# Лабораторная работа 1. Вариант 2

# Механизм привязки данных в Windows Presentation Foundation

В лабораторной работе надо создать пользовательский интерфейс приложения для интерполяции дискретных данных с помощью одномерного интерполяционного кубического сплайна, пересчета данных на другую сетку и вычисления интеграла от интерполяционного сплайна по заданному отрезку с использованием пакета сплайн-интерполяции из библиотеки Intel MKL.

В среде VisualStudio надо cоздать решение (solution) с тремя проектами:

* тип одного проекта – **Dll-библиотека C++;**
* тип второго проекта – **библиотека классов (class library) C#**, в которой находятся типы **FRaw, FRawEnum, RawData,** **SplineData, SplineDataItem;**
* тип третьего проекта – приложение **Windows Presentation Foundation (WPF).**

Dll−библиотека C++

В Dll−библиотеке **C++** находится глобальная функция, которая вызывается из кода **C#** с использованием механизма **PInvoke.** Из этой глобальной функции вызываются функции из библиотеки **Intel MKL**.

Из кода **C#** через параметры глобальной функции передаются данные, которые необходимы для вычисления сплайнов. Через параметры глобальной функции в код **C#** возвращаются результаты и код ошибки.

Библиотека классов C#

Библиотека классов C# содержит следующие типы.

Делегат типа **double FRaw (double x);**

Перечисление (enum) **FRawEnum** с именами функций одной переменной для инициализации данных, для которых будет выполняться сплайн-интерполяция. Среди функций, которые можно вызвать для инициализации данных, должны быть

* линейная функция;
* кубический многочлен;
* функция, которая вычисляет значения с помощью генератора псевдослучайных чисел **Random**.

Класс **RawData** содержит данные, для которых будет выполняться сплайн-интерполяция. Данные заданы на сетке значений независимой переменной на отрезке **[a,b]**.

В классе **RawData** определены открытые

* два свойства типа **double (**или одно свойство типа **double[])** для концов отрезка **[a,b]**;
* свойство типа **int** для числа узлов сетки;
* свойство типа **bool** для информации о том, что сетка является равномерной/неравномерной;
* свойство типа **FRaw;**
* свойство типа **double[]** для массива узлов сетки;
* свойство типа **double[]** для массива значений функции в узлах сетки;
* конструктор c двумя параметрами типа **double (**или одним типа **double[])** для концов отрезка **[a,b],** типа **int** (для числа узлов сетки), **bool** (для информации о том, что сетка равномерная/неравномерная) и **FRaw** (для информации о функции, с помощью которой инициализируются данные для сплайн-интерполяции);
* конструктор c одним параметрами типа **string** для имени файла, в котором хранятся данные для инициализации **RawData**;
* функции, отвечающие делегату **FRaw**, для инициализации массива значений функции в узлах сетки;
* метод **Save(string filename),** который сохраняет данные объекта **RawData** в файле с именем filename;
* статический метод **Load(string filename, out RawData rawData),** который читает данные объекта **RawData** из файла.

Метод **void Save(string filename)**

* сохраняет данные объекта **RawData** в файл с именем **filename**;
* если файл с именем **filename** существует, приложение его перезаписывает; если такого файла нет, приложение его создает;
* метод бросает исключение (перебрасывает в вызывающий метод), если в процессе сохранения или при создании/открытии файла произошла ошибка;
* независимо от того, как завершилось сохранение данных, все файловые потоки должны быть закрыты в блоке **finally**.

Статический метод **void** **Load(string filename, out** **RawData rawData)**

* восстанавливает данные объекта **RawData** из файла с именем **filename**;
* метод бросает исключение (перебрасывает в вызывающий метод), если в процессе восстановления данных или при открытии файла произошла ошибка;
* независимо от того, как завершилась восстановление данных, все файловые потоки должны быть закрыты в блоке **finally**.

Структура **SplineDataItem** содержит открытые автореализуемые свойства

* типа **double** с координатой точки, в которой вычислены значение сплайна и его первой и второй производных;
* три свойства типа **double** для значений сплайна и его первой и второй производных (или одно свойство типа **double[]**).

В структуре **SplineDataItem** определены открытые

* конструктор с параметрами типа **double** для значений координаты точки и значений сплайна и его первой и второй производных в этой точке;
* метод **string ToString(string** **format),** возвращающий строку, которая содержит значения всех свойств структуры**;** параметр **format** задает формат вывода чисел с плавающей запятой;
* перегруженная (**override**) версия виртуального метода **string ToString().**

Класс **SplineData** содержит входные данные для сплайн-интерполяции, метод, в котором вызываются функции из библиотеки MKL для вычисления сплайна, и результаты сплайн-интерполяции.

Класс **SplineData** содержит открытые

* свойство типа **RawData**;
* свойство типа **int** для числа узлов равномерной сетки, на которой будут вычислены значения сплайна; первый узел сетки совпадает с левой границей отрезка интерполяции **[a,b]**; последний узел сетки совпадает с правой границей отрезка интерполяции;
* свойство типа **List<SplineDataItem>** для коллекции вычисленных значений сплайна и его производных на равномерной сетке;
* конструктор с параметрами типа **RawData** (для инициализации ссылки на объект с данными, для которых вычисляется сплайн), типа **double** (для значения второй производной на левом конце отрезка интерполяции),типа **double** (для значения второй производной на правом конце отрезка), типа **int** (для числа узлов равномерной сетки, на которой вычисляются значения сплайна и его производных);
* метод, в котором вызываются функции из библиотеки **Intel MKL** для построения сплайна, вычисления значений сплайна, его производных и интеграла по отрезку **[a,b]**; в этом методе добавляются элементы в коллекцию **List<SplineDataItem>**;
* свойство типа **double** для значения интеграла от интерполяционного сплайна по отрезку **[a,b]**.

Пользовательский интерфейс приложения

Главное окно приложения содержит следующие элементы управления, в которые пользователь вводит данные для инициализации объектов **RawData** и **SplineData.**

Элементы управления для ввода параметров для объекта **RawData**

* **TextBox** для левой границы отрезка интерполяции;
* **TextBox** для правой границы отрезка интерполяции;
* **TextBox** для числа узлов интерполяции (число узлов сплайна);
* два элемента типа **RadioButton**, с помощью которых пользователь выбирает тип сетки **(**равномерная/неравномерная), на которой заданы данные для интерполяции;
* **ComboBox** для выбора функции для инициализации массива значений функции.

Элементы управления для ввода параметров для объекта **SplineData**

* **TextBox** для числа узлов сетки, на которой вычисляются значения интерполяционного сплайна;
* **TextBox** для значений второй производной на концах конца отрезка интерполяции; значения второй производной пользователь вводит в один элемент управления **TextBox** через разделитель; в привязке используется преобразователь типа (converter).

**Элементы управления для вывода данных.**

Главное окно приложения содержит следующие элементы для вывода информации**:**

* **ListBox** для вывода информации из **RawData**; для каждого узла сетки в элемент управления **ListBox** выводятся значение координаты узла и значение функции в узле;
* **ListBox** для вывода информации из **SplineData**; источником данных для этого элемента является коллекция **List<SplineDataItem>;** для каждого узла сетки, на которой вычисляются значения сплайна, в элемент управления **ListBox** выводятся значение координаты узла, значение сплайна и значение второй производной в узле; при выводе используется шаблон данных **DataTemplate**;
* **TextBlock**, в который выводитсязначение координаты узла, значение сплайна и его первой и второй производных в узле, который пользователь выбрал в элементе управления **ListBox** для вывода информации из **SplineData**.
* **TextBlock** для вывода значения интеграла от интерполяционного сплайна по всему отрезку **[a,b]**.

Элементы управления для ввода информации должны быть подписаны – в главном окне приложения должна быть информация, для ввода каких данных предназначены элементы управления.

Главное окно приложения содержит меню с элементами

* **File** (c элементом **Save**);
* **Execute** (c элементами **RawData from Controls и RawData from File**).

и три кнопки, дублирующие элементы меню.

Реакция приложения на выбор пользователем элементов меню (и дублирующих их кнопок):

**Save** – пользователь выбирает имя файла в стандартном диалоге **Microsoft.Win32**.**SaveFileDialog**. Если пользователь сделал выбор (закрыл диалог кнопкой **Save**), данные **RawData** сохраняются в файле с именем, который выбрал пользователь.

**RawData from File** – пользователь выбирает имя файла в стандартном диалоге **Microsoft.Win32**.**OpenFileDialog**. Если пользователь сделал выбор (закрыл диалог кнопкой **Open**) выполняется инициализация объекта **RawData** из файла, вызываются функции для вычисления сплайна и результаты выводятся в элементы управления.

**RawData from Controls** – выполняется инициализация объекта **RawData** данными, которые пользователь ввел в элементы управления, вызываются функции для вычисления сплайна и результаты выводятся в элементы управления.

Класс ViewData

В проекте **WPF** надо определить класс **ViewData,** который содержит свойства и методы для обмена данными между пользовательским интерфейсом и данными из классов **RawData** и **SplineData.** В классе главного окна приложениянадо определить поле (или свойство) типа **ViewData** и использовать его как источник данных в привязкахк элементам управления для ввода параметров. Класс **ViewData** содержит ссылки на объекты **RawData** и **SplineData**. Все операции приложения с данными из классов **RawData** и **SplineData** происходят через вызовы методов или свойств класса **ViewData.**

Класс **ViewData** содержит открытые свойства, которые используются в привязках к элементам управления для ввода параметров для **RawData** и **SplineData**. Некоторые свойства, определенные в классе **ViewData**, дублируют свойства классов **RawData** и **SplineData**.

Свойства класса **ViewData**, которые используются в привязках при вводе параметров для объекта **RawData**

* свойства типа **double** (или **double[]**)для границ отрезка интерполяции;
* свойство типа **int** для числа узлов сетки (узлы сплайна), на которой заданы значения функции, для которой будет выполняться интерполяция;
* свойство типа **bool** для информации о том, что сетка является равномерной/неравномерной;
* свойство типа **FRawEnum** для выбора функции для инициализации массива значений функции.

Свойства класса **ViewData**, которые используются в привязках при вводе параметров для объекта **SplineData**

* свойство типа **int** для числа узлов сетки, на которой вычисляются значения интерполяционного сплайна и его производных;
* свойство типа **double** для значения второй производной на левом конце отрезка интерполяции;
* свойство типа **double** для значения второй производной на правом конце отрезка интерполяции.

Класс **ViewData** содержит

* метод **void Save(string filename**), в котором вызывается метод **Save** из класса **RawData**; если метод **Save** из класса **RawData** бросает исключение, пользователю с помощью окна сообщений **MessageBox** выводится информация о причине, по которой не удалось сохранить данные в файле;
* метод **void** **Load(string filename)**, в котором вызывается метод **Load** из класса **RawData;** в случае успешного завершения свойствам класса **ViewData** присваиваются новые значения; если метод **Load** из класса **RawData** бросает исключение, пользователю с помощью окна сообщений **MessageBox** выводится информация о причине, по которой не удалось прочитать данные из файла;
* конструктор;
* методы, в которых создаются объекты **RawData** и **SplineData** и вызываются методы из этих классов.

Обработка исключений

Все исключения, которые могут возникать при обработке некорректного ввода пользователя, должны обрабатываться приложением.

Приложение должно оставаться в рабочем состоянии до тех пор, пока пользователь не закроет главное окно приложения.

Срок сдачи лабораторной работы

27 марта (группа 341/2)

28 марта (группы 301, 302, 306, 309)