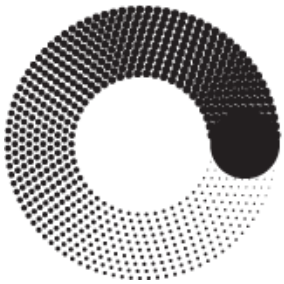
2021

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



МОСКОВСКИЙ ПОЛИТЕХ

Записка

По блоку лабораторных/практических работ И.П.

дисциплина: «Инженерный проект»

Группа 201-324

Студент Пересторонин Аким Максимович

Структура записки.

1. Задание на КП.
2. Процесса создания программного обеспечения.
3. Структура программного обеспечения.
4. Функции программного обеспечения.
5. Результаты работы программы.
6. Инструкций пользователя и системного программиста.
7. Код программы.

1. Задание на КП.[1]

Целью данного курсового проекта является разработка программного обеспечения на языке высокого уровня C# в среде Visual Studio 2022 по анализу напряженно-деформированного состояния балок и визуализации результатов расчёта.

Расчёт балки представляет собой построение эпюр (например, эпюр поперечных сил и изгибающих моментов). Формулы и принципы всех вычислений и построений изучаются в дисциплине «Сопротивление материалов». Данное программное обеспечение автоматизирует процесс анализа балки (в учебных целях).

В теоретической части уточняются выбранные для работы инструменты, в практической части описывается подробная реализация интерфейса и сам процесс расчётов. В заключении можно видеть результат работы программы.

В приложении доступна часть исходного кода, которого будет достаточно, чтобы восстановить работающую программу в случае утери или повреждения оригинала.

[1] Взято из системы lms.

2. Процесса создания программного обеспечения.

Мною было проанализирована задача, представленная в задании условной схемой на Рис. 1.1. После его был проанализированы источники (преимущественно иностранные) и порешена задача с условными величинами. В результат чего был сделан вывод о необходимом разбиении на участки, которые требовали отдельного решения (пример решения прикреплен).

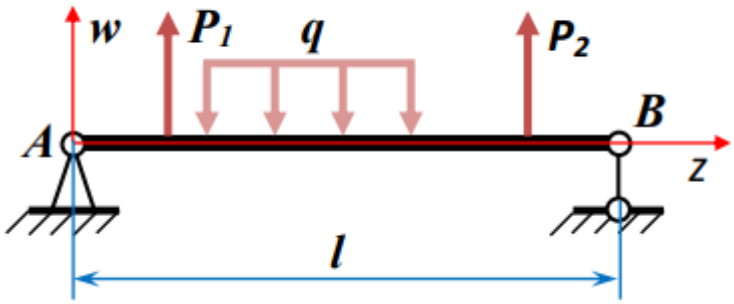


Рис 1.1.

Участки решено было представить в виде методов (PlotOne, PlotTwo, …) находящимся в соответствующем классе (Plots).

Заделки и расчёт сил, возникающих в них было решено так же записать отдельно методами (OneStageFastening, TwoDegreeFastening) в отдельном классе (Support). [2]

В результате структура программы выглядит так:

A. Painting (служебный)
    a. PutPixel
    b. BresenhamLine
B. Plots
    a. PlotOne
    b. PlotTwo
    c. PlotThree
    d. PlotFour
    e. PlotFive
C. Support
    a. OneStageFastening
    b. TwoDegreeFastening

[2]Так как расчеты просты, то использовать потоки, вызываемые в кнопке было неоправданном, хоть и не правильным.

3. Структура программного обеспечения.

Структура была рассмотрена в пункте 2. Так же по рекомендации Линусу Торвальдсу методы принимают от 0 до 3 переменных и по рекомендации Эндрю Таненбаума методы (функции) не возвращают значений. [3]

Структура классов:

D. Painting (служебный)
    a. PutPixel
    b. BresenhamLine
E. Plots
    a. PlotOne
    b. PlotTwo
    c. PlotThree
    d. PlotFour
    e. PlotFive
F. Support
    a. OneStageFastening
    b. TwoDegreeFastening

Глобальные переменные:

A. VDistributedLoad
B. VPowerOne
C. VPowerTwo
D. VLength
E. VForceOneLength
F. VForceTwoLength
G. VDistributedStartLength
H. VDistributedEndLength
I. VLateralForce
J. VBendingMoment

[3]В качестве рекомендуемого пособия по написанию кода было выбраны книги Линусу Торвальдса и Эндрю Таненбаума.

4. Функции программного обеспечения.

Функции программы были.

Вывод эпюр:

1. Поперечной силы.
2. Изгибающего момента.

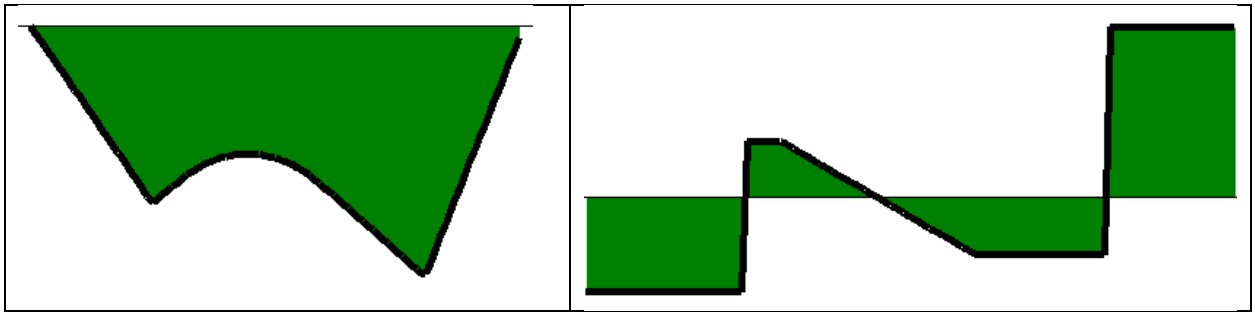
Так же в программе можно вводить значения параметров объектов системы.[4]

[4] Остальные функции находятся в доработки.

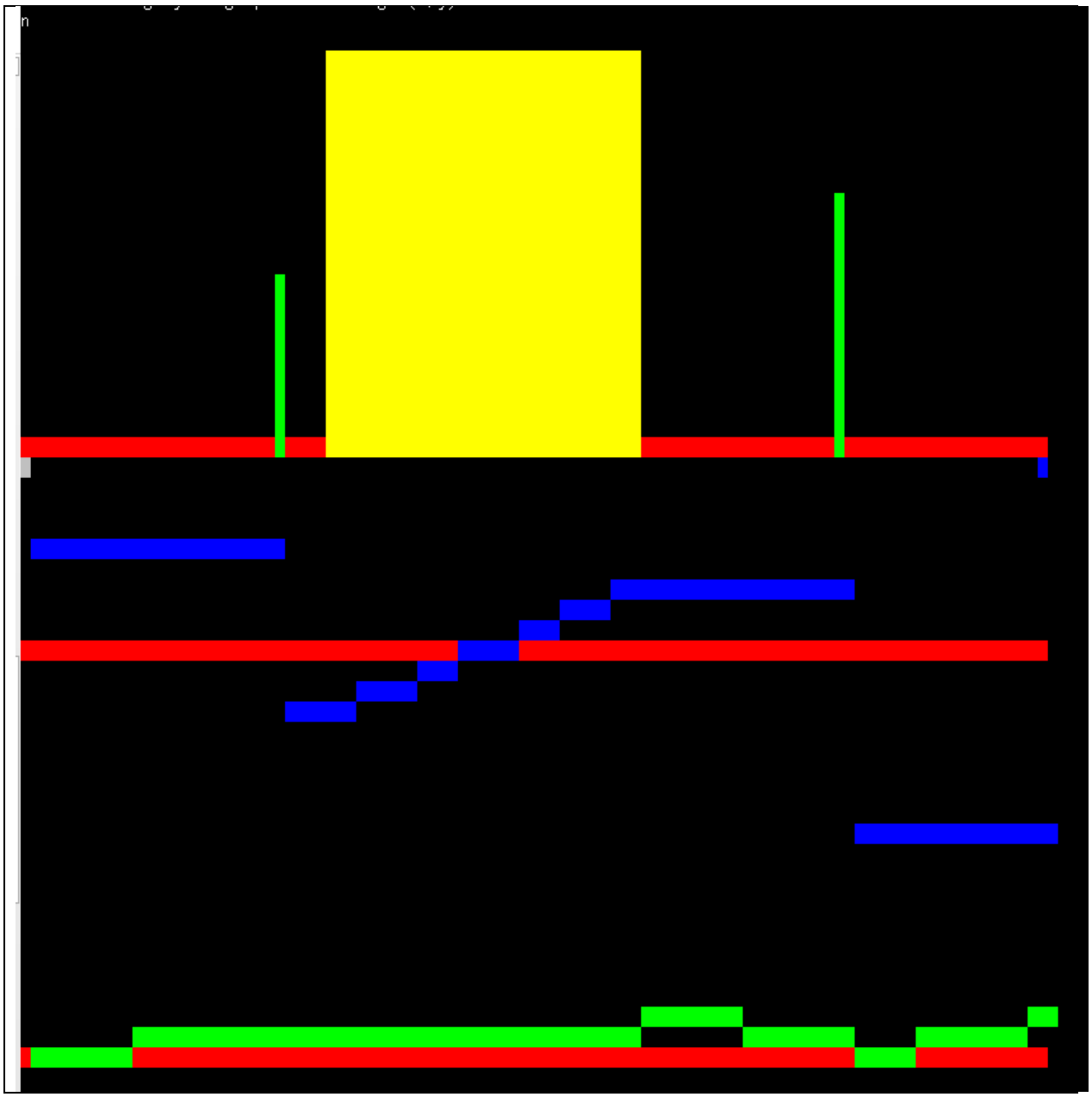
4. Результаты работы программы.

В результате работы программы выводится две эпюры представленных в таблице 1.

Таб. 1.



Так же было сделан консольный вывод для отладки, представленной в Таб. 2.



5. Инструкций пользователя и системного программиста.

После запуска программы можно будет увидеть окно с названием (Beam calculation). Пример Рис 2.1.

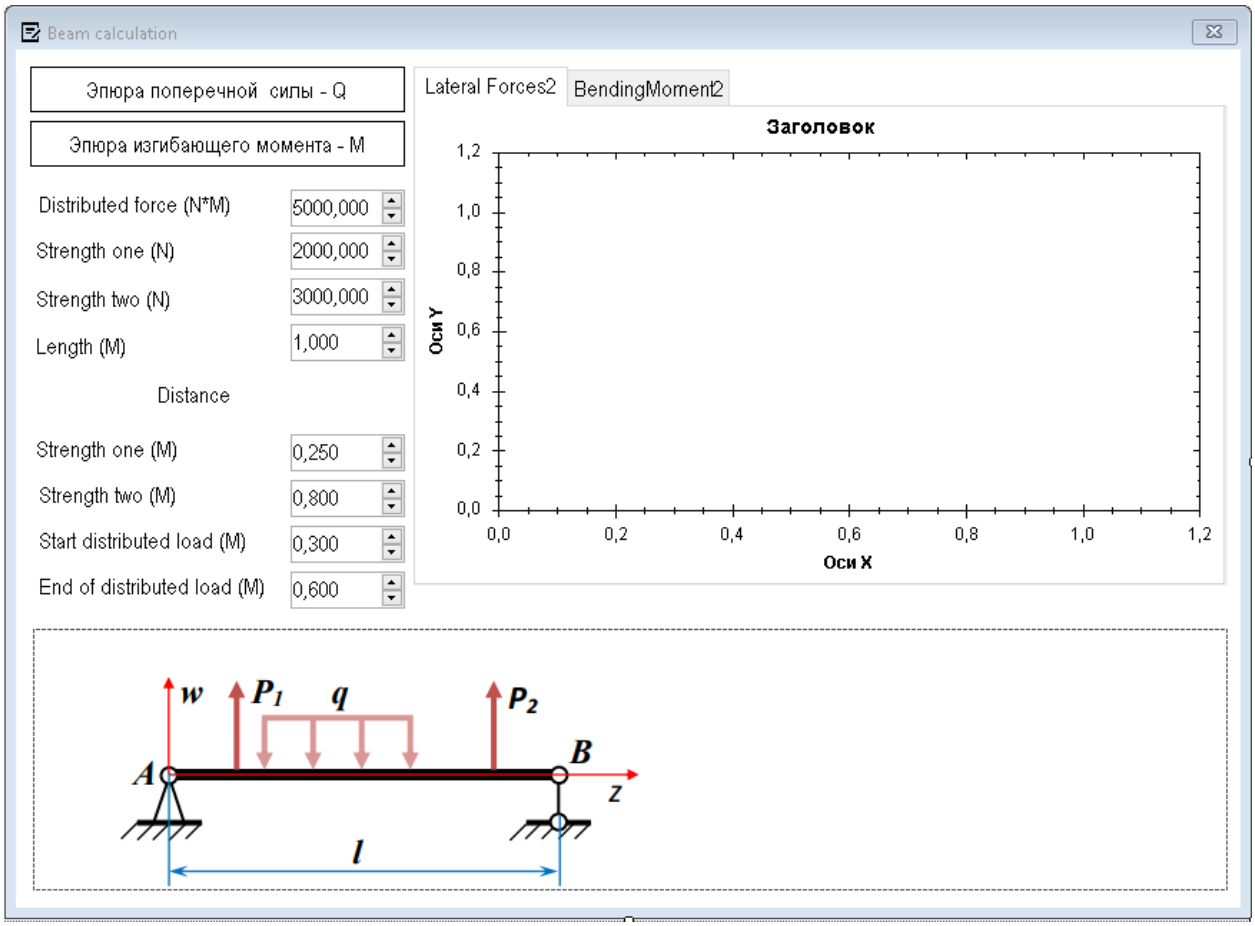


Рис 2.1.

Далее, если требуется, можно изменить значения объектов балки. Пример Рис 2.2.

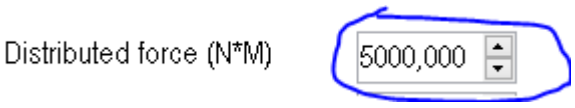


Рис 2.2.

После вышеописанных действий можно нажать кнопку. Если все ведено правильно, то в соответствующем окне (программа) нарисует график эпюр. Пример Рис 2.3.

Если введение данные были неправильны, то вылезет окно где будет описаны недопустимые значения. Измените недопустимые значения в необходимый диапазон. Пример окна Рис. 2.4.

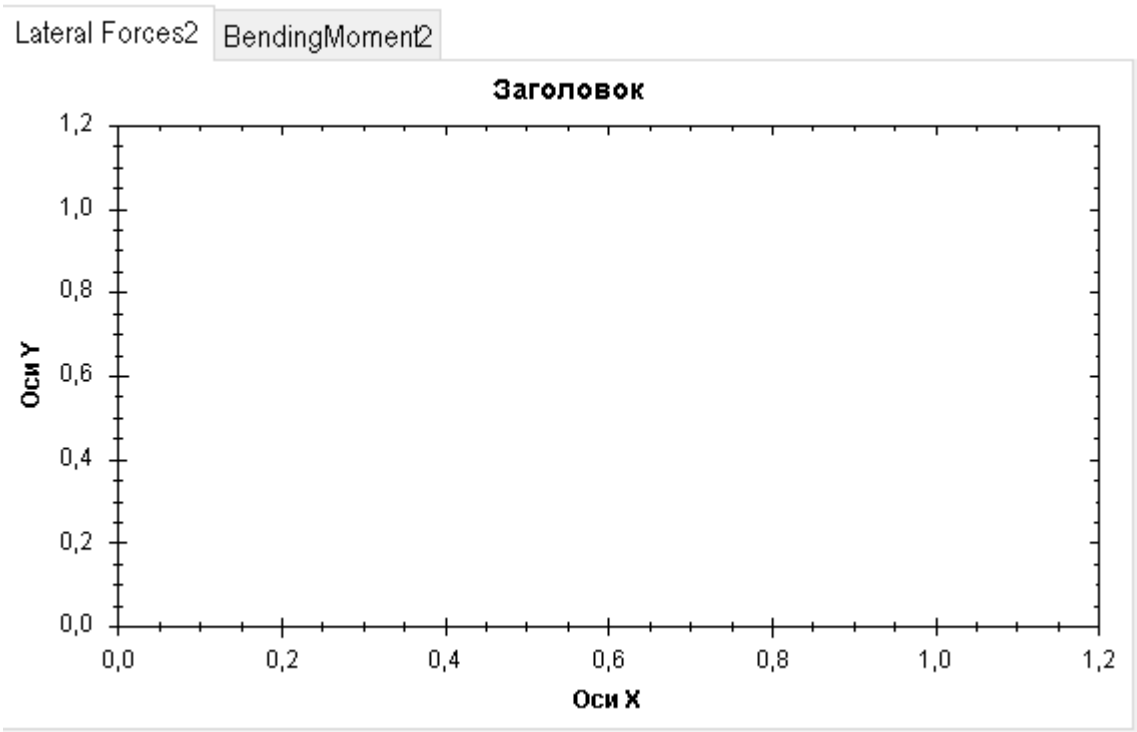

Lateral Forces2
BendingMoment2
Заголовок
1,2
1,0
0,8
0,6
0,4
0,2
0,0
Оси Y
0,0
0,2
0,4
0,6
0,8
1,0
1,2
Оси X

Рис 2.3.

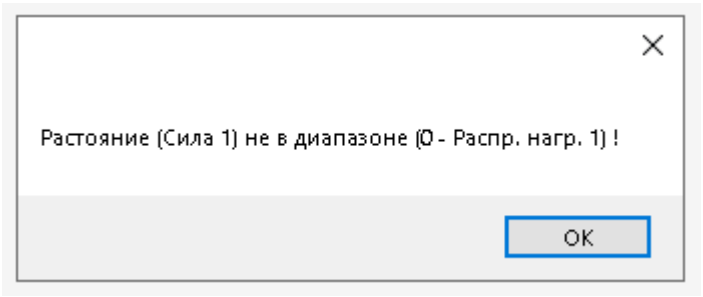

Растояние (Сила 1) не в диапазоне (0 - Распр. нагр. 1) !
OK

Рис 2.4.

6. Код программы.

- **Глобальные переменные.**

```
// Значения конструкции (основные).
        public static double VDistributedLoad = 5000;
        public static double VPowerOne = 2000;
        public static double VPowerTwo = 3000;
        public static double VLength = 1;

        // Размеры (которых нету на чертиже) отмеряющие растояние до точек приложения
нагрузок.
        public static double VForceOneLength = 0.25;
        public static double VForceTwoLength = 0.8;
        public static double VDistributedStartLength = 0.3;
        public static double VDistributedEndLength = 0.6;

        // Переменные (временные для расчетов) момент и сила (Q и M). // Это могут быть
не глобальные так как все расчеты видутся заного!!!!!!!
        public static double VLateralForce = 0;
        public static double VBendingMoment = 0;
```

- **Изгибающей силы.**

```
if (Examination())
            {
                return;
            }
            double Yb = Support.OneStageFastening(); // 2225
            double Ya = Support.TwoDegreeFastening(Yb); // 1275
            ShearForcePlot.GraphPane.CurveList.Clear();
            PointPairList list1 = new PointPairList();
            double x = 0;
            double Step = 0.01;
            // 0 <= z1 <= 0.25(0 - F1)
            for (double step = 0; step < VForceOneLength; step += Step)
            {
                x += Step;
                Plots.PlotOne(Ya, step);
                list1.Add(x, VLateralForce);
            }
            // 0 <= z2 <= 0.05(F1 - q)
            for (double step = 0; step < (VDistributedStartLength - VForceOneLength);
step += Step)
            {
                x += Step;
                Plots.PlotTwo(Ya, step);
                list1.Add(x, VLateralForce);
            }
            // 0 <= z3 <= 0.3(q1 - q2)// парабола
            for (double step = 0; step < (VDistributedEndLength -
VDistributedStartLength); step += Step)
            {
                x += Step;
                Plots.PlotThree(Ya, step);
                list1.Add(x, VLateralForce);
            }
            // 0 <= z4 <= 0.2(q2 - F2)
            for (double step = (VForceTwoLength - VDistributedEndLength); step > 0; step
-= Step)
            {
                x += Step;
                Plots.PlotFive(Yb, step);
                list1.Add(x, VLateralForce);
            }
```

```
            // 0 <= z4 <= 0.2(F2 - l)
            for (double step = (VLength - VForceTwoLength); step > 0; step -= Step)
            {
                x += Step;
                Plots.PlotFour(Yb, step);
                list1.Add(x, VLateralForce);
            }
            LineItem MyLine1 = ShearForcePlot.GraphPane.AddCurve("Эпюра поперечной  силы
- Q", list1, Color.Black, SymbolType.None);
            MyLine1.Line.Width = 5;
            MyLine1.Line.Fill = new Fill(Color.Green);
            MyLine1.Symbol.IsVisible = false;
            ShearForcePlot.RestoreScale(ShearForcePlot.GraphPane);
```

- **Изгибающей момент.**

```
if (Examination())
            {
                return;
            }
            double Yb = Support.OneStageFastening(); // 2225
            double Ya = Support.TwoDegreeFastening(Yb); // 1275
            BendingMomentPlot.GraphPane.CurveList.Clear();
            PointPairList list1 = new PointPairList();
            double x = 0;
            double Step = 0.01;
            // 0 <= z1 <= 0.25(0 - F1)
            for (double step = 0; step < VForceOneLength; step += Step)
            {
                x += Step;
                Plots.PlotOne(Ya, step);
                list1.Add(x, VBendingMoment);
            }
            // 0 <= z2 <= 0.05(F1 - q)
            for (double step = 0; step < (VDistributedStartLength - VForceOneLength);
step += Step)
            {
                x += Step;
                Plots.PlotTwo(Ya, step);
                list1.Add(x, VBendingMoment);
            }
            // 0 <= z3 <= 0.3(q1 - q2)// парабола
            for (double step = 0; step < (VDistributedEndLength -
VDistributedStartLength); step += Step)
            {
                x += Step;
                Plots.PlotThree(Ya, step);
                list1.Add(x, VBendingMoment);
            }
            // 0 <= z4 <= 0.2(q2 - F2)
            for (double step = (VForceTwoLength - VDistributedEndLength); step > 0; step
-= Step)
            {
                x += Step;
                Plots.PlotFive(Yb, step);
                list1.Add(x, VBendingMoment);
            }
            // 0 <= z4 <= 0.2(F2 - l)
            for (double step = (VLength - VForceTwoLength); step > 0; step -= Step)
            {
                x += Step;
                Plots.PlotFour(Yb, step);
                list1.Add(x, VBendingMoment);
            }
```

```
            LineItem MyLine1 = BendingMomentPlot.GraphPane.AddCurve("M экстремум = ",
list1, Color.Black, SymbolType.None);
            MyLine1.Line.Width = 5;
            MyLine1.Line.Fill = new Fill(Color.Green);
            MyLine1.Symbol.IsVisible = false;
            BendingMomentPlot.RestoreScale(BendingMomentPlot.GraphPane);
```

- **Функция проверки правильности введённых данных.**

```
if (((0 < (double)MeaningStrengthOneM.Value) && ((double)MeaningStrengthOneM.Value <
(double)MeaningStartDistributedLoadM.Value)) != true)
            {
                MessageBox.Show("Растояние (Сила 1) не в диапазоне (0 - Распр. нагр. 1)
!");
                return true;
            }
            if ((((double)MeaningStrengthOneM.Value <
(double)MeaningStartDistributedLoadM.Value) &&
((double)MeaningStartDistributedLoadM.Value <
(double)MeaningEndOfDistributedLoadM.Value)) != true)
            {
                MessageBox.Show("Растояние (Распр. нагр. 1) не в диапазоне (Сила 1 -
Распр. нагр. 2) !");
                return true;
            }
            if ((((double)MeaningStartDistributedLoadM.Value <
(double)MeaningEndOfDistributedLoadM.Value) &&
((double)MeaningEndOfDistributedLoadM.Value < (double)MeaningStrengthTwoM.Value)) !=
true)
            {
                MessageBox.Show("Растояние (Распр. нагр. 2) не в диапазоне (Распр. нагр.
1 - Сила 2) !");
                return true;
            }
            if ((((double)MeaningEndOfDistributedLoadM.Value <
(double)MeaningStrengthTwoM.Value) && ((double)MeaningStrengthTwoM.Value <
(double)MeaningLengthM.Value)) != true)
            {
                MessageBox.Show("Растояние (Сила 2) не в диапазоне (Распр. нагр. 2 -
Длина) !");
                return true;
            }
            VDistributedLoad = (double)MeaningDistributedForceNM.Value;
            VPowerOne = (double)MeaningStrengthOneN.Value;
            VPowerTwo = (double)MeaningStrengthTwoN.Value;
            VLength = (double)MeaningLengthM.Value;
            VForceOneLength = (double)MeaningStrengthOneM.Value;
            VForceTwoLength = (double)MeaningStrengthTwoM.Value;
            VDistributedStartLength = (double)MeaningStartDistributedLoadM.Value;
            VDistributedEndLength = (double)MeaningEndOfDistributedLoadM.Value;
            return false;
```

- **Класс заделок.**

```
public static double OneStageFastening()
        {
            // Фактически расчет момента относительно ya (для нахождения yb).
            return ((BeamCalculation.VPowerOne * BeamCalculation.VForceOneLength) +
(BeamCalculation.VPowerTwo * BeamCalculation.VForceTwoLength) -
(BeamCalculation.VDistributedLoad * BeamCalculation.VDistributedStartLength *
(BeamCalculation.VDistributedStartLength + ((BeamCalculation.VDistributedEndLength -
BeamCalculation.VDistributedStartLength) / 2)))) / BeamCalculation.VLength;
        }
        public static double TwoDegreeFastening(double OneStageFastening)
        {
```

```
            // Фактически расчет силы при известном ya (для нахождения ya).
            return BeamCalculation.VPowerOne + BeamCalculation.VPowerTwo -
(BeamCalculation.VDistributedLoad * (BeamCalculation.VDistributedEndLength -
BeamCalculation.VDistributedStartLength)) - OneStageFastening;
        }
```

- **Класс участков.**

```
public static void PlotOne(double Ya, double z1)
        {
            // 0 <= z1 <= 0.25(0 - F1)
            BeamCalculation.VLateralForce = (-1) * Ya;
            BeamCalculation.VBendingMoment = (-1) * Ya * z1;
        }
        public static void PlotTwo(double Ya, double z2)
        {
            // 0 <= z2 <= 0.05(F1 - q)
            BeamCalculation.VLateralForce = BeamCalculation.VPowerOne - Ya;
            BeamCalculation.VBendingMoment = (BeamCalculation.VPowerOne * z2) - (Ya * (z2
+ BeamCalculation.VForceOneLength));
        }
        public static void PlotThree(double Ya, double z3)
        {
            // 0 <= z3 <= 0.3(q1 - q2)
            double z2 = BeamCalculation.VDistributedStartLength -
BeamCalculation.VForceOneLength;
            BeamCalculation.VLateralForce = ((-1) * BeamCalculation.VDistributedLoad *
z3) + BeamCalculation.VPowerOne - Ya;
            BeamCalculation.VBendingMoment = ((-1) * (BeamCalculation.VDistributedLoad *
Math.Pow(z3, 2) / 2)) + (BeamCalculation.VPowerOne * (z3 + z2)) - (Ya * (z3 +
BeamCalculation.VDistributedStartLength));
        }
        public static void PlotFour(double Yb, double z4)
        {
            // 0 <= z4 <= 0.2(F2 - l)
            BeamCalculation.VLateralForce = Yb;
            BeamCalculation.VBendingMoment = (-1) * Yb * z4;
        }
        public static void PlotFive(double Yb, double z5)
        {
            // 0 <= z4 <= 0.2(q2 - F2)
            BeamCalculation.VLateralForce = ((-1) * BeamCalculation.VPowerTwo) + Yb;
            BeamCalculation.VBendingMoment = (BeamCalculation.VPowerTwo * z5) - (Yb * (z5
+ (BeamCalculation.VForceTwoLength - BeamCalculation.VDistributedEndLength)));
        }
```

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

· Анурьев В.И. Справочник конструктора-машиностроителя: в 3 т.: Т.1. — 8 издание, переработанное и дополненное. Под редакцией И.Н. Жестковой. — М.: Машиностроение, 2001. — 920 с.

· Джеффри Рихтер. CLR visual C#. Программирование на платформе Microsoft .NET Framework 4.0 на языке C#. 3-е изд. — СПб.: Питер, 2011. — 928 с.

· Эндрю Троелсен. Язык программирования C# 2010 и платформа .NET 4 Framework, 5-е изд. — М.: Вильямс, 2010. — 1392 с.

· Эндрю Стиллмен, Дженнифер Грин. Изучаем C#. — СПб.: Питер, 2011. — 696 с.

· Холзнер С. Visual C++ 6: учебный курс – СПб: ЗАО «Издательство «Питер», 1999. – 576 с.

· Грегори Кэйт. Использование Visual C++ 6. Специальное издание: Пер. с англ. – М., СПб.: Издательский дом «Вильямс», 2000. – 864 с.

· Современные промышленные роботы: Каталог / Под ред. Козырева Ю.Г., Шифрина Я.Г. – М.: Машиностроение, 1984. – 148 с.

· Stroustrup B. The Design and Evolution of C++. – Reading, MA: Addison – Wesley, 1994. – 509 p.