLibNet.dll。



正在建立專案 'TermStruct'... 專案建立成功。



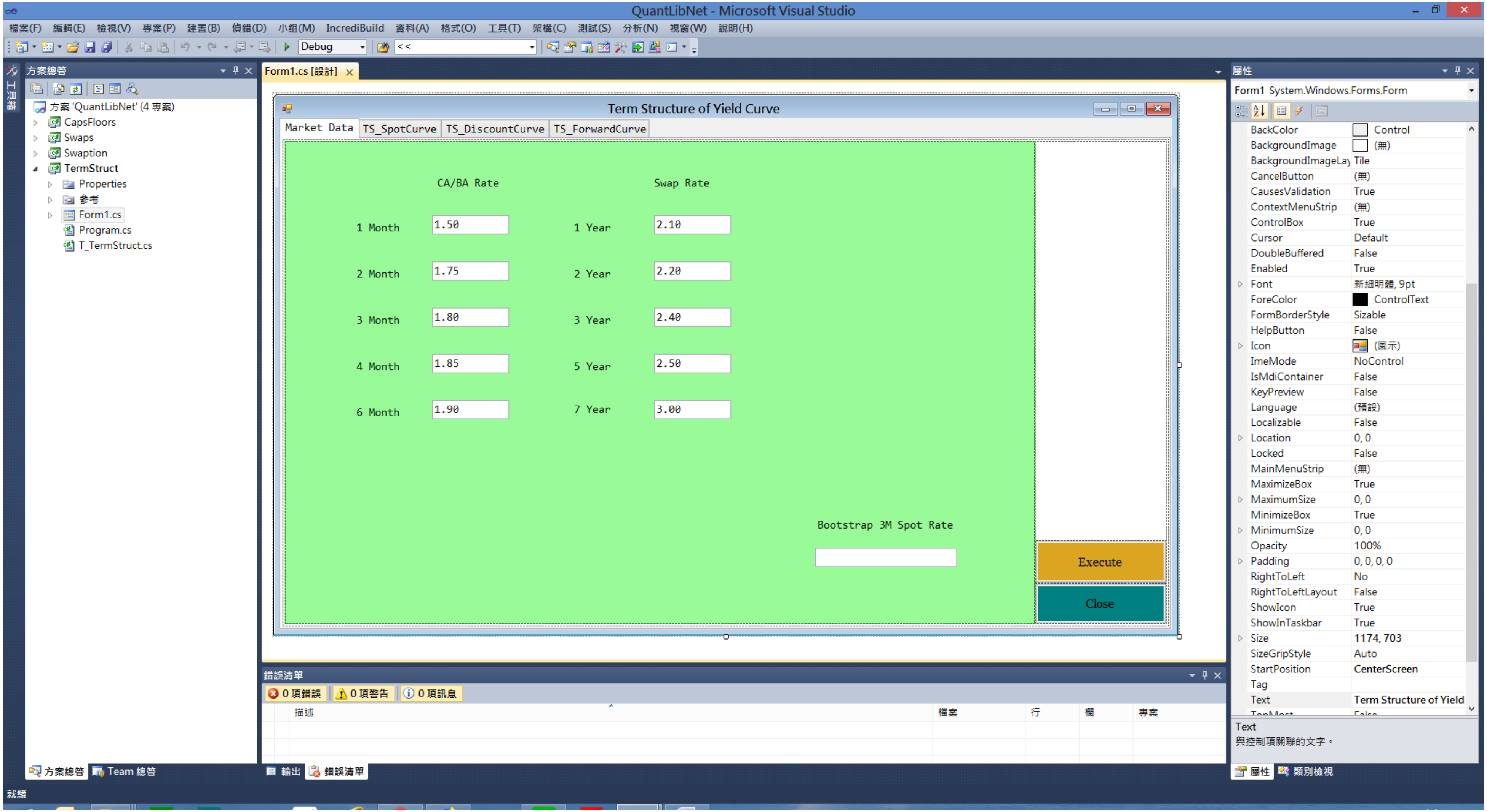
搜尋位置(I): Library

名稱	修改日期	類
QLNet.dll	2019/8/25 下午 06:27	應
QuantLibNet.dll	2015/8/28 上午 11:30	應

檔案名稱(N): QuantLibNet.dll

檔案類型(T): 元件檔 (*.dll;*.tlb;*.olb;*.ocx;*.exe;*.manifest)

➢ Add GUI Widgets



	CA/BA Rate		Swap Rate
1 Month	1.50	1 Year	2.10
2 Month	1.75	2 Year	2.20
3 Month	1.80	3 Year	2.40
4 Month	1.85	5 Year	2.50
6 Month	1.90	7 Year	3.00

Bootstrap 3M Spot Rate

屬性

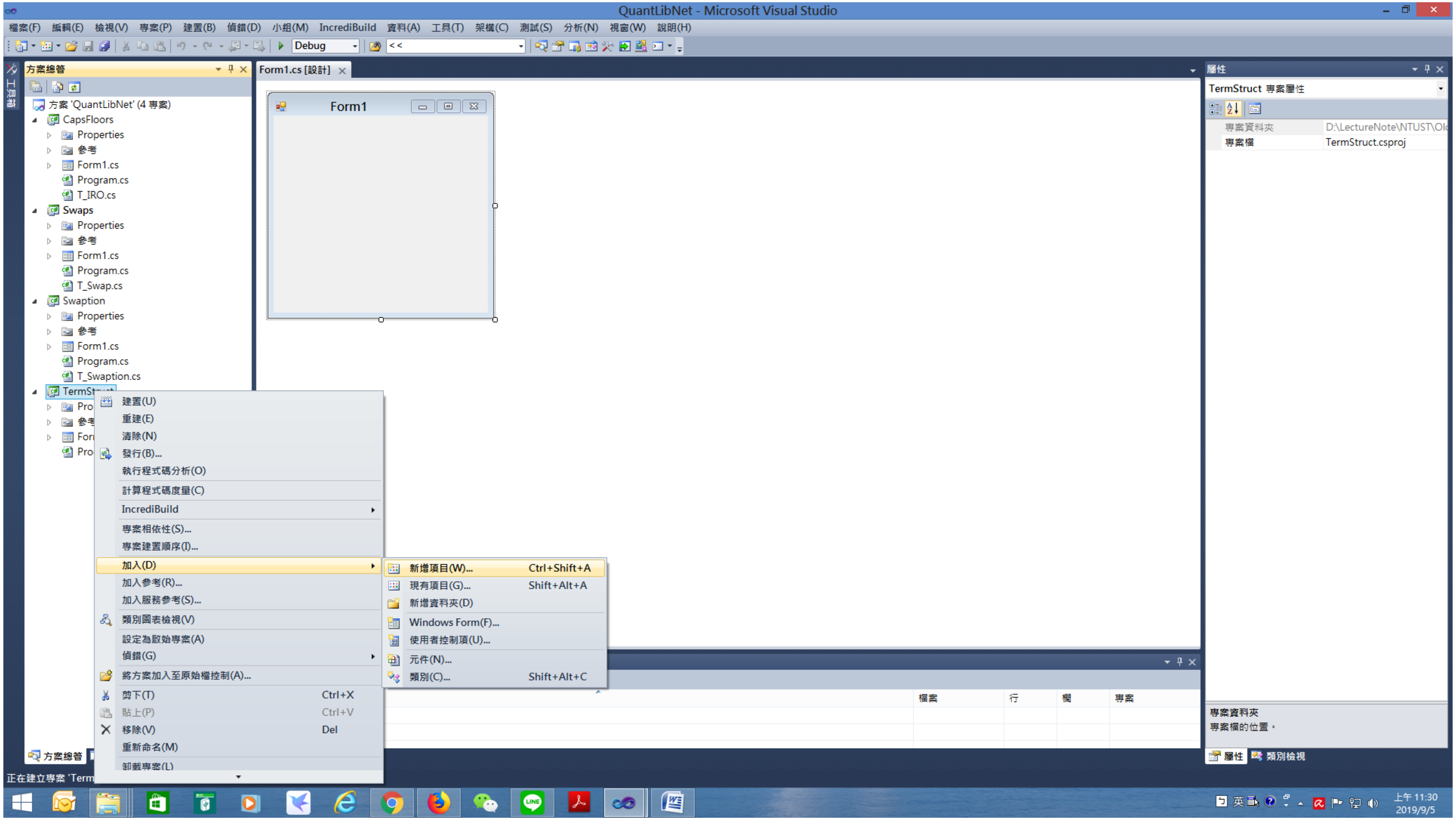
Form1 System.Windows.Forms.Form

BackColor	Control
BackgroundImage	(無)
BackgroundImageLay	Tile
CancelButton	(無)
CausesValidation	True
ContextMenuStrip	(無)
ControlBox	True
Cursor	Default
DoubleBuffered	False
Enabled	True
Font	新細明體, 9pt
ForeColor	ControlText
FormBorderStyle	Sizable
HelpButton	False
Icon	(圖示)
ImeMode	NoControl
IsMdiContainer	False
KeyPreview	False
Language	(預設)
Localizable	False
Location	0, 0
Locked	False
MainMenuStrip	(無)
MaximizeBox	True
MaximumSize	0, 0
MinimizeBox	True
MinimumSize	0, 0
Opacity	100%
Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	1174, 703
SizeGripStyle	Auto
StartPosition	CenterScreen
Tag	
Text	Term Structure of Yield
TopMost	False

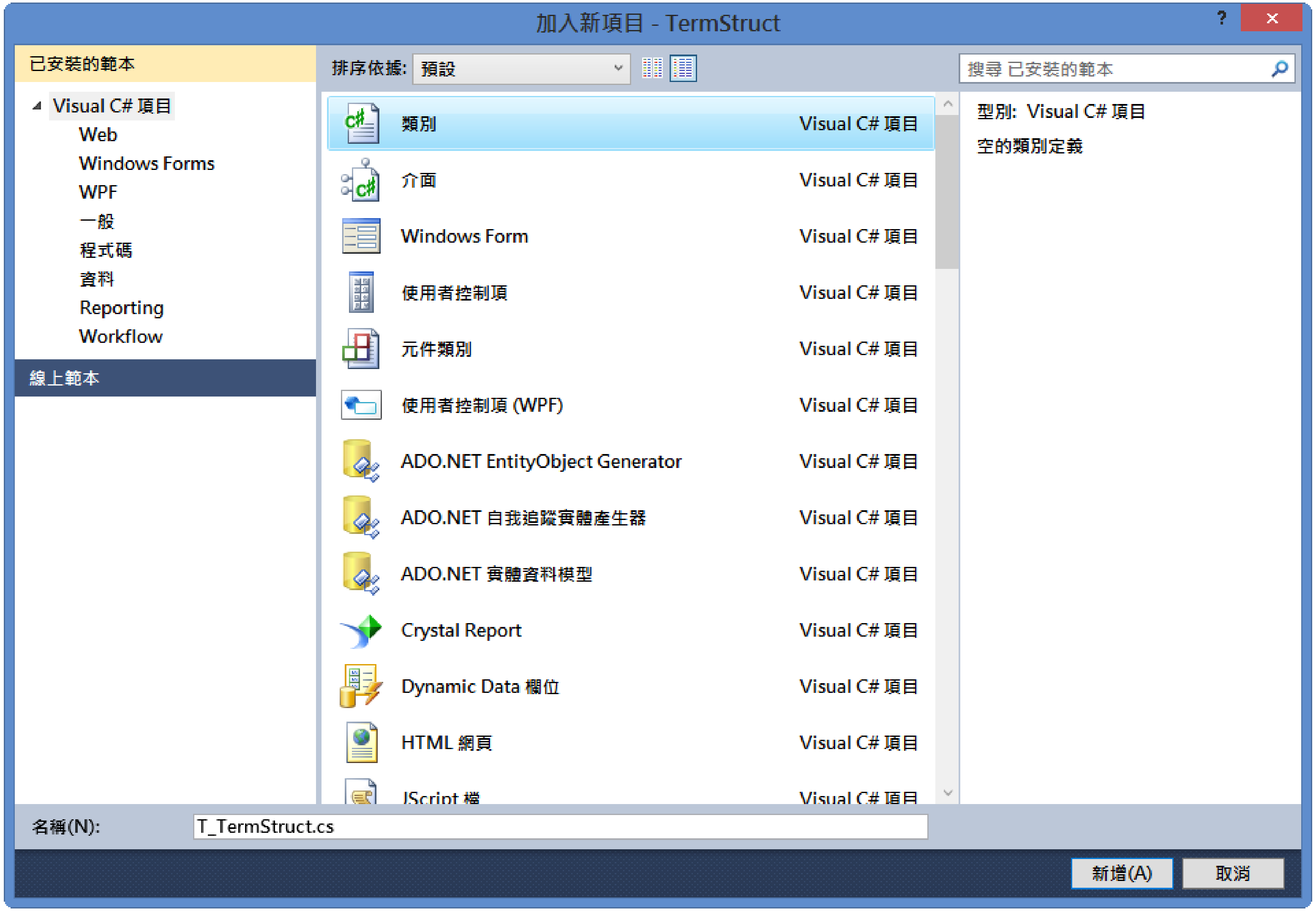
Text

與控制項關聯的文字。

◆ Add New Item


QuantLibNet - Microsoft Visual Studio
檔案(F) 編輯(E) 檢視(V) 專案(P) 建置(B) 偵錯(D) 小組(M) IncrediBuild 資料(A) 工具(T) 架構(C) 測試(S) 分析(N) 視窗(W) 說明(H)
Debug
方案總管
Form1.cs [設計]
Form1
方案 'QuantLibNet' (4 專案)
CapsFloors
Properties
參考
Form1.cs
Program.cs
T_IRO.cs
Swaps
Properties
參考
Form1.cs
Program.cs
T_Swap.cs
Swaption
Properties
參考
Form1.cs
Program.cs
T_Swaption.cs
TermStruct
建置(U)
重建(E)
清除(N)
發行(B)...
執行程式碼分析(O)
計算程式碼度量(C)
IncrediBuild
專案相依性(S)...
專案建置順序(I)...
加入(D)
加入參考(R)...
加入服務參考(S)...
類別圖表檢視(V)
設定為啟始專案(A)
偵錯(G)
將方案加入至原始檔控制(A)...
剪下(T) Ctrl+X
貼上(P) Ctrl+V
移除(V) Del
重新命名(M)
卸載專案(L)
新增項目(W)... Ctrl+Shift+A
現有項目(G)... Shift+Alt+A
新增資料夾(D)
Windows Form(F)...
使用者控制項(U)...
元件(N)...
類別(C)... Shift+Alt+C
屬性
TermStruct 專案屬性
專案資料夾 D:\LectureNote\NTUST\Ol
專案檔 TermStruct.csproj
專案資料夾
專案檔的位置。
檔案 行 欄 專案
方案總管
屬性 類別檢視
正在建立專案 'Term
上午 11:30
2019/9/5

- T_TermStruct.cs



排序依據: 預設

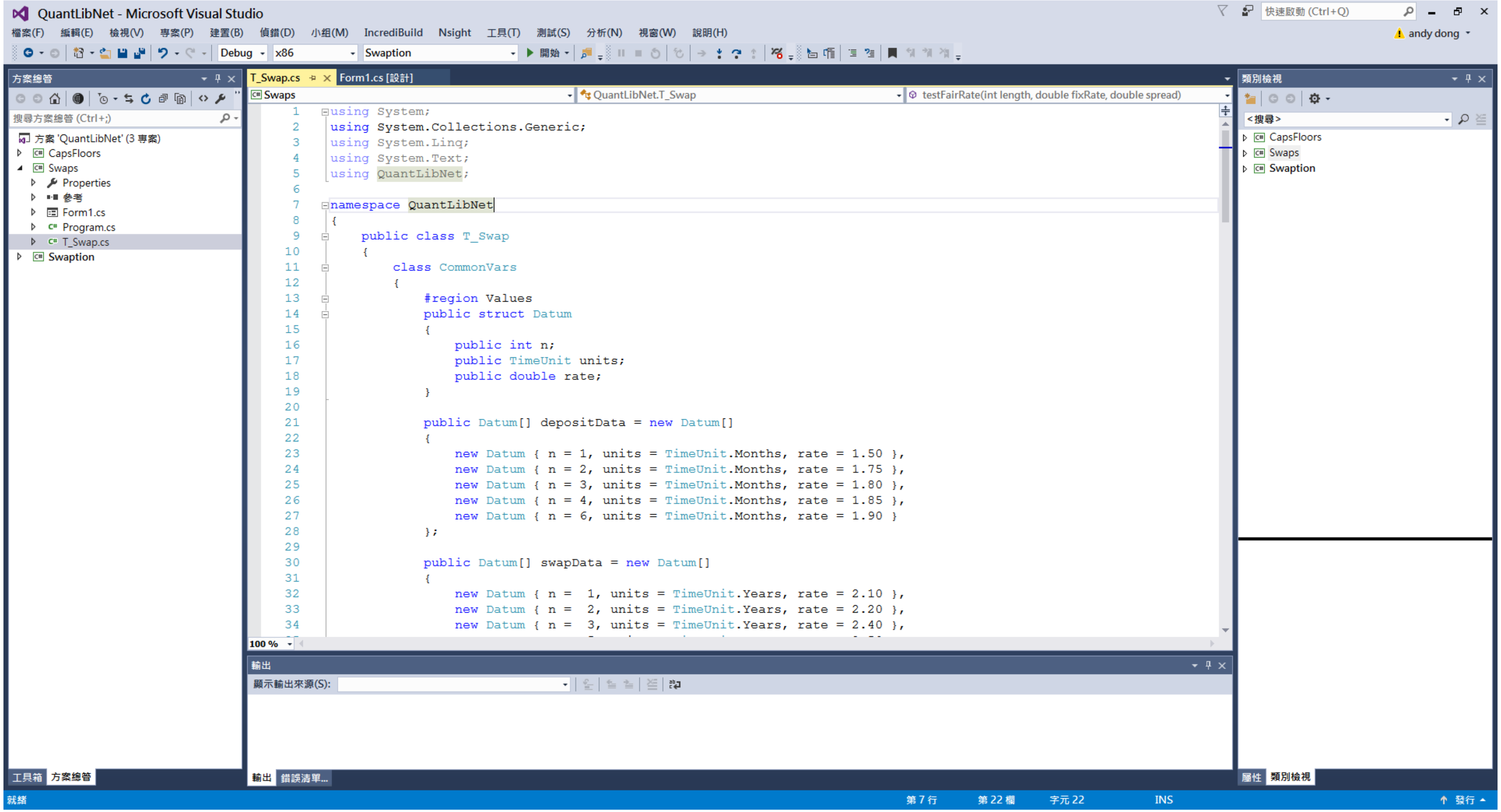
類別	Visual C# 項目
介面	Visual C# 項目
Windows Form	Visual C# 項目
使用者控制項	Visual C# 項目
元件類別	Visual C# 項目
使用者控制項 (WPF)	Visual C# 項目
ADO.NET EntityObject Generator	Visual C# 項目
ADO.NET 自我追蹤實體產生器	Visual C# 項目
ADO.NET 實體資料模型	Visual C# 項目
Crystal Report	Visual C# 項目
Dynamic Data 欄位	Visual C# 項目
HTML 網頁	Visual C# 項目
JScript 檔	Visual C# 項目

型別: Visual C# 項目

空的類別定義

名稱(N): T_TermStruct.cs

➢ Add Code



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using QuantLibNet;

namespace QuantLibNet
{
    public class T_Swap
    {
        class CommonVars
        {
            #region Values
            public struct Datum
            {
                public int n;
                public TimeUnit units;
                public double rate;
            }

            public Datum[] depositData = new Datum[]
            {
                new Datum { n = 1, units = TimeUnit.Months, rate = 1.50 },
                new Datum { n = 2, units = TimeUnit.Months, rate = 1.75 },
                new Datum { n = 3, units = TimeUnit.Months, rate = 1.80 },
                new Datum { n = 4, units = TimeUnit.Months, rate = 1.85 },
                new Datum { n = 6, units = TimeUnit.Months, rate = 1.90 }
            };

            public Datum[] swapData = new Datum[]
            {
                new Datum { n =  1, units = TimeUnit.Years, rate = 2.10 },
                new Datum { n =  2, units = TimeUnit.Years, rate = 2.20 },
                new Datum { n =  3, units = TimeUnit.Years, rate = 2.40 },
```

- T_TermStruct 物件

```
using System;
using System.Collections.Generic;
using System.Text;

using QuantLibNet;

namespace TermStruct
{
    public class CommonVars
    {
        #region Values
        public struct Datum
        {
            public int n;
            public TimeUnit units;
            public double rate;
        }

        public Datum[] depositData;
        public Datum[] swapData;

        #endregion
```

```
// global data
public Date today, settlement;
public Calendar calendar;
public IborIndex index;
public DayCounter fixedDayCount;
public Frequency fixedFrequency, floatingFrequency;
public BusinessDayConvention fixedConvention, floatingConvention;

public YieldTermStructure termstructure;
public RelinkableHandle<YieldTermStructure> RHtermstructure =
    new RelinkableHandle<YieldTermStructure>();

public VanillaSwap.Type type;
public double nominal;
public int settlementDays;


public CommonVars()
{
    depositData[0].n = 1; depositData[0].units = TimeUnit.Months; depositData[0].rate = 1.50;
    depositData[1].n = 2; depositData[1].units = TimeUnit.Months; depositData[1].rate = 1.75;
    depositData[2].n = 3; depositData[2].units = TimeUnit.Months; depositData[2].rate = 1.80;
    depositData[3].n = 4; depositData[3].units = TimeUnit.Months; depositData[3].rate = 1.85;
    depositData[4].n = 6; depositData[4].units = TimeUnit.Months; depositData[4].rate = 1.90;
```

```
swapData[0].n = 1; swapData[0].units = TimeUnit.Years; swapData[0].rate = 2.10;
swapData[2].n = 2; swapData[2].units = TimeUnit.Years; swapData[1].rate = 2.20;
swapData[3].n = 3; swapData[3].units = TimeUnit.Years; swapData[2].rate = 2.40;
swapData[4].n = 4; swapData[4].units = TimeUnit.Years; swapData[3].rate = 2.50;
swapData[5].n = 5; swapData[5].units = TimeUnit.Years; swapData[4].rate = 3.00;

type = VanillaSwap.Type.Payer;
settlementDays = 2;
nominal = 100.0;

fixedConvention = BusinessDayConvention.Unadjusted;
floatingConvention = BusinessDayConvention.Unadjusted;

fixedFrequency = Frequency.Quarterly;
floatingFrequency = Frequency.Quarterly;
fixedDayCount = new Actual365Fixed();

this.index = new Twcpba(new Period(floatingFrequency), RHtermstructure);
calendar = this.index.fixingCalendar();

today = calendar.adjust(Date.Today);
Settings.setEvaluationDate(today);
settlement = calendar.advance(today, settlementDays, TimeUnit.Days);
```

```
// ****************************************************************

int deposits = depositData.Length,  // 5
    swaps = swapData.Length;       // 5

var instruments = new List<BootstrapHelper<YieldTermStructure>>(deposits + swaps);  // 10

IborIndex index = new IborIndex("TWCPBA", new Period(3,
   TimeUnit.Months), settlementDays, new Currency(), calendar,
   BusinessDayConvention.Unadjusted, false, new Actual365Fixed());

for (int i = 0; i < deposits; i++)
{
   instruments.Add(new DepositRateHelper(depositData[i].rate / 100,
      new Period(depositData[i].n, depositData[i].units),
      settlementDays, calendar,
      BusinessDayConvention.ModifiedFollowing,
      true, new Actual365Fixed()));
}
```

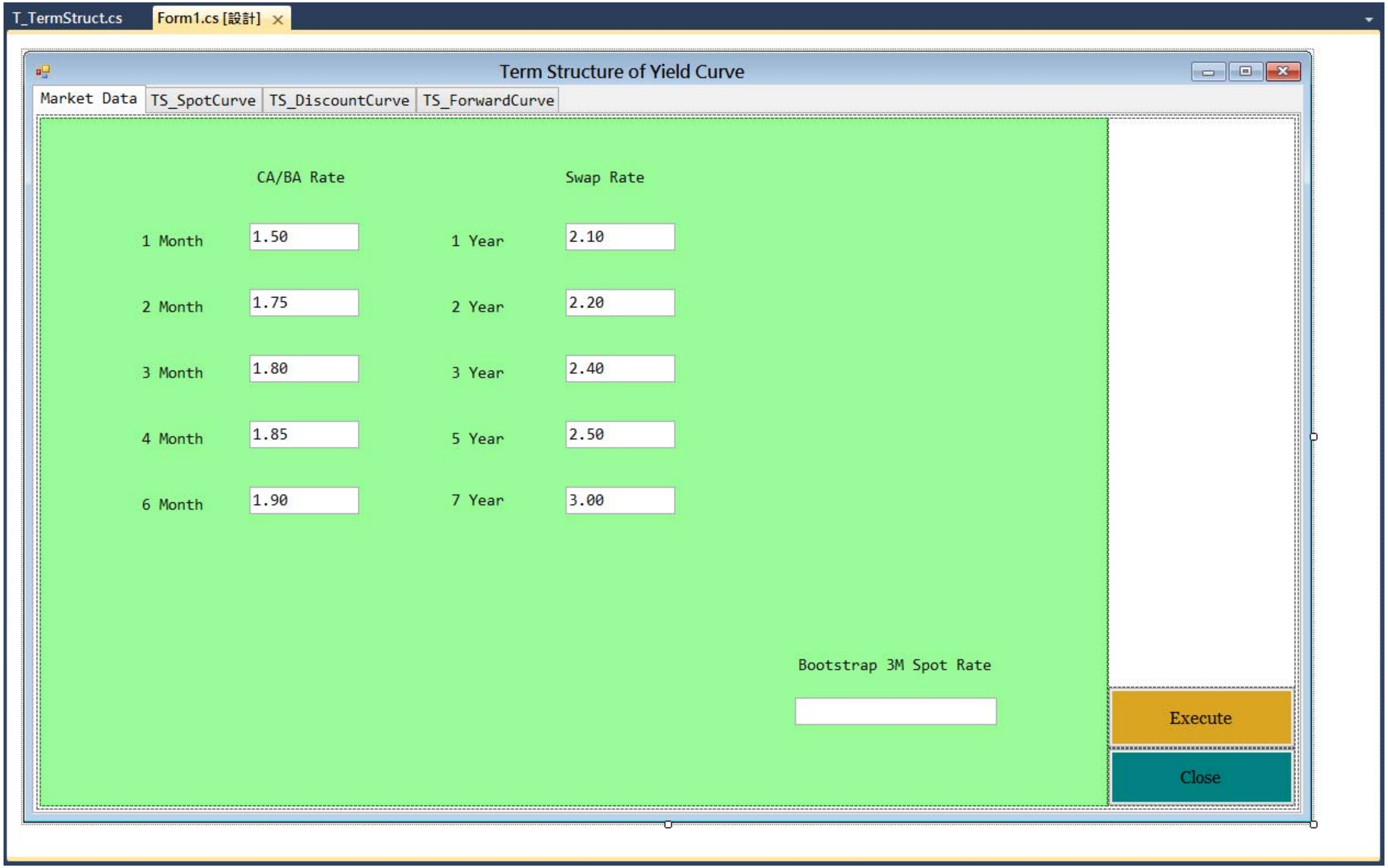
```
    for (int i = 0; i < swaps; ++i)
    {
        instruments.Add(new SwapRateHelper(swapData[i].rate / 100,
            new Period(swapData[i].n, swapData[i].units), calendar,
            Frequency.Quarterly, BusinessDayConvention.Unadjusted,
            new Actual365Fixed(), index));
    }

    termstructure = new PiecewiseYieldCurve<Discount, Linear>(settlement, instruments,
        new Actual365Fixed());

    // ****************************************************************

    RHtermstructure.linkTo(termstructure);
}
```

```
public CommonVars(Datum[] sw, Datum[] cp)
{
    depositData = new Datum[5];
    swapData = new Datum[5];

    for (int i = 0; i < 5; i++)
    {
        depositData[i].n = cp[i].n;
        depositData[i].units = cp[i].units;
        depositData[i].rate = cp[i].rate;

        swapData[i].n = sw[i].n;
        swapData[i].units = sw[i].units;
        swapData[i].rate = sw[i].rate;
    }

    type = VanillaSwap.Type.Payer;
    settlementDays = 2;
    nominal = 100.0;

    fixedConvention = BusinessDayConvention.Unadjusted;
    floatingConvention = BusinessDayConvention.Unadjusted;

    fixedFrequency = Frequency.Quarterly;
    floatingFrequency = Frequency.Quarterly;
    fixedDayCount = new Actual365Fixed();
```

```
this.index = new Twcpba(new Period(floatingFrequency), RHtermstructure);
calendar = this.index.fixingCalendar();

today = calendar.adjust(Date.Today);
Settings.setEvaluationDate(today);
settlement = calendar.advance(today, settlementDays, TimeUnit.Days);

// ****************************************************************

int deposits = depositData.Length,  // 5
    swaps = swapData.Length;       // 5

var instruments = new List<BootstrapHelper<YieldTermStructure>>(deposits + swaps);  // 10

IborIndex index = new IborIndex("TWCPBA", new Period(3,
   TimeUnit.Months), settlementDays, new Currency(), calendar,
   BusinessDayConvention.Unadjusted, false, new Actual365Fixed());

for (int i = 0; i < deposits; i++)
{
   instruments.Add(new DepositRateHelper(depositData[i].rate / 100,
       new Period(depositData[i].n, depositData[i].units),
       settlementDays, calendar,
       BusinessDayConvention.ModifiedFollowing,
       true, new Actual365Fixed()));
}
```

```
                for (int i = 0; i < swaps; ++i)
                {
                    instruments.Add(new SwapRateHelper(swapData[i].rate / 100,
                        new Period(swapData[i].n, swapData[i].units), calendar,
                        Frequency.Quarterly, BusinessDayConvention.Unadjusted,
                        new Actual365Fixed(), index));
                }

                termstructure = new PiecewiseYieldCurve<Discount, Linear>
                    (settlement, instruments, new Actual365Fixed());

                // ****************************************************************

                RHtermstructure.linkTo(termstructure);
            }
        }

        public class T_TermStruct{ }
}
```

◆ Main Form



	CA/BA Rate		Swap Rate
1 Month	1.50	1 Year	2.10
2 Month	1.75	2 Year	2.20
3 Month	1.80	3 Year	2.40
4 Month	1.85	5 Year	2.50
6 Month	1.90	7 Year	3.00

Bootstrap 3M Spot Rate

- Double Click Close Button，Add Code。

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Text;
using System.Windows.Forms;

using QuantLibNet;
using System.Windows.Forms.DataVisualization.Charting;

namespace TermStruct
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         Application.Exit();
      }
}
```

- Double Click Execute Button 。

```
public CommonVars Comm;
public YieldTermStructure TS;
public Calendar cal;
public DayCounter dc;

private void button2_Click(object sender, EventArgs e)
{
    CommonVars.Datum[] CPData = new CommonVars.Datum[5];
    CPData[0].n = 1; CPData[0].units = TimeUnit.Months; CPData[0].rate = Convert.ToDouble(textBox1.Text);
    CPData[1].n = 2; CPData[1].units = TimeUnit.Months; CPData[1].rate = Convert.ToDouble(textBox2.Text);
    CPData[2].n = 3; CPData[2].units = TimeUnit.Months; CPData[2].rate = Convert.ToDouble(textBox3.Text);
    CPData[3].n = 4; CPData[3].units = TimeUnit.Months; CPData[3].rate = Convert.ToDouble(textBox4.Text);
    CPData[4].n = 6; CPData[4].units = TimeUnit.Months; CPData[4].rate = Convert.ToDouble(textBox5.Text);

    CommonVars.Datum[] SWData = new CommonVars.Datum[5];
    SWData[0].n = 1; SWData[0].units = TimeUnit.Years; SWData[0].rate = Convert.ToDouble(textBox6.Text);
    SWData[1].n = 2; SWData[1].units = TimeUnit.Years; SWData[1].rate = Convert.ToDouble(textBox7.Text);
    SWData[2].n = 3; SWData[2].units = TimeUnit.Years; SWData[2].rate = Convert.ToDouble(textBox8.Text);
    SWData[3].n = 5; SWData[3].units = TimeUnit.Years; SWData[3].rate = Convert.ToDouble(textBox9.Text);
    SWData[4].n = 7; SWData[4].units = TimeUnit.Years; SWData[4].rate = Convert.ToDouble(textBox10.Text);
```

```
Comm = new CommonVars(CPData, SWData);
TS = Comm.termstructure;

cal = new Taiwan();
dc = new Actual365Fixed();

Date basedate = cal.advance(Date.Today, 2, TimeUnit.Days);
Date date3M = cal.advance(basedate, new Period(3, TimeUnit.Months));

textBox11.Text = TS.zeroRate(date3M, dc, Compounding.Simple).value().ToString("F6");
```

```
// ************************************** Yield Curve ***********************************

string[] seriesArray1 = { "SpotRate" };
double[] points1 = new double[29]; // 7 * 4 + 1

for (int i = 0; i < 29; i++)
{
   Date nextdate = cal.advance(basedate, new Period(3 * i, TimeUnit.Months));
   points1[i] = TS.zeroRate(nextdate, dc, Compounding.Compounded).value();
}

// Set title.
this.chart1.Titles.Clear();
this.chart1.Titles.Add("Yield Curve");

// Add series.
Series series1 = new Series();
this.chart1.Series.Clear();

for (int i = 0; i < seriesArray1.Length; i++)
{
   // Add series.
   series1 = this.chart1.Series.Add(seriesArray1[i]);
   series1.ChartType = SeriesChartType.Line;
   series1.BorderWidth = 2;
```

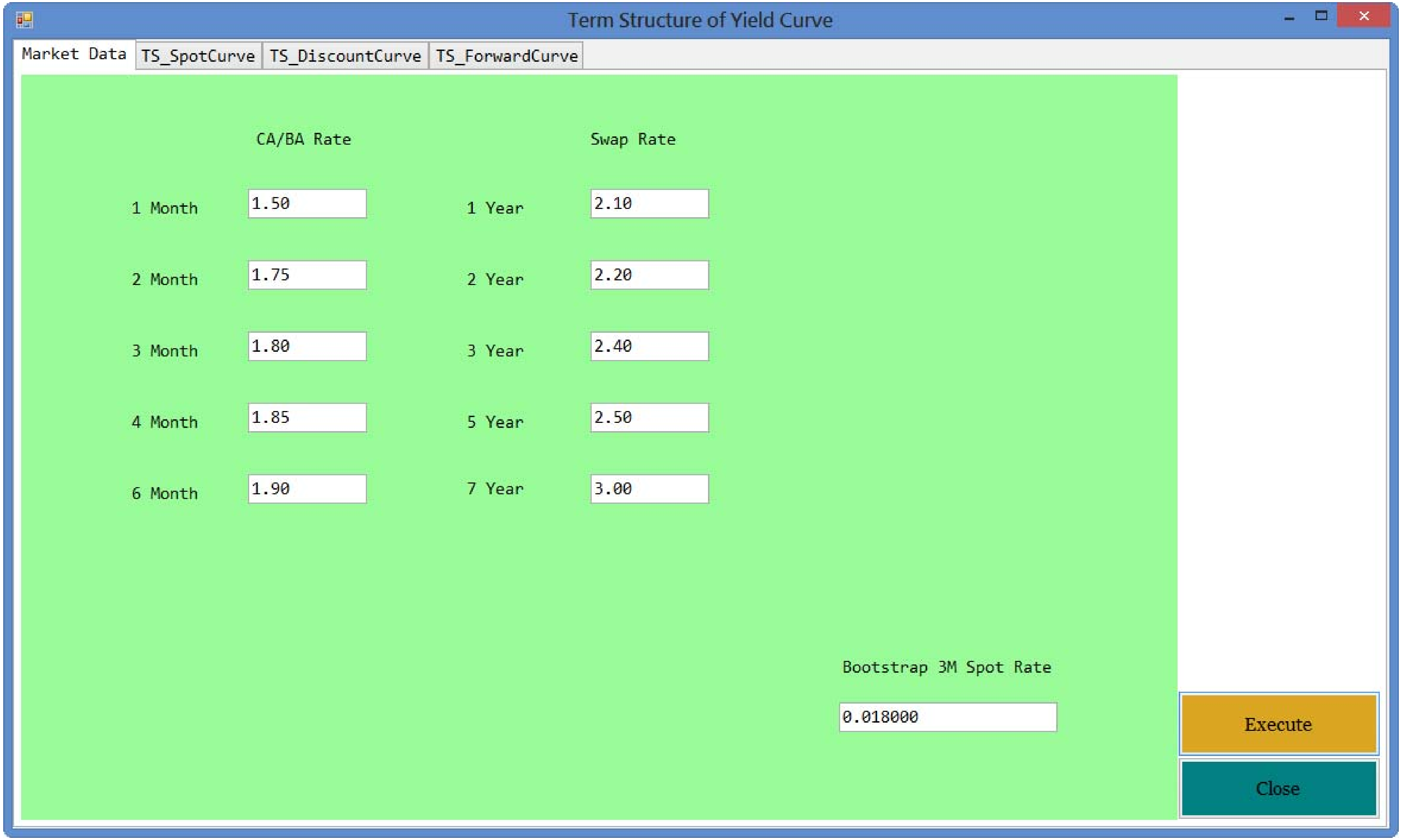
```
    // Add point.
    for (int j = 0; j < 29; j++)
    {
        series1.Points.AddXY(j, points1[j]);
    }
}

textBox12.Text = basedate.ToShortDateString();
```

```
// *********************************** Discount Function *********************************

string[] seriesArray2 = { "DiscountFunction" };
double[] points2 = new double[29]; // 7 * 4 + 1

for (int i = 0; i < 29; i++)
{
    Date nextdate = cal.advance(basedate, new Period(3 * i, TimeUnit.Months));
    points2[i] = TS.discount(nextdate, true);
}

// Set title.
this.chart2.Titles.Clear();
this.chart2.Titles.Add("Discount Curve");

// Add series.
Series series2 = new Series();
this.chart2.Series.Clear();

for (int i = 0; i < seriesArray2.Length; i++)
{
    // Add series.
    series2 = this.chart2.Series.Add(seriesArray2[i]);
    series2.ChartType = SeriesChartType.Line;
    series2.BorderWidth = 2;
    // Add point.
    for (int j = 0; j < 29; j++)
    {
        series2.Points.AddXY(j, points2[j]);
    }
}

textBox17.Text = basedate.ToShortDateString();
```

```
// *********************************** Forward Rate *********************************

string[] seriesArray3 = { "3MForwardRate" };
double[] points3 = new double[29]; // 7 * 4 + 1

for (int i = 0; i < 28; i++)
{
   Date firstdate = cal.advance(basedate, new Period(3 * i, TimeUnit.Months));
   Date seconddate = cal.advance(basedate, new Period(3 * (i+1), TimeUnit.Months));
   points3[i] = TS.forwardRate(firstdate, seconddate, dc , Compounding.Simple).value();
}

Date terminaldate = cal.advance(basedate, new Period(3 * 28, TimeUnit.Months));
Date priordate = terminaldate - 1;
points3[28] = TS.forwardRate(priordate, terminaldate, dc, Compounding.Simple).value();

// Set title.
this.chart3.Titles.Clear();
this.chart3.Titles.Add("3 Month Forward Rate");
```

```
    // Add series.
    Series series3 = new Series();
    this.chart3.Series.Clear();

    for (int i = 0; i < seriesArray3.Length; i++)
    {
        // Add series.
        series3 = this.chart3.Series.Add(seriesArray3[i]);
        series3.ChartType = SeriesChartType.Line;
        series3.BorderWidth = 2;

        // Add point.
        for (int j = 0; j < 29; j++)
        {
            series3.Points.AddXY(j, points3[j]);
        }
    }

    textBox20.Text = basedate.ToShortDateString();
}
```

➢ Execute

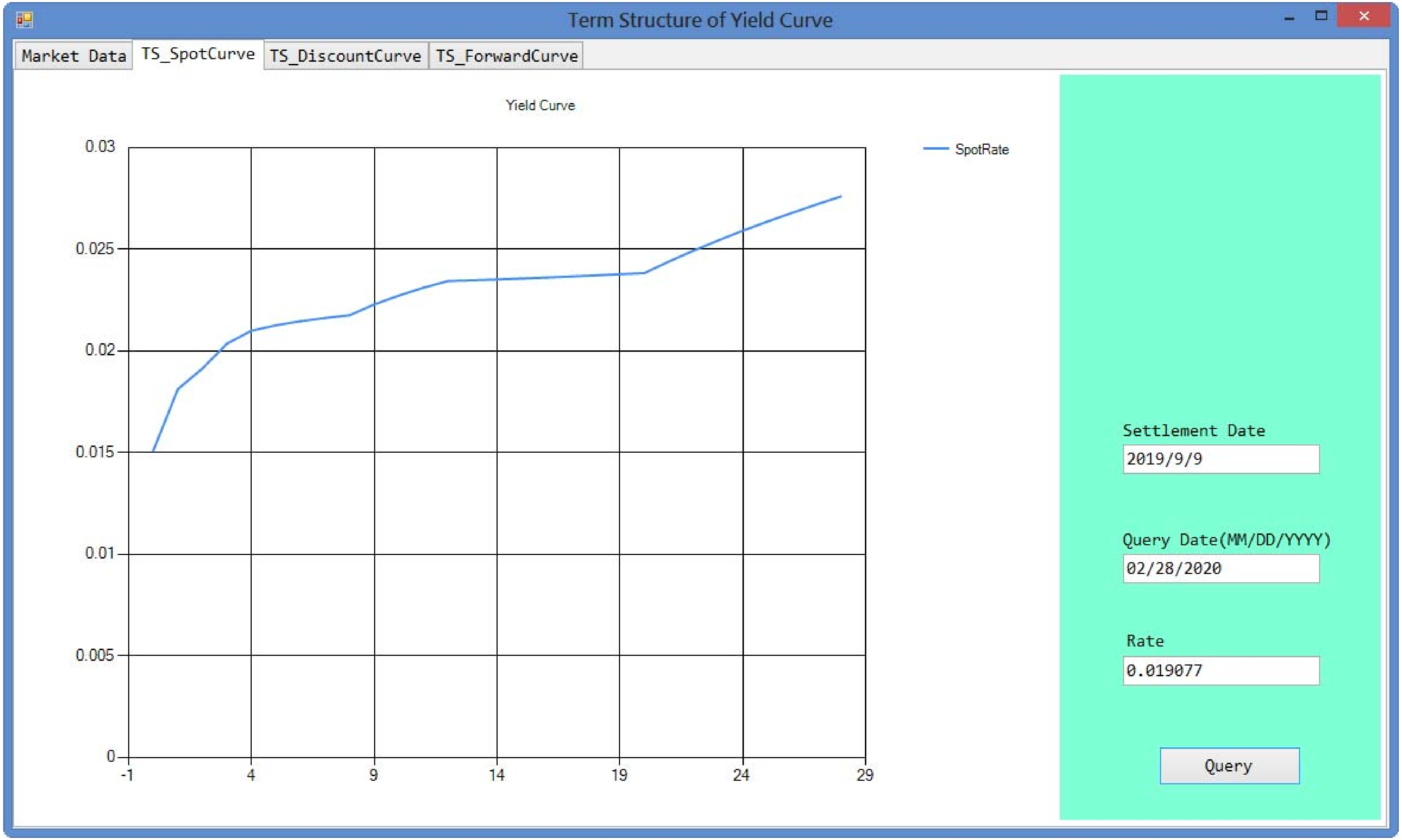


Term Structure of Yield Curve

CA/BA Rate		Swap Rate	
1 Month	1.50	1 Year	2.10
2 Month	1.75	2 Year	2.20
3 Month	1.80	3 Year	2.40
4 Month	1.85	5 Year	2.50
6 Month	1.90	7 Year	3.00

Bootstrap 3M Spot Rate

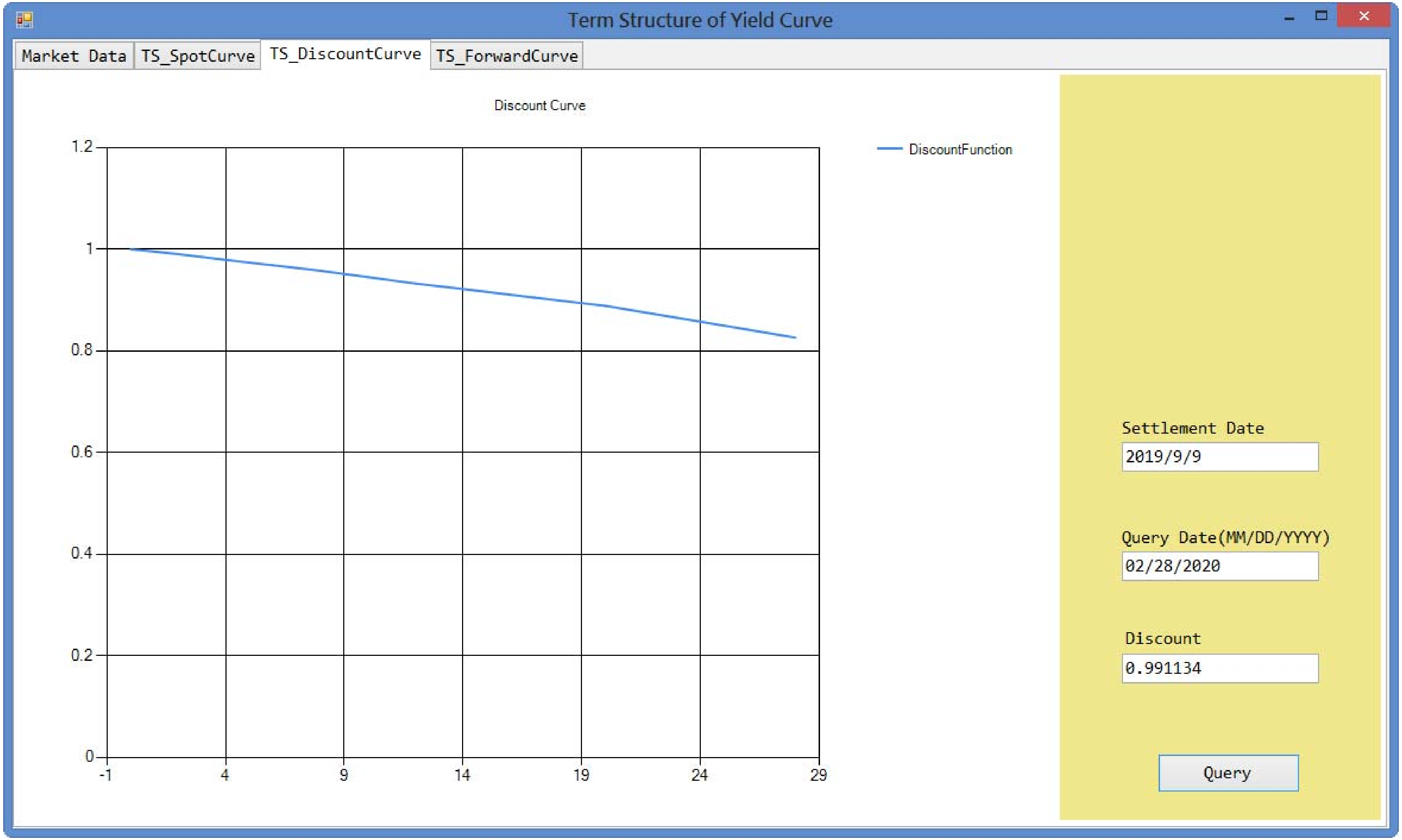
0.018000



Settlement Date
2019/9/9

Query Date(MM/DD/YYYY)
02/28/2020

Rate
0.019077



Settlement Date

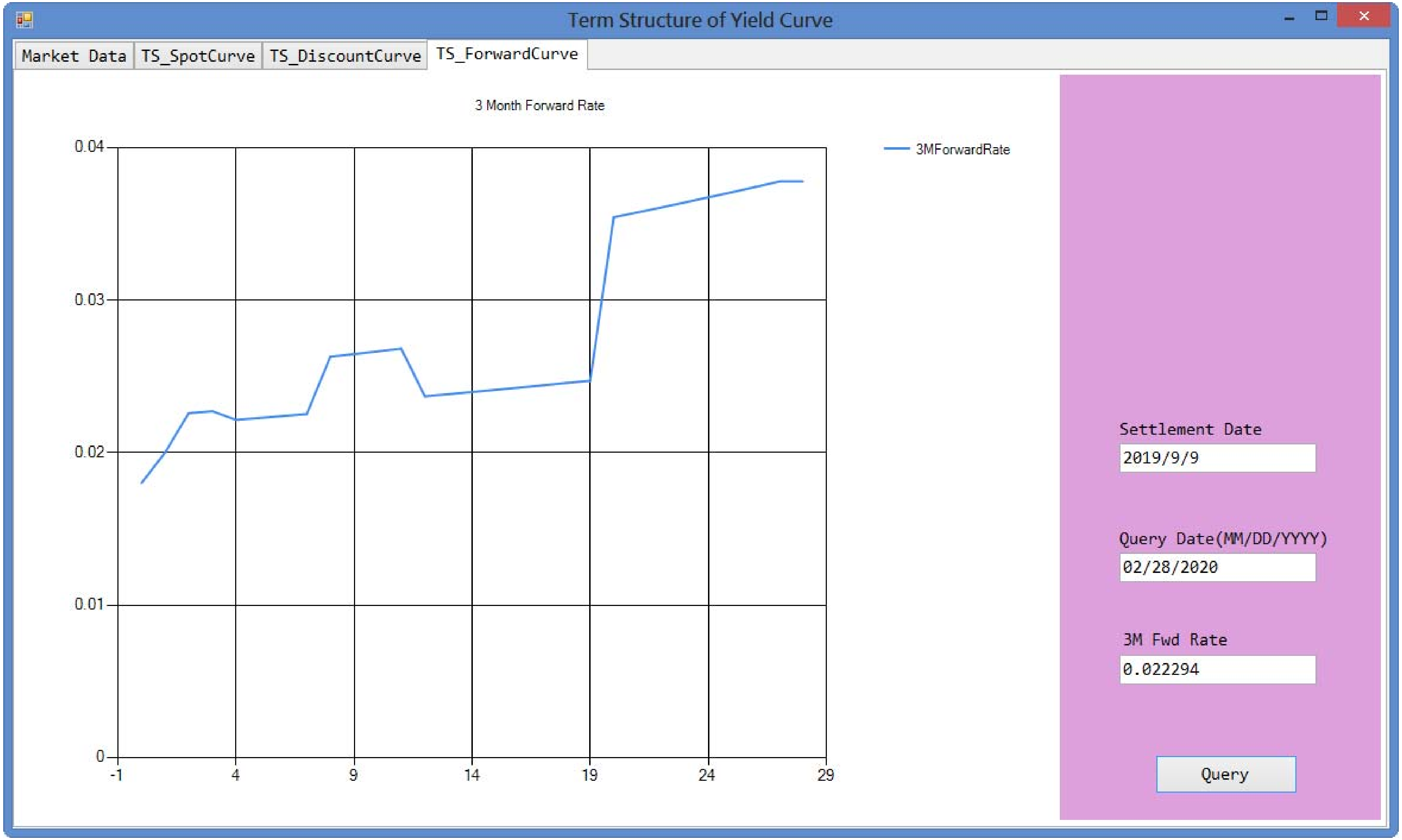
2019/9/9

Query Date(MM/DD/YYYY)

02/28/2020

Discount

0.991134



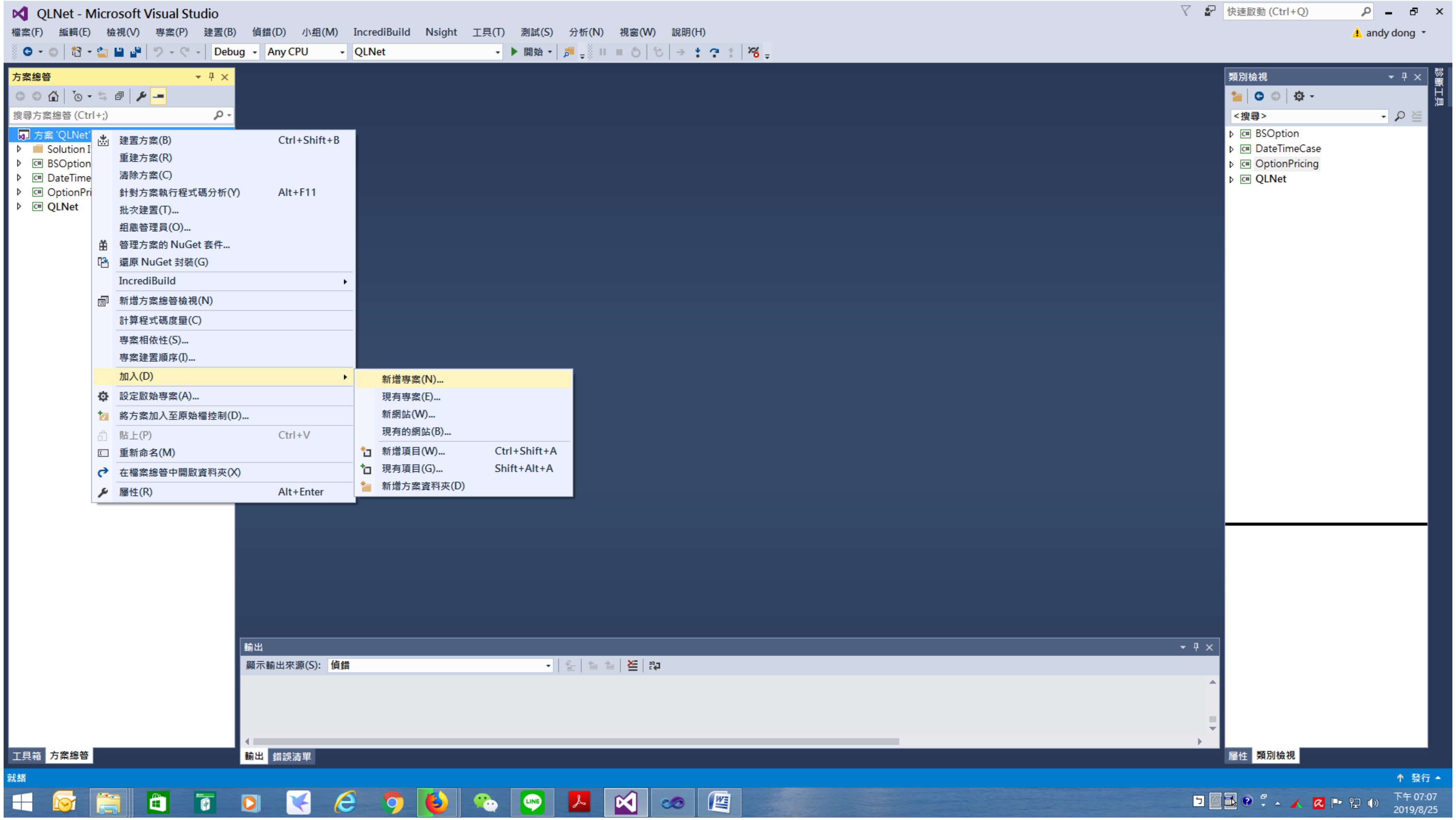
Settlement Date
2019/9/9

Query Date(MM/DD/YYYY)
02/28/2020

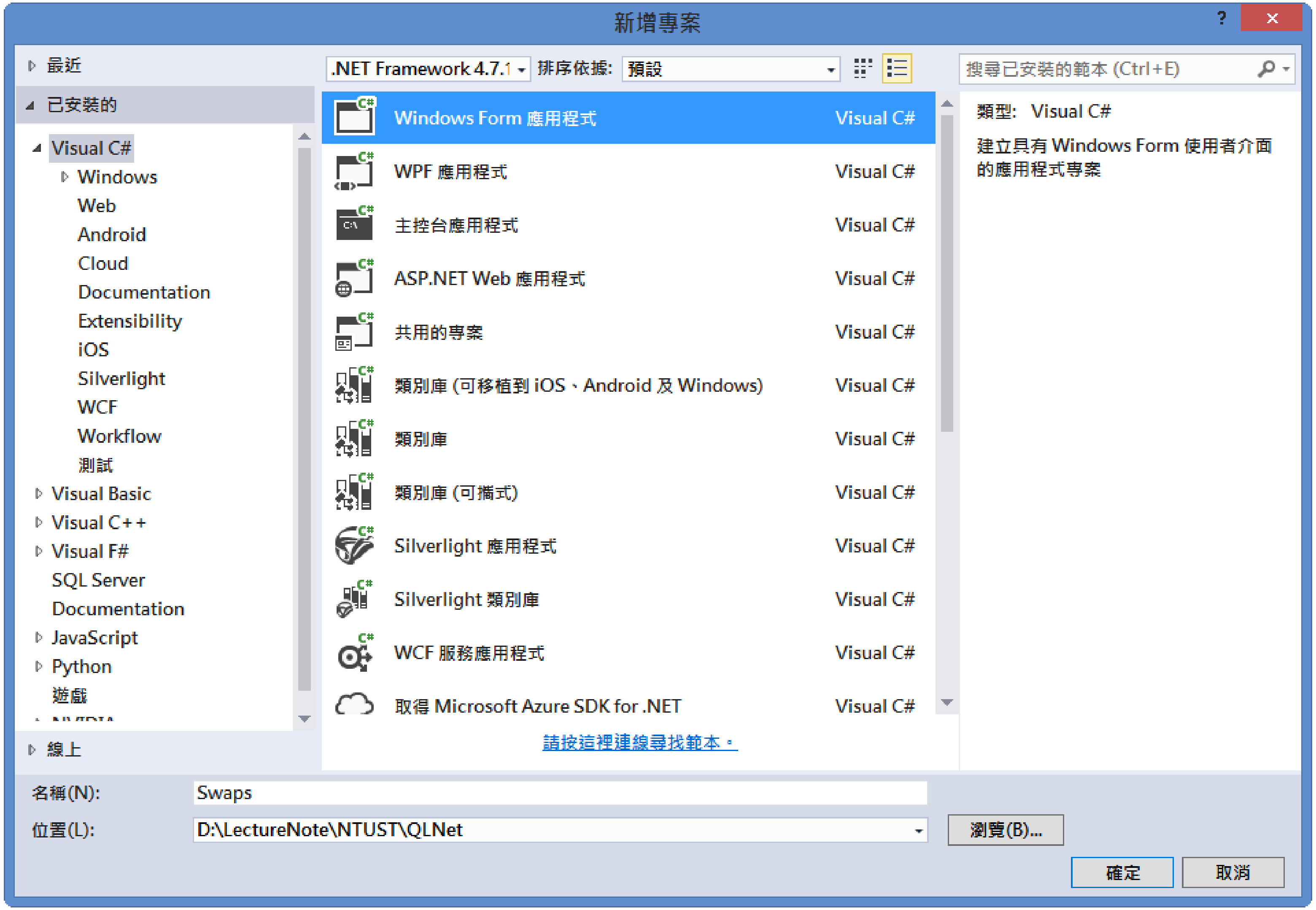
3M Fwd Rate
0.022294

3.2 Swaps Calculation

◆ Add New Project



◆ Name：Swaps，Windows Form Application。



新增專案

▷ 最近

◢ 已安裝的

◢ Visual C#
- ▷ Windows
- Web
- Android
- Cloud
- Documentation
- Extensibility
- iOS
- Silverlight
- WCF
- Workflow
- 測試

▷ Visual Basic

▷ Visual C++

▷ Visual F#

SQL Server

Documentation

▷ JavaScript

▷ Python

遊戲

▷ 線上

.NET Framework 4.7.1　排序依據: 預設

Windows Form 應用程式	Visual C#
WPF 應用程式	Visual C#
主控台應用程式	Visual C#
ASP.NET Web 應用程式	Visual C#
共用的專案	Visual C#
類別庫 (可移植到 iOS、Android 及 Windows)	Visual C#
類別庫	Visual C#
類別庫 (可攜式)	Visual C#
Silverlight 應用程式	Visual C#
Silverlight 類別庫	Visual C#
WCF 服務應用程式	Visual C#
取得 Microsoft Azure SDK for .NET	Visual C#

請按這裡連線尋找範本。

搜尋已安裝的範本 (Ctrl+E)

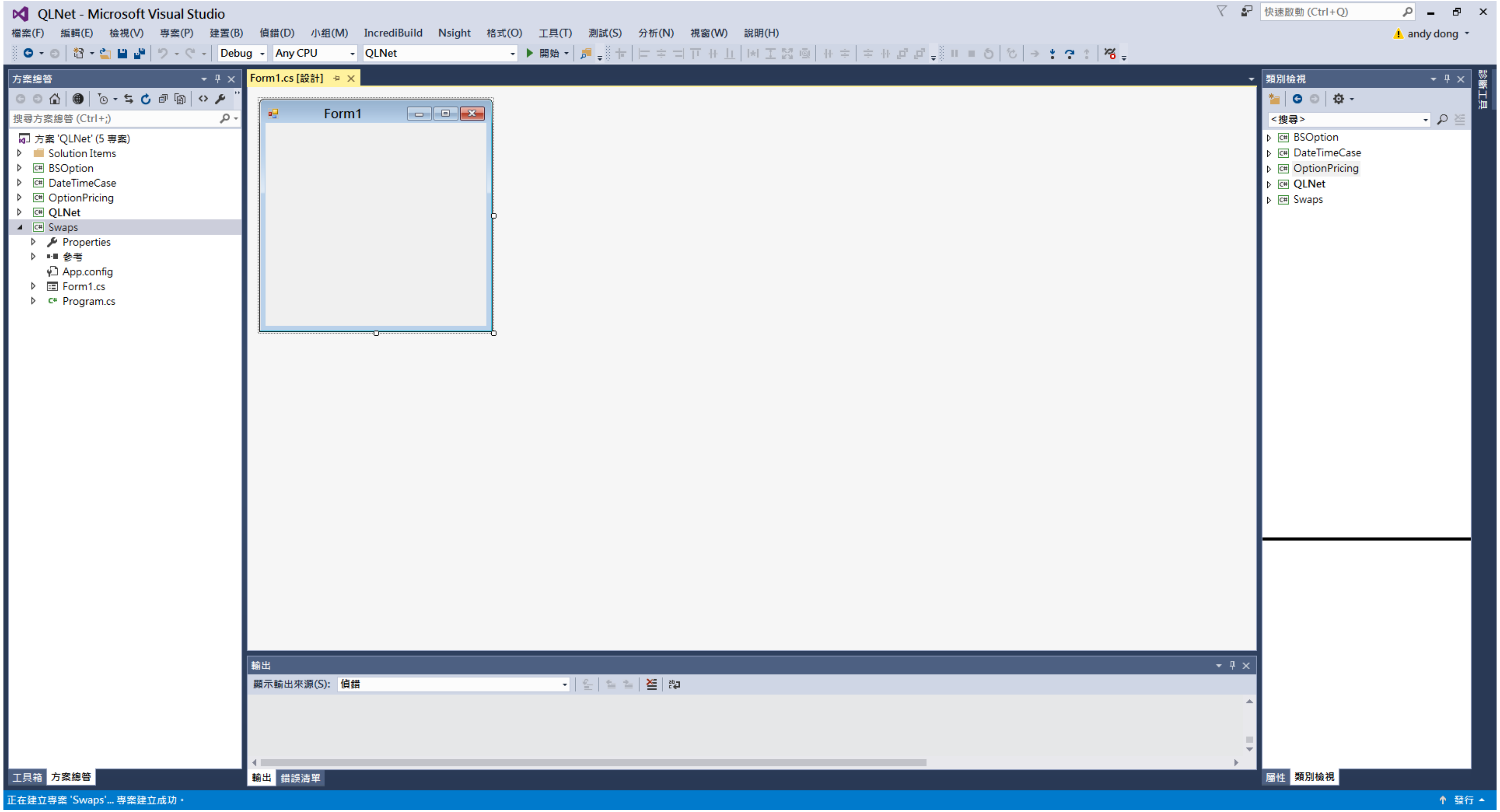
類型:　Visual C#

建立具有 Windows Form 使用者介面的應用程式專案

名稱(N): Swaps

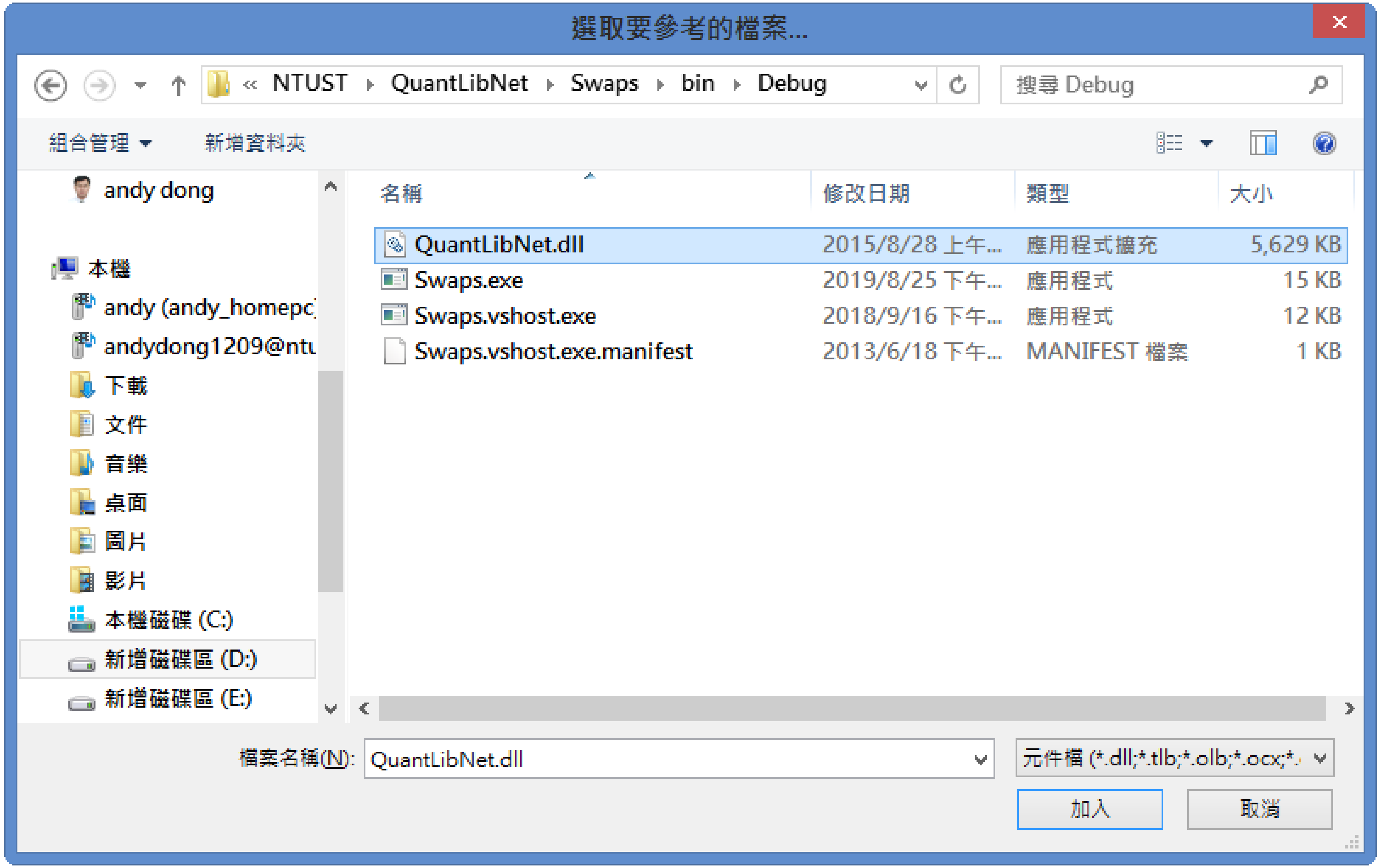
位置(L): D:\LectureNote\NTUST\QLNet

◆ Create New Form



正在建立專案 'Swaps'... 專案建立成功。

- Add Reference QuantLibNet.dll。



選取要參考的檔案...

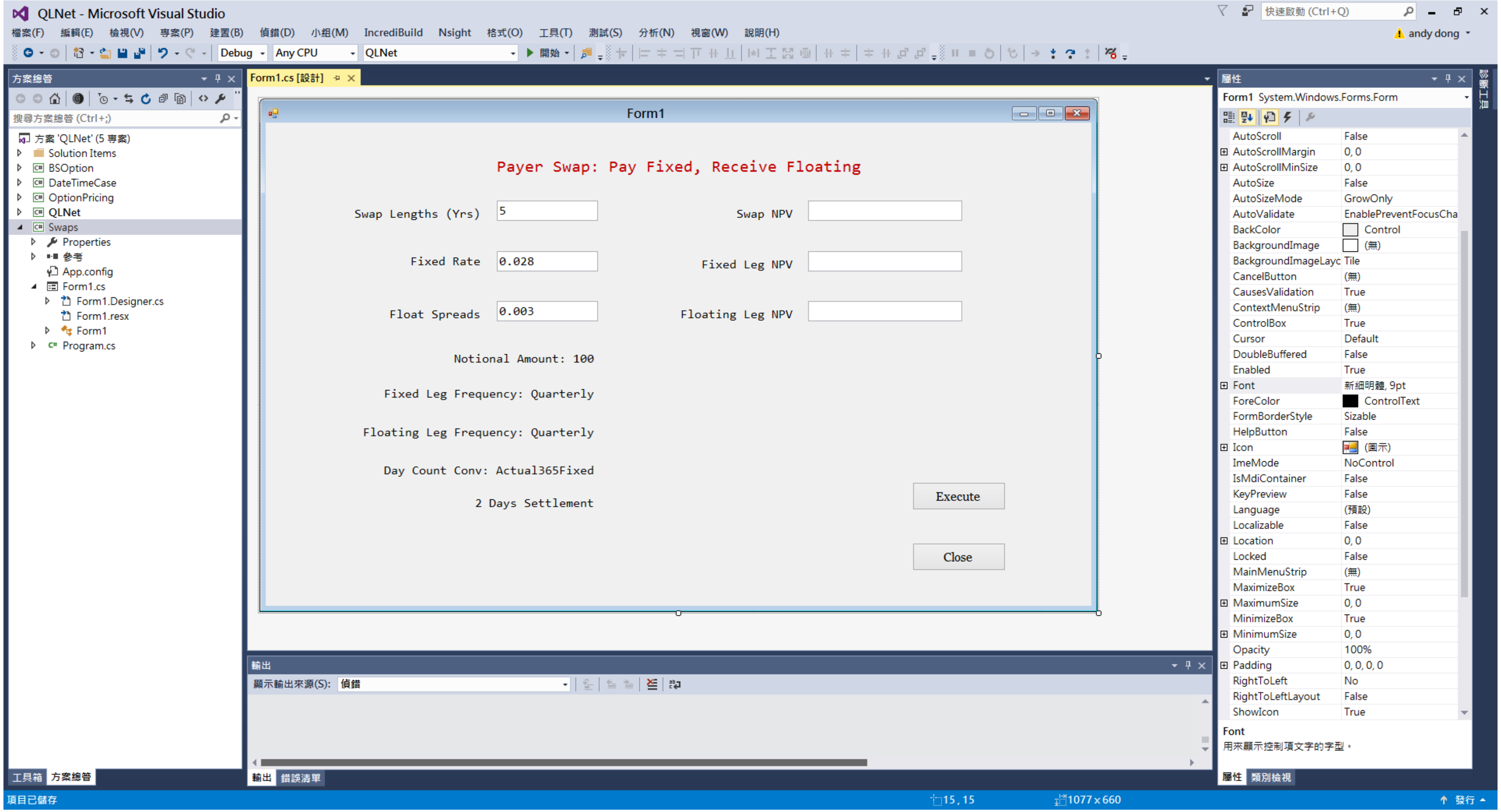
« NTUST ▸ QuantLibNet ▸ Swaps ▸ bin ▸ Debug

名稱	修改日期	類型	大小
QuantLibNet.dll	2015/8/28 上午...	應用程式擴充	5,629 KB
Swaps.exe	2019/8/25 下午...	應用程式	15 KB
Swaps.vshost.exe	2018/9/16 下午...	應用程式	12 KB
Swaps.vshost.exe.manifest	2013/6/18 下午...	MANIFEST 檔案	1 KB

檔案名稱(N): QuantLibNet.dll

元件檔 (*.dll;*.tlb;*.olb;*.ocx;*.

➢ Add GUI Widgets



Form1

Payer Swap: Pay Fixed, Receive Floating

Swap Lengths (Yrs)	5	Swap NPV	
Fixed Rate	0.028	Fixed Leg NPV	
Float Spreads	0.003	Floating Leg NPV	

Notional Amount: 100

Fixed Leg Frequency: Quarterly

Floating Leg Frequency: Quarterly

Day Count Conv: Actual365Fixed

2 Days Settlement

Execute

Close

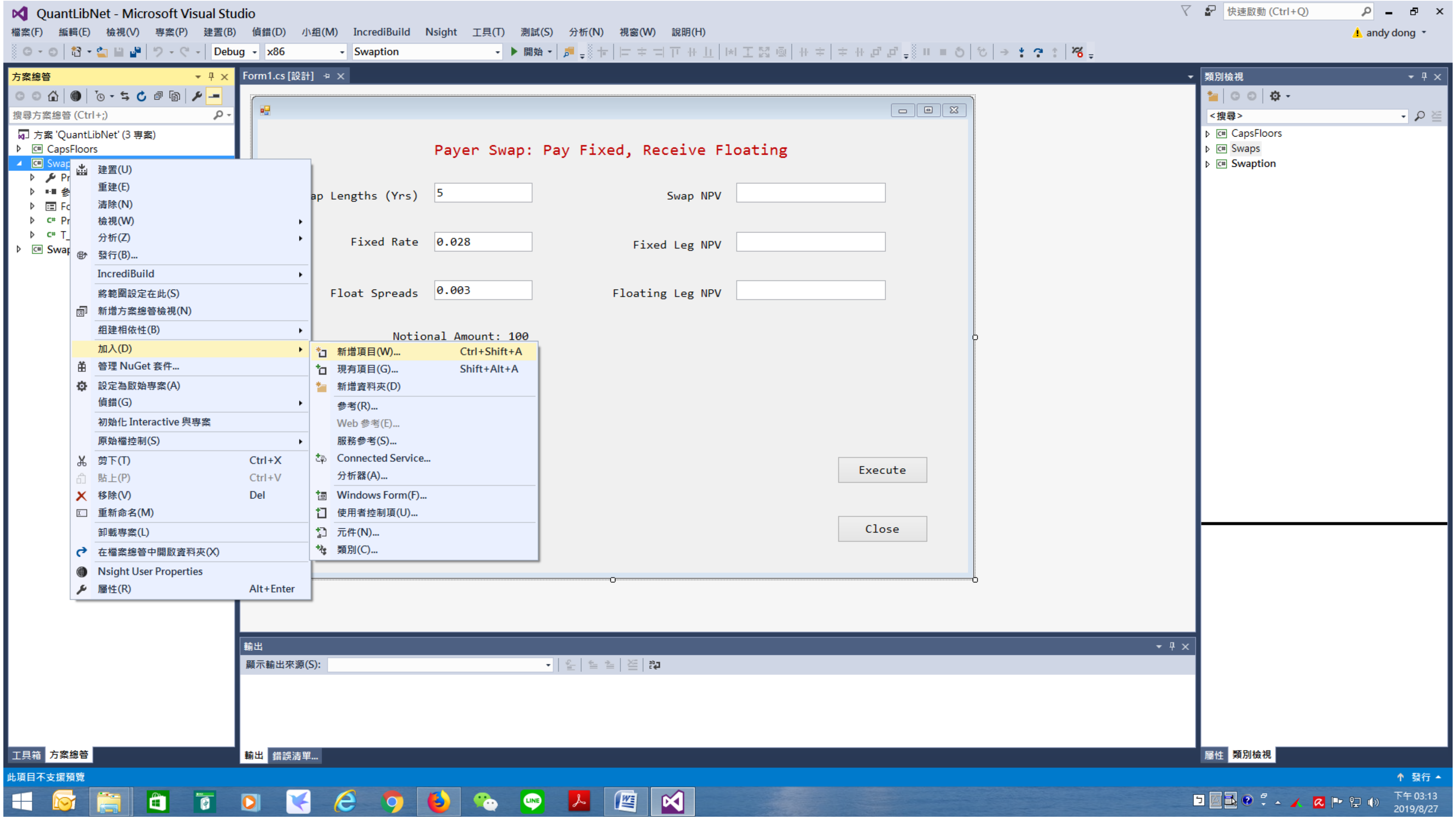
屬性

Form1 System.Windows.Forms.Form

AutoScroll	False
⊞ AutoScrollMargin	0, 0
⊞ AutoScrollMinSize	0, 0
AutoSize	False
AutoSizeMode	GrowOnly
AutoValidate	EnablePreventFocusCha
BackColor	□ Control
BackgroundImage	□ (無)
BackgroundImageLayc	Tile
CancelButton	(無)
CausesValidation	True
ContextMenuStrip	(無)
ControlBox	True
Cursor	Default
DoubleBuffered	False
Enabled	True
⊞ Font	新細明體, 9pt
ForeColor	■ ControlText
FormBorderStyle	Sizable
HelpButton	False
⊞ Icon	(圖示)
ImeMode	NoControl
IsMdiContainer	False
KeyPreview	False
Language	(預設)
Localizable	False
⊞ Location	0, 0
Locked	False
MainMenuStrip	(無)
MaximizeBox	True
⊞ MaximumSize	0, 0
MinimizeBox	True
⊞ MinimumSize	0, 0
Opacity	100%
⊞ Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True

Font
用來顯示控制項文字的字型。

◆ Add New Item



Payer Swap: Pay Fixed, Receive Floating

ap Lengths (Yrs) 5

Fixed Rate 0.028

Float Spreads 0.003

Swap NPV

Fixed Leg NPV

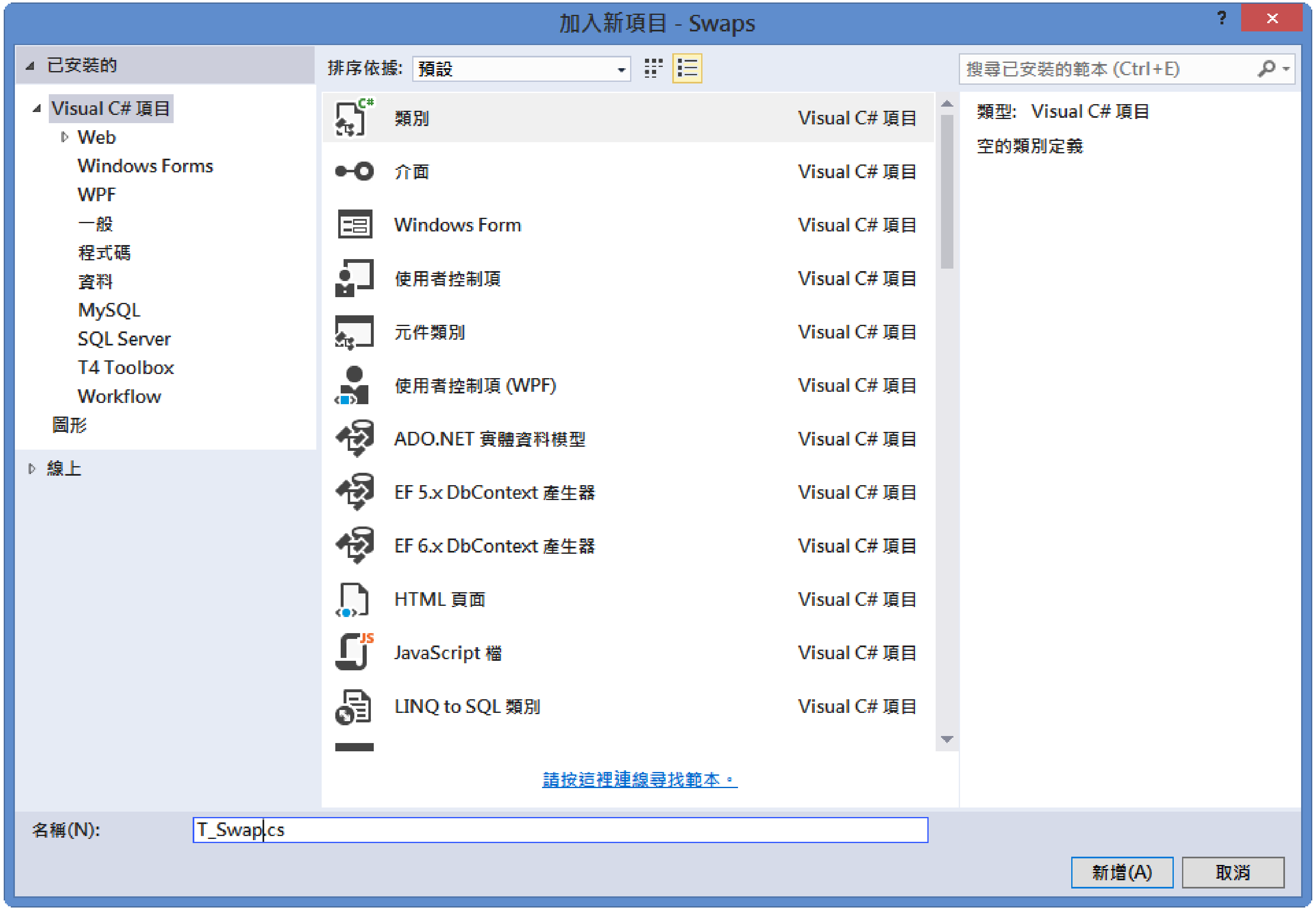
Floating Leg NPV

Notional Amount: 100

Execute

Close

- T_Swap.cs



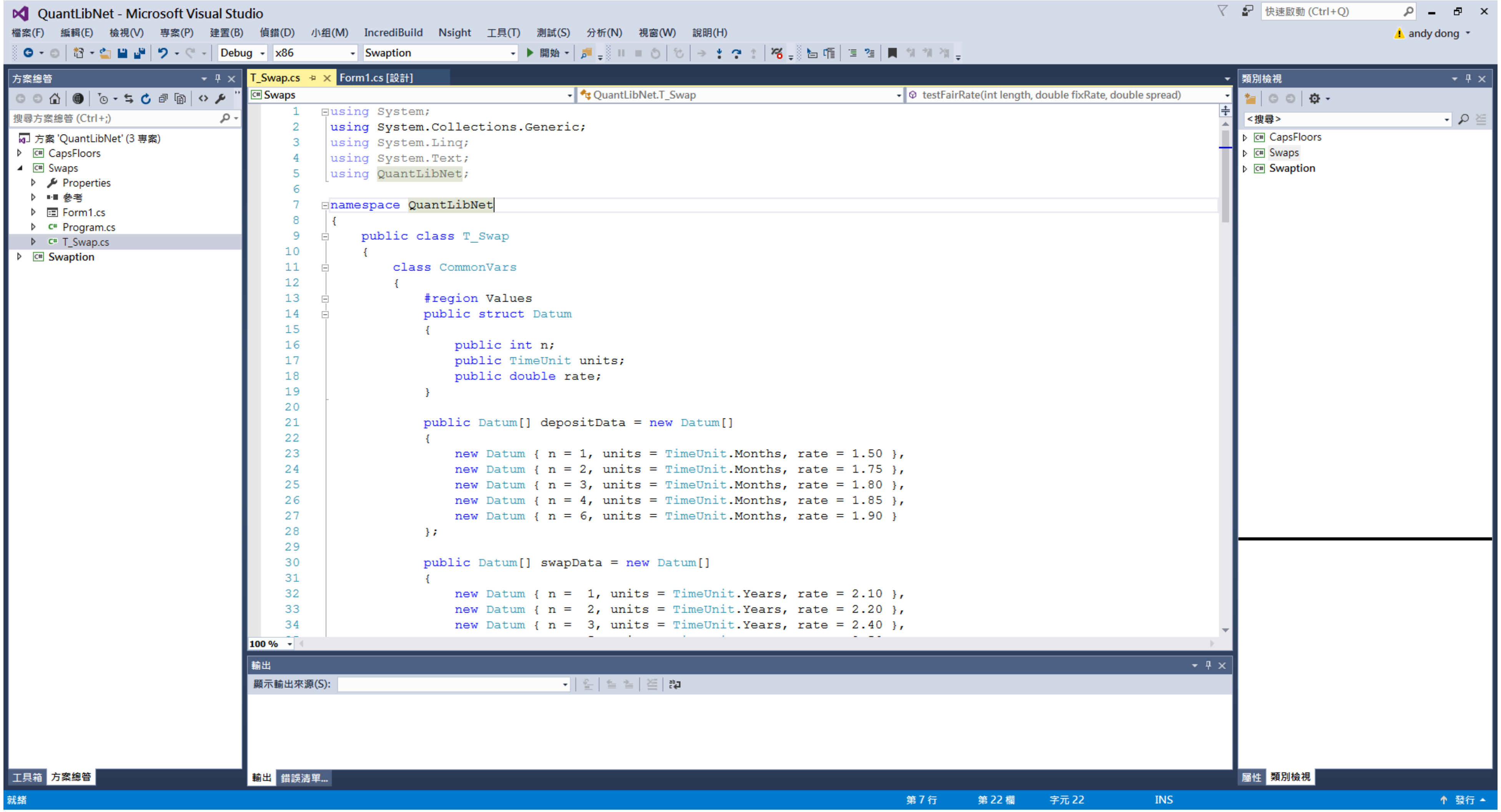
名稱(N): T_Swap.cs

類型: Visual C# 項目

空的類別定義

加入新項目 - Swaps
已安裝的
Visual C# 項目
Web
Windows Forms
WPF
一般
程式碼
資料
MySQL
SQL Server
T4 Toolbox
Workflow
圖形
線上
排序依據: 預設
搜尋已安裝的範本 (Ctrl+E)
類別 Visual C# 項目
介面 Visual C# 項目
Windows Form Visual C# 項目
使用者控制項 Visual C# 項目
元件類別 Visual C# 項目
使用者控制項 (WPF) Visual C# 項目
ADO.NET 實體資料模型 Visual C# 項目
EF 5.x DbContext 產生器 Visual C# 項目
EF 6.x DbContext 產生器 Visual C# 項目
HTML 頁面 Visual C# 項目
JavaScript 檔 Visual C# 項目
LINQ to SQL 類別 Visual C# 項目
請按這裡連線尋找範本。
新增(A)
取消

➢ Add Code



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using QuantLibNet;

namespace QuantLibNet
{
    public class T_Swap
    {
        class CommonVars
        {
            #region Values
            public struct Datum
            {
                public int n;
                public TimeUnit units;
                public double rate;
            }

            public Datum[] depositData = new Datum[]
            {
                new Datum { n = 1, units = TimeUnit.Months, rate = 1.50 },
                new Datum { n = 2, units = TimeUnit.Months, rate = 1.75 },
                new Datum { n = 3, units = TimeUnit.Months, rate = 1.80 },
                new Datum { n = 4, units = TimeUnit.Months, rate = 1.85 },
                new Datum { n = 6, units = TimeUnit.Months, rate = 1.90 }
            };

            public Datum[] swapData = new Datum[]
            {
                new Datum { n =  1, units = TimeUnit.Years, rate = 2.10 },
                new Datum { n =  2, units = TimeUnit.Years, rate = 2.20 },
                new Datum { n =  3, units = TimeUnit.Years, rate = 2.40 },
```

➢ T_Swap Object

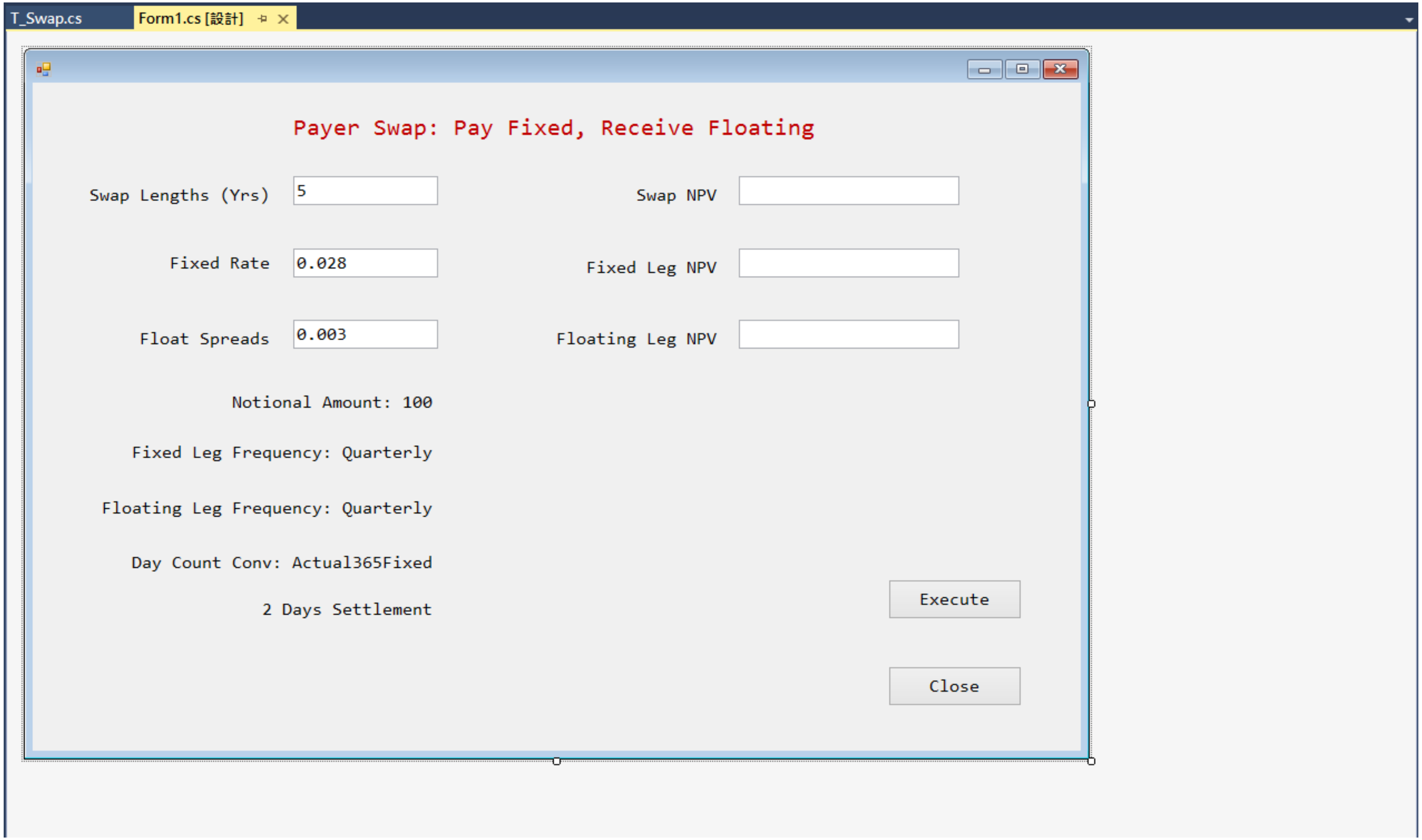
```
public class T_Swap
{
    class CommonVars
    {
        #region Values
        public struct Datum
        {
            public int n;
            public TimeUnit units;
            public double rate;
        }
        public Datum[] depositData = new Datum[]
        {
            new Datum { n = 1, units = TimeUnit.Months, rate = 1.50 },
            new Datum { n = 2, units = TimeUnit.Months, rate = 1.75 },
            new Datum { n = 3, units = TimeUnit.Months, rate = 1.80 },
            new Datum { n = 4, units = TimeUnit.Months, rate = 1.85 },
            new Datum { n = 6, units = TimeUnit.Months, rate = 1.90 }
        };
        public Datum[] swapData = new Datum[]
        {
            new Datum { n =  1, units = TimeUnit.Years, rate = 2.10 },
            new Datum { n =  2, units = TimeUnit.Years, rate = 2.20 },
    new Datum { n =  3, units = TimeUnit.Years, rate = 2.40 },
    new Datum { n =  5, units = TimeUnit.Years, rate = 2.50 },
    new Datum { n =  7, units = TimeUnit.Years, rate = 3.00 }
};
#endregion

// global data
public Date today, settlement;
public Calendar calendar;
public IborIndex index;
public DayCounter fixedDayCount;
public Frequency fixedFrequency, floatingFrequency;
public BusinessDayConvention fixedConvention, floatingConvention;
public YieldTermStructure termstructure;
public RelinkableHandle<YieldTermStructure> RHtermstructure = new RelinkableHandle<YieldTermStructure>();
public VanillaSwap.Type type;
public double nominal;
public int settlementDays;

public CommonVars()
{
    type = VanillaSwap.Type.Payer;
    settlementDays = 2;
    nominal = 100.0;
```

```
fixedConvention = BusinessDayConvention.Unadjusted;
floatingConvention = BusinessDayConvention.Unadjusted;
fixedFrequency = Frequency.Quarterly;
floatingFrequency = Frequency.Quarterly;
fixedDayCount = new Actual365Fixed();
this.index = new Twcpba(new Period(floatingFrequency), RHtermstructure);
calendar = this.index.fixingCalendar();
today = calendar.adjust(Date.Today);
Settings.setEvaluationDate(today);
settlement = calendar.advance(today, settlementDays, TimeUnit.Days);

//*****************************************************************************************************
int deposits = depositData.Length,  // 5
     swaps = swapData.Length;       // 5
var instruments = new List<BootstrapHelper<YieldTermStructure>>
    (deposits + swaps);  // 10
IborIndex index = new IborIndex("TWCPBA", new Period(3, TimeUnit.Months), settlementDays,
    new Currency(), calendar, BusinessDayConvention.Unadjusted, false, new Actual365Fixed());
for (int i = 0; i < deposits; i++)
{
    instruments.Add(new DepositRateHelper(depositData[i].rate / 100, new Period(depositData[i].n,
        depositData[i].units), settlementDays, calendar, BusinessDayConvention.ModifiedFollowing,
        true, new Actual365Fixed()));
}
```

```
    for (int i = 0; i < swaps; ++i)
    {
        instruments.Add(new SwapRateHelper(swapData[i].rate / 100, new Period(swapData[i].n, swapData[i].units),
            calendar, Frequency.Quarterly, BusinessDayConvention.Unadjusted, new Actual365Fixed(), index));
    }
    termstructure = new PiecewiseYieldCurve<Discount, Linear>(settlement, instruments, new Actual365Fixed());

    //****************************************************************************************************
    RHtermstructure.linkTo(termstructure);
}

public VanillaSwap makeSwap(int length,double fixedRate,double floatingSpread)
{
    Date maturity = calendar.advance(settlement, length, TimeUnit.Years, floatingConvention);
    Schedule fixedSchedule = new Schedule(settlement, maturity, new Period(fixedFrequency), calendar,
        fixedConvention, fixedConvention, DateGeneration.Rule.Forward, false);
    Schedule floatSchedule = new Schedule(settlement, maturity, new Period(floatingFrequency), calendar,
        floatingConvention, floatingConvention, DateGeneration.Rule.Forward, false);
    VanillaSwap swap = new VanillaSwap(type, nominal, fixedSchedule,
        fixedRate, fixedDayCount, floatSchedule, index, floatingSpread, index.dayCounter());
    swap.setPricingEngine(new DiscountingSwapEngine(RHtermstructure));
    return swap;
}
```

```
        }

        public static VanillaSwap testFairRate(int length, double fixRate, double spread)
        {
            CommonVars vars = new CommonVars();
            int lengths = length;
            double spreads = spread;
            VanillaSwap swap = vars.makeSwap(lengths, fixRate, spreads);
            return swap;
        }
    }
```

◆ Main Form



Payer Swap: Pay Fixed, Receive Floating

Swap Lengths (Yrs)	5	Swap NPV	
Fixed Rate	0.028	Fixed Leg NPV	
Float Spreads	0.003	Floating Leg NPV	

Notional Amount: 100

Fixed Leg Frequency: Quarterly

Floating Leg Frequency: Quarterly

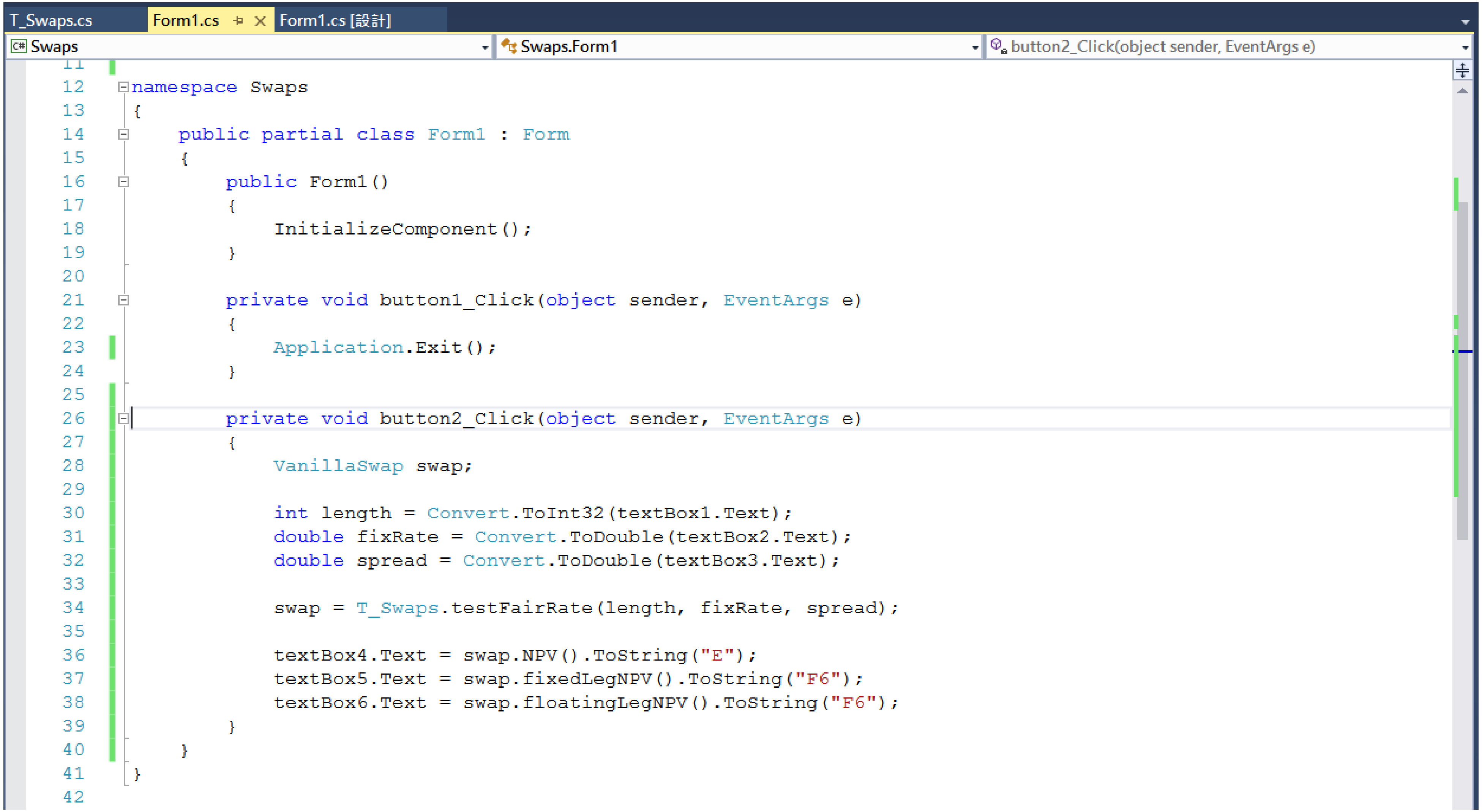
Day Count Conv: Actual365Fixed

2 Days Settlement

- Double Click Close Button，Add Code。

```
using System;
using System.Windows.Forms;
using QuantLibNet;

namespace Swaps
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
    }
}
```

- ➢ Double Click Execute Button。

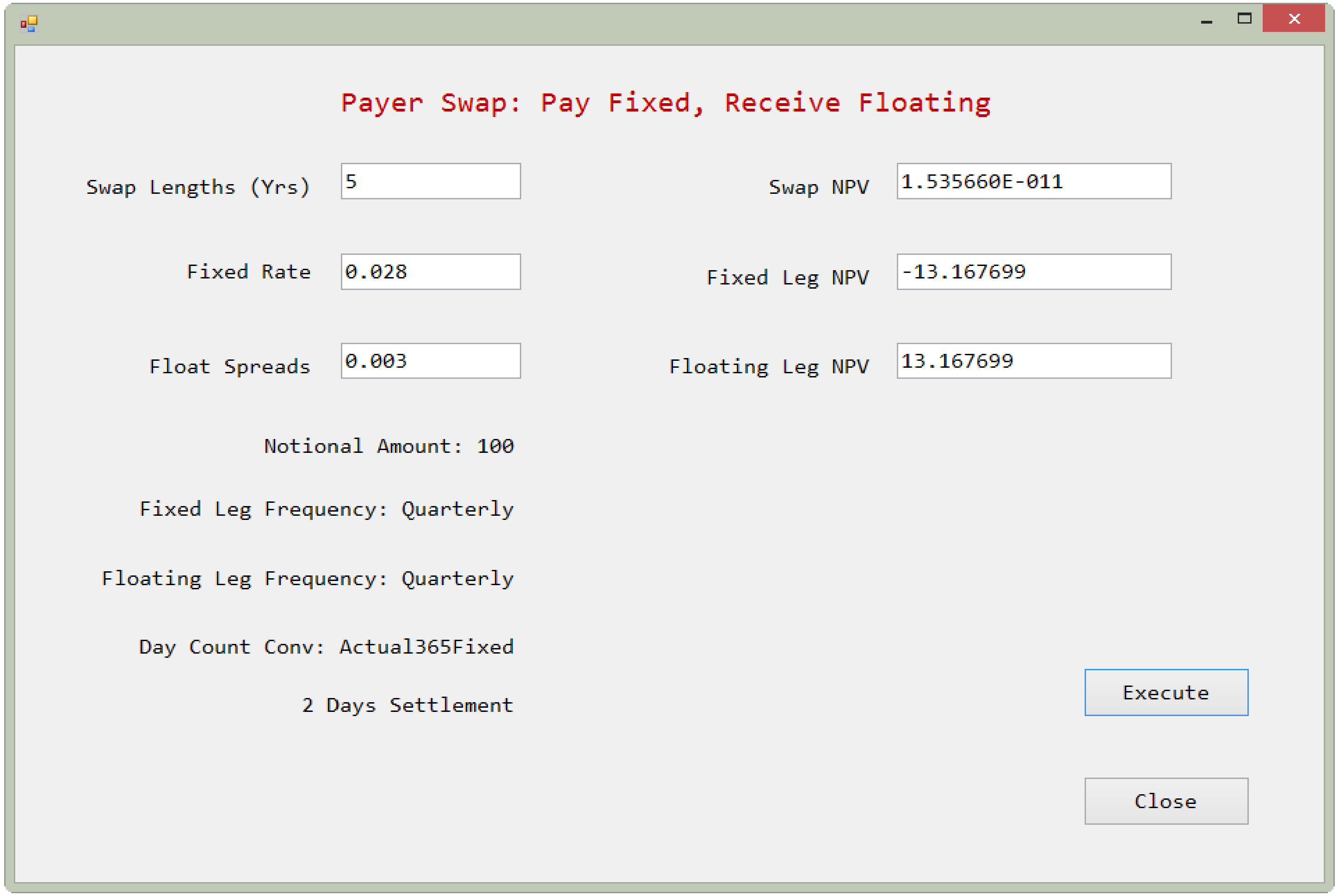


```
namespace Swaps
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            VanillaSwap swap;

            int length = Convert.ToInt32(textBox1.Text);
            double fixRate = Convert.ToDouble(textBox2.Text);
            double spread = Convert.ToDouble(textBox3.Text);

            swap = T_Swaps.testFairRate(length, fixRate, spread);

            textBox4.Text = swap.NPV().ToString("E");
            textBox5.Text = swap.fixedLegNPV().ToString("F6");
            textBox6.Text = swap.floatingLegNPV().ToString("F6");
        }
    }
}
```

Add Codes, Main Form

```
namespace Swaps
{
    public partial class Form1 : Form
    {
        private void button1_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
        private void button2_Click(object sender, EventArgs e)
        {
            VanillaSwap swap;
            int length = Convert.ToInt32(textBox1.Text);
            double fixRate = Convert.ToDouble(textBox2.Text);
            double spread = Convert.ToDouble(textBox3.Text);
            swap = T_Swap.testFairRate(length, fixRate, spread);

            textBox4.Text = swap.NPV().ToString("E");
            textBox5.Text = swap.fixedLegNPV().ToString("F6");
            textBox6.Text = swap.floatingLegNPV().ToString("F6");
        }
    }
}
```

- Execute
  - Output



Payer Swap: Pay Fixed, Receive Floating

Swap Lengths (Yrs) 5

Fixed Rate 0.028

Float Spreads 0.003

Swap NPV 1.535660E-011

Fixed Leg NPV -13.167699

Floating Leg NPV 13.167699

Notional Amount: 100

Fixed Leg Frequency: Quarterly

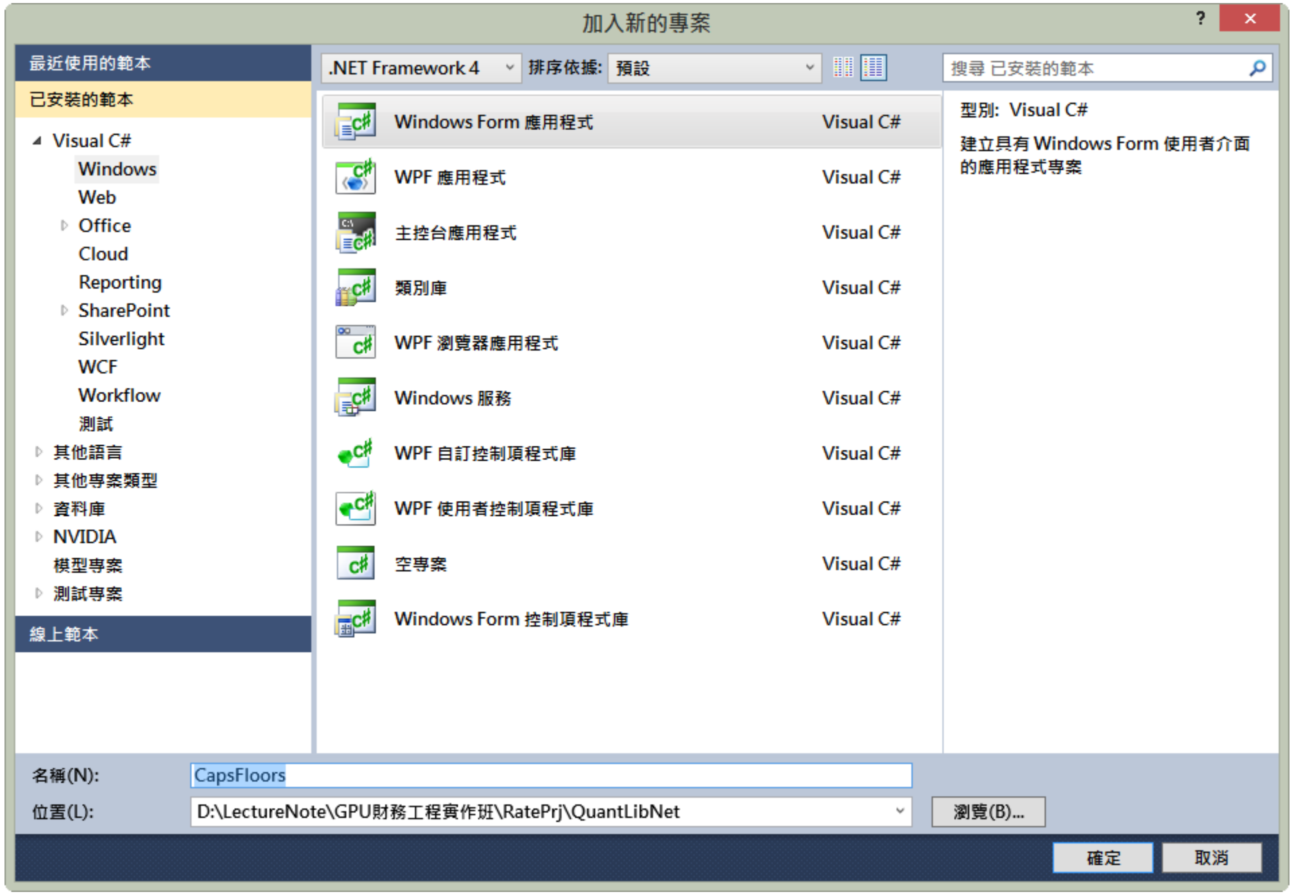
Floating Leg Frequency: Quarterly

Day Count Conv: Actual365Fixed

2 Days Settlement

3.3 Caps/Floors Valuation

- Add Project，CapsFloors



.NET Framework 4　排序依據: 預設

型別: Visual C#

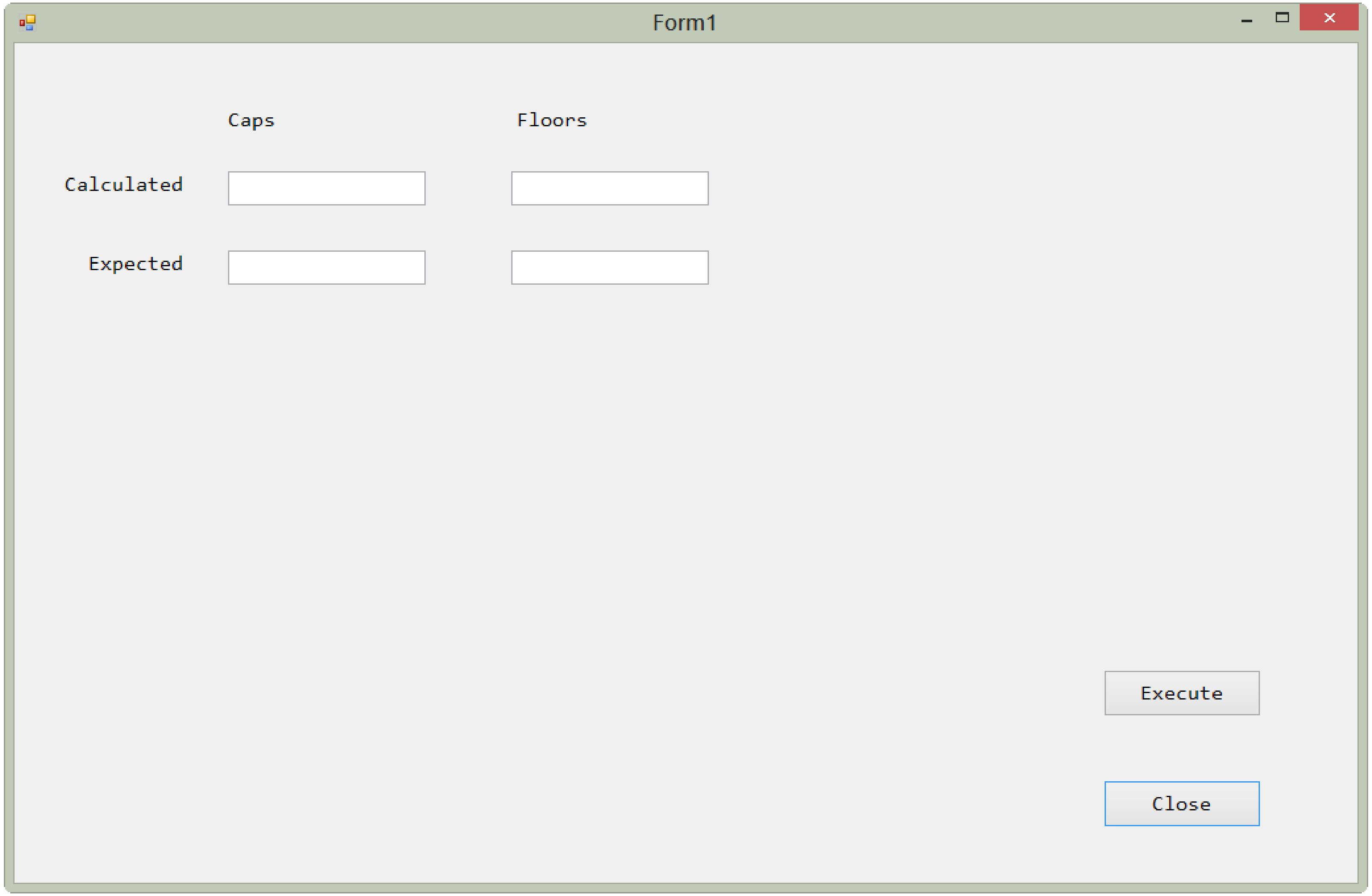
建立具有 Windows Form 使用者介面的應用程式專案

名稱(N): CapsFloors

位置(L): D:\LectureNote\GPU財務工程實作班\RatePrj\QuantLibNet

- Add Reference

Main Form



➢ Add Codes

```
using QuantLibNet;
namespace CapsFloors
{
    public partial class Form1 : Form
    {
        private void button1_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
        private void button2_Click(object sender, EventArgs e)
        {
            T_IRO aIRO = new T_IRO(); Cap cap; Floor floor;
            cap = (Cap) aIRO.testCapsValue();
            floor = (Floor)aIRO.testFloorsValue();
            // par coupon price
            double cachedCapNPV = 6.87570026732, cachedFloorNPV = 2.65812927959;
            textBox1.Text = cap.NPV().ToString("F12");
            textBox3.Text = cachedCapNPV.ToString("F12");
            textBox2.Text = floor.NPV().ToString("F12");
            textBox4.Text = cachedFloorNPV.ToString("F12");
        }
    }
}
```

Test Program

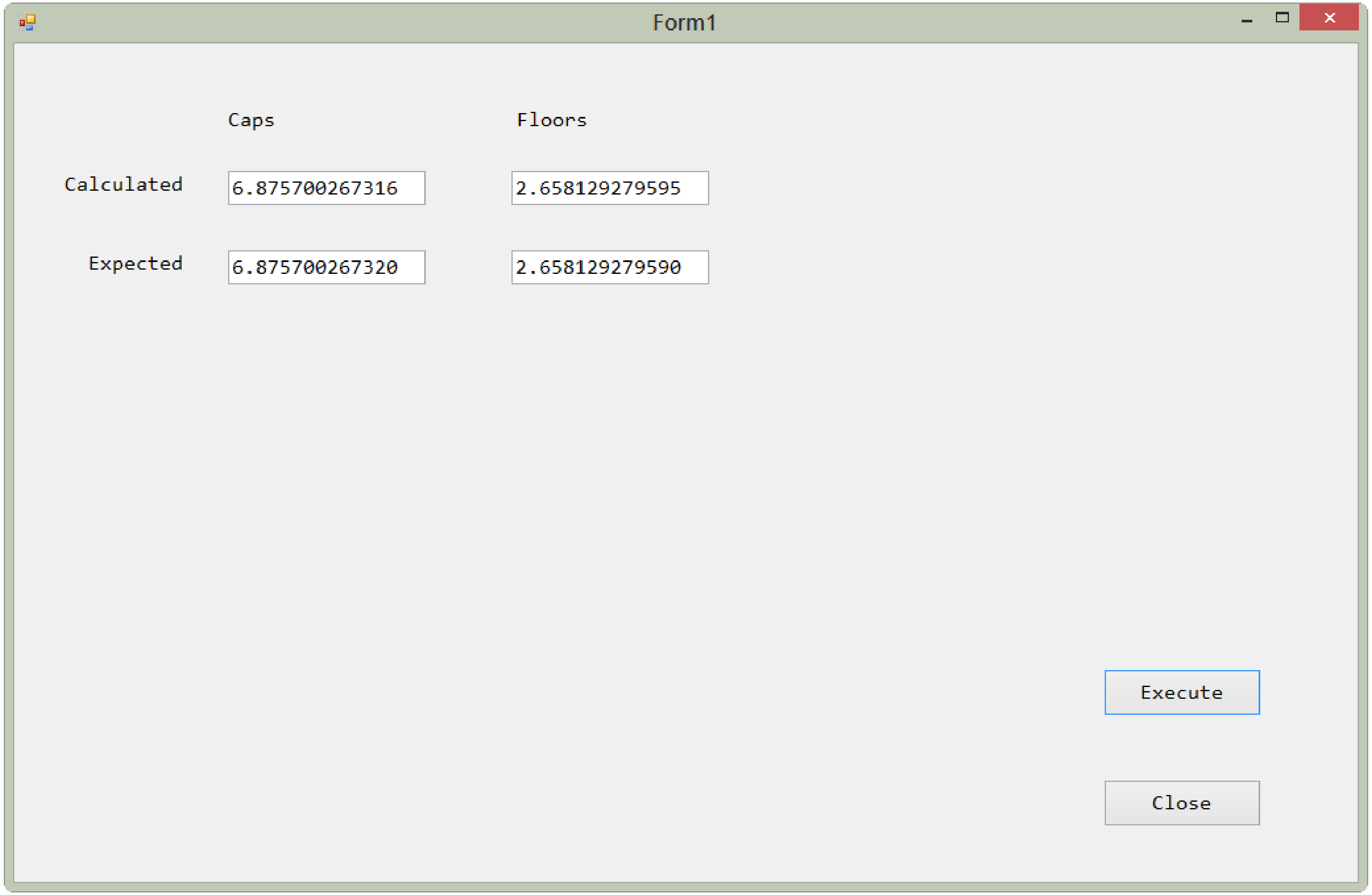
```
public class T_IRO
{
   public class CommonVars
   {
    // common data
    public Date settlement;
    public List<double> nominals;
    public BusinessDayConvention convention;
    public Frequency frequency;
    public IborIndex index;
    public Calendar calendar;
    public int fixingDays;
    public RelinkableHandle<YieldTermStructure> termStructure = new RelinkableHandle<YieldTermStructure>();
    // cleanup
    public SavedSettings backup;

    // setup
    public CommonVars()
    {
        nominals = new List<double>() { 100 };
        frequency = Frequency.Semiannual;
        index = (IborIndex)new Euribor6M(termStructure);
        calendar = index.fixingCalendar();
        convention = BusinessDayConvention.ModifiedFollowing;
        Date today = calendar.adjust(Date.Today);
        Settings.setEvaluationDate(today);
        int settlementDays = 2;
        fixingDays = 2;
        settlement = calendar.advance(today, settlementDays, TimeUnit.Days);
        termStructure.linkTo(Utilities.flatRate(settlement, 0.05, new ActualActual(ActualActual.Convention.ISDA)));
    }

    // utilities
    public List<CashFlow> makeLeg(Date startDate, int length)
    {
        Date endDate = calendar.advance(startDate, new Period(length, TimeUnit.Years), convention);
        Schedule schedule = new Schedule(startDate, endDate, new Period(frequency),calendar, convention, convention,
             DateGeneration.Rule.Forward, false);
        return new IborLeg(schedule, index).withNotionals(nominals)
              .withPaymentDayCounter(index.dayCounter())
              .withPaymentAdjustment(convention)
              .withFixingDays(fixingDays).value();
    }

    public IPricingEngine makeEngine(double volatility)
    {
        Handle<Quote> vol = new Handle<Quote>(new SimpleQuote(volatility));
        return (IPricingEngine)new BlackCapFloorEngine(termStructure, vol);
    }

    public CapFloor makeCapFloor(CapFloor.Type type, List<CashFlow> leg, double strike, double volatility)
    {
        CapFloor result;
        switch (type)
        {
            case CapFloor.Type.Cap:
                result = (CapFloor)new Cap(leg, new List<double>() { strike });
                break;
            case CapFloor.Type.Floor:
                result = (CapFloor)new Floor(leg, new List<double>() { strike });
                break;
            default:
                throw new ArgumentException("unknown cap/floor type");
        }
        result.setPricingEngine(makeEngine(volatility));
        return result;
    }
}
```

```
bool checkAbsError(double x1, double x2, double tolerance)
{
    return Math.Abs(x1 - x2) < tolerance;
}

string typeToString(CapFloor.Type type)
{
    switch (type)
    {
        case CapFloor.Type.Cap:
            return "cap";
        case CapFloor.Type.Floor:
            return "floor";
        case CapFloor.Type.Collar:
            return "collar";
        default:
            throw new ArgumentException("unknown cap/floor type");
    }
}
```

```
public Cap testCapsValue()
{
    CommonVars vars = new CommonVars();

    Date cachedToday = new Date(14, Month.March, 2002), cachedSettlement = new Date(18, Month.March, 2002);
    Settings.setEvaluationDate(cachedToday);

    vars.termStructure.linkTo(Utilities.flatRate(cachedSettlement, 0.05, new Actual360()));
    Date startDate = vars.termStructure.link.referenceDate();
    List<CashFlow> leg = vars.makeLeg(startDate, 20);
    Cap cap = (Cap) vars.makeCapFloor(CapFloor.Type.Cap, leg, 0.07, 0.20);

    return cap;
}
```

```
    public Floor testFloorsValue()
    {
        CommonVars vars = new CommonVars();

        Date cachedToday = new Date(14, Month.March, 2002), cachedSettlement = new Date(18, Month.March, 2002);
        Settings.setEvaluationDate(cachedToday);

        vars.termStructure.linkTo(Utilities.flatRate(cachedSettlement, 0.05, new Actual360()));
        Date startDate = vars.termStructure.link.referenceDate();
        List<CashFlow> leg = vars.makeLeg(startDate, 20);
        Floor floor = (Floor) vars.makeCapFloor(CapFloor.Type.Floor, leg, 0.03, 0.20);

        return floor;
    }
}
```

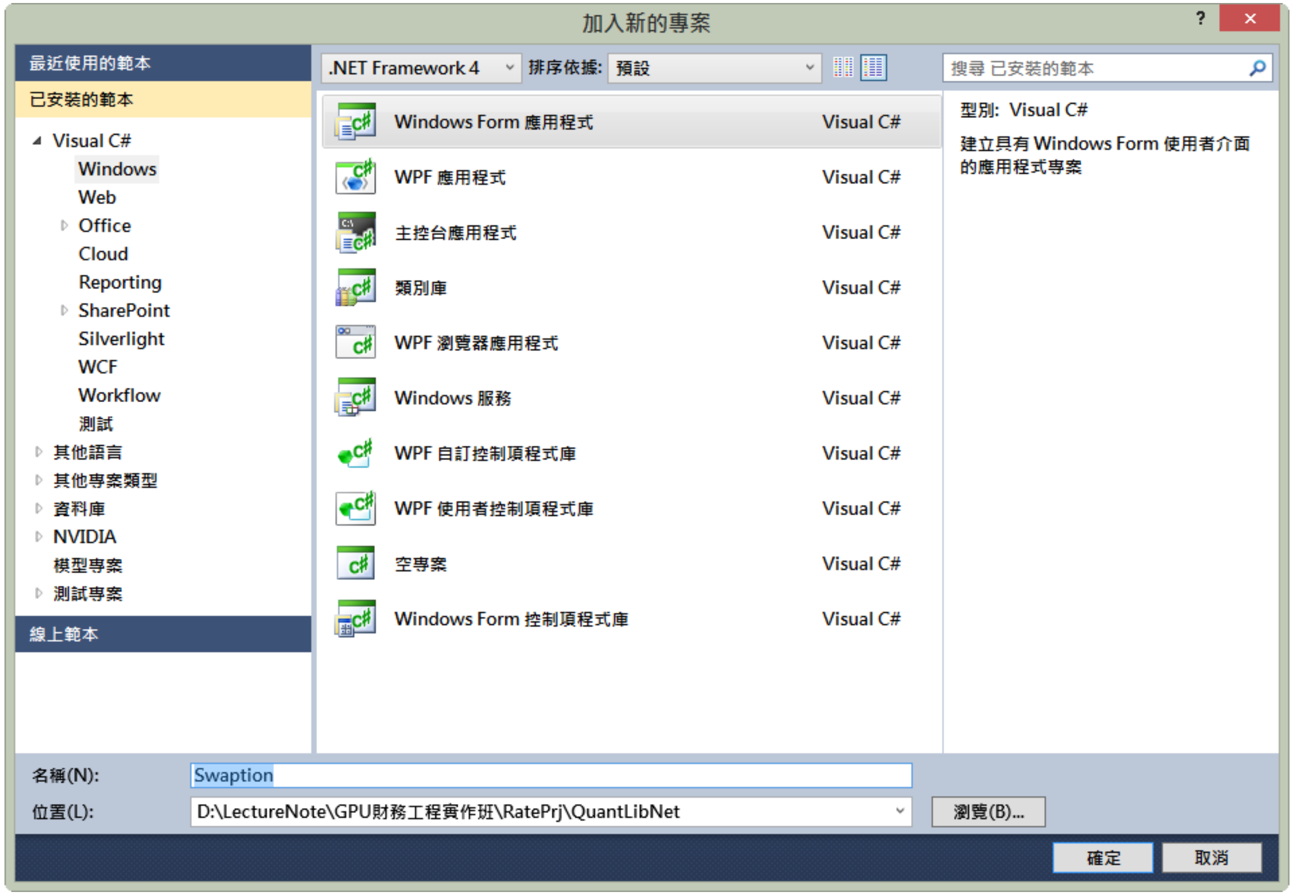
◆ Output Result



	Caps	Floors
Calculated	6.875700267316	2.658129279595
Expected	6.875700267320	2.658129279590

3.4 Swaption Valuation

- Add New Project



.NET Framework 4 排序依據: 預設

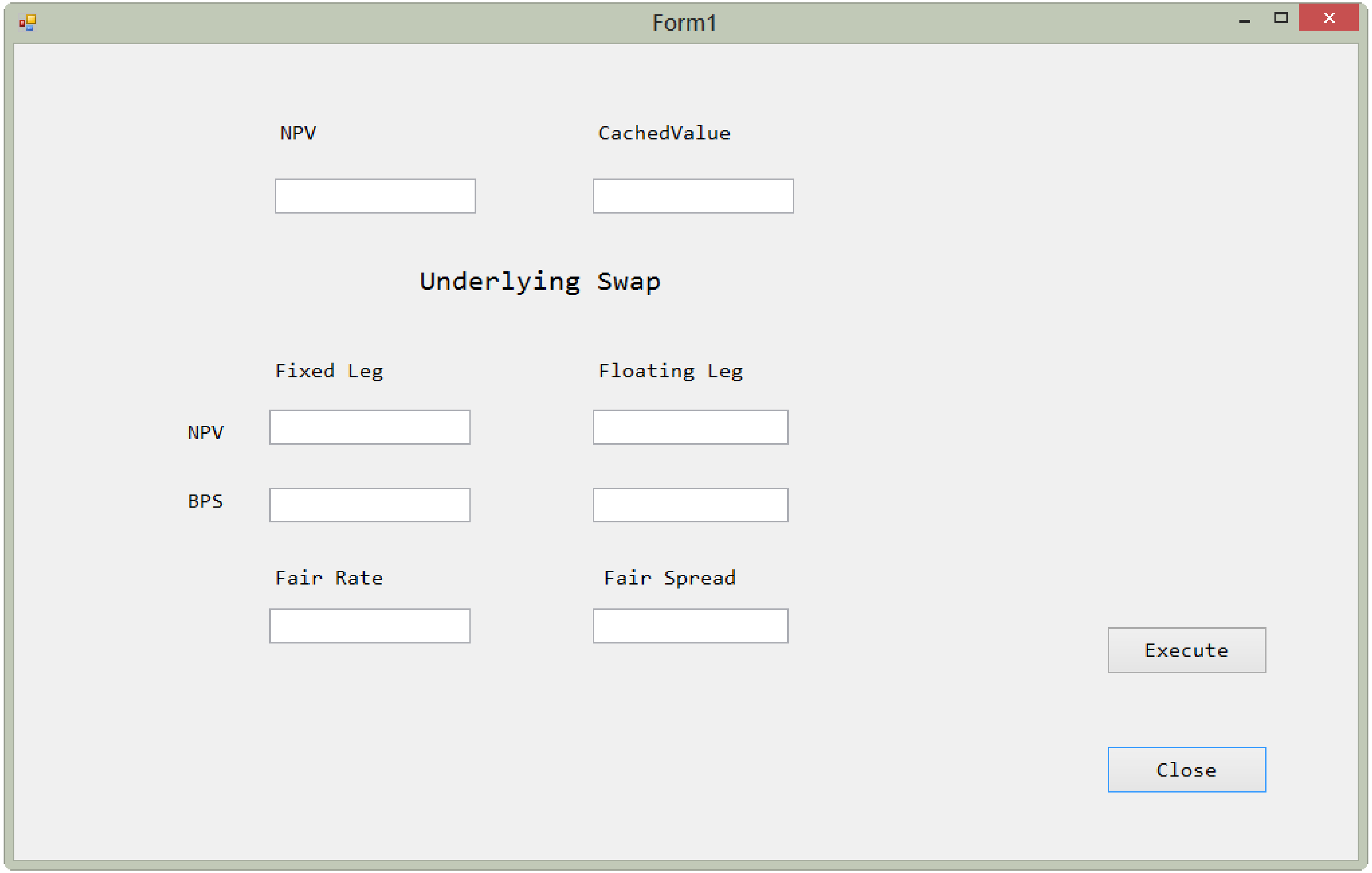
型別: Visual C#

建立具有 Windows Form 使用者介面的應用程式專案

名稱(N): Swaption

位置(L): D:\LectureNote\GPU財務工程實作班\RatePrj\QuantLibNet

◆ Main Form

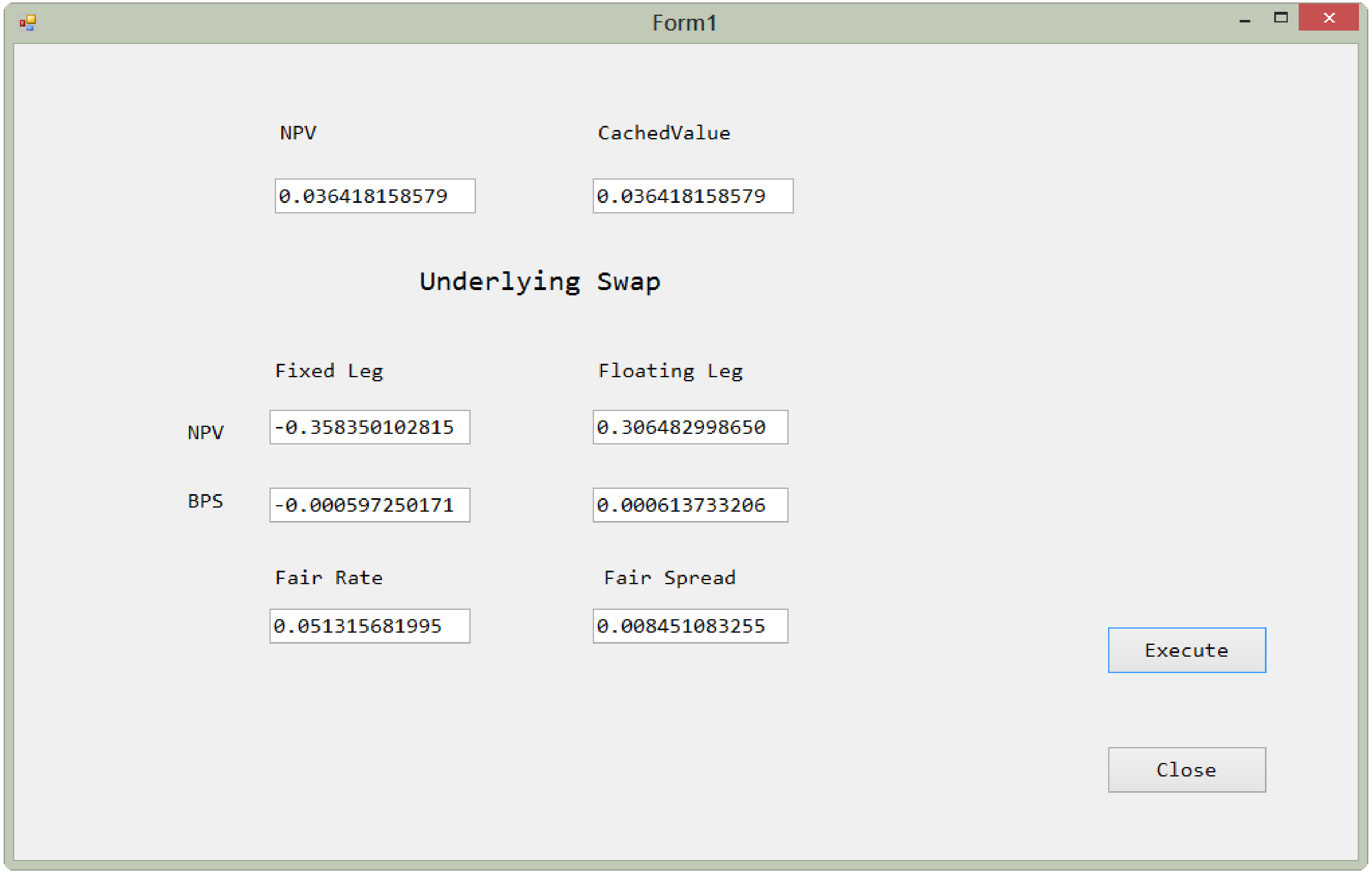

Form1
NPV
CachedValue
Underlying Swap
Fixed Leg
Floating Leg
NPV
BPS
Fair Rate
Fair Spread
Execute
Close

◆ Add Code

```
public class T_Swaption
{
    public Period[] exercises =
    {   new Period(1, TimeUnit.Years), new Period(2, TimeUnit.Years),
        new Period(3, TimeUnit.Years), new Period(5, TimeUnit.Years),
        new Period(7, TimeUnit.Years), new Period(10, TimeUnit.Years) };

    public Period[] lengths =
    {   new Period(1, TimeUnit.Years), new Period(2, TimeUnit.Years),
        new Period(3, TimeUnit.Years), new Period(5, TimeUnit.Years),
        new Period(7, TimeUnit.Years), new Period(10, TimeUnit.Years),
        new Period(15, TimeUnit.Years), new Period(20, TimeUnit.Years) };

    public VanillaSwap.Type[] type = {VanillaSwap.Type.Receiver, VanillaSwap.Type.Payer};

    public static QuantLibNet.Swaption testCachedValue()
    {
        Console.WriteLine("Testing swaption value against cached value...");
        TSwaptionCommonVars vars = new TSwaptionCommonVars();

        vars.today = new Date(13, 3, 2002);
        vars.settlement = new Date(15, 3, 2002);
        Settings.setEvaluationDate(vars.today);
        vars.termStructure.linkTo(Utilities.flatRate(vars.settlement, 0.05, new Actual365Fixed()));
        Date exerciseDate = vars.calendar.advance(vars.settlement,
            new Period(5, TimeUnit.Years));
        Date startDate = vars.calendar.advance(exerciseDate, vars.settlementDays,
            TimeUnit.Days);
        VanillaSwap swap = new MakeVanillaSwap(new Period(10, TimeUnit.Years), vars.index,
            0.06).withEffectiveDate(startDate);
        QuantLibNet.Swaption swaption = vars.makeSwaption(swap, exerciseDate, 0.20);

        return swaption;
    }
}
```

```
public class TSwaptionCommonVars
{
    // global data
    public Date today, settlement;
    public double nominal;
    public Calendar calendar;
    public BusinessDayConvention fixedConvention;
    public Frequency fixedFrequency;
    public DayCounter fixedDayCount;
    public BusinessDayConvention floatingConvention;
    public Period floatingTenor;
    public IborIndex index;
    public int settlementDays;
    public RelinkableHandle<YieldTermStructure> termStructure
        = new RelinkableHandle<YieldTermStructure>(new YieldTermStructure());

    // utilities
    public Swaption makeSwaption(VanillaSwap swap,  Date exercise, double volatility)
    {
        Settlement.Type settlementType = Settlement.Type.Physical;
        Handle<Quote> vol = new Handle<Quote>(new SimpleQuote(volatility));
        IPricingEngine engine = new BlackSwaptionEngine(termStructure, vol);
        Swaption result = new Swaption(swap, new EuropeanExercise(exercise), settlementType);
        result.setPricingEngine(engine);
        return result;
    }
    public IPricingEngine makeEngine(double volatility)
    {
        Handle<Quote> h = new Handle<Quote>(new SimpleQuote(volatility));
        return new BlackSwaptionEngine(termStructure, h);
    }

    public TSwaptionCommonVars()
    {
        settlementDays = 2;
        nominal = 1000000.0;
        fixedConvention = BusinessDayConvention.Unadjusted;
        fixedFrequency = Frequency.Annual;
        fixedDayCount = new Thirty360();

        index = new Euribor6M(termStructure);
        floatingConvention = index.businessDayConvention();
        floatingTenor = index.tenor();
        calendar = index.fixingCalendar(); //new TARGET();
        today = calendar.adjust(Date.Today);
        Settings.setEvaluationDate(today);
        settlement = calendar.advance(today, settlementDays, TimeUnit.Days);
        termStructure.linkTo(Utilities.flatRate(settlement, 0.05, new Actual365Fixed()));
    }
}
```

◆ Output Result



NPV

0.036418158579

CachedValue

0.036418158579

Underlying Swap

	Fixed Leg	Floating Leg
NPV	-0.358350102815	0.306482998650
BPS	-0.000597250171	0.000613733206

Fair Rate

0.051315681995

Fair Spread

0.008451083255