**Министерство науки и высшего образования Российской Федерации**

**ФГАОУ ВО «УрФУ имени первого Президента России Б.Н. Ельцина»**

Кафедра «школы бакалавриата (школа)»

Оценка работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель от УрФУ Желонкина Н.И.

Тема задания на практику

Механика и математическое моделирование

ОТЧЕТ

Вид практики Учебная практика

Тип практики Учебная практика, Научно-исследовательская работа (получение первичных навыков научно-исследовательской работы)

Руководитель практики от предприятия (организации) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ФИО руководителя Подпись**

Студент Баруздин Н. А.

**ФИО студента**

Специальность (направление подготовки) 01.03.03 Механика и математическое моделирование

Группа МЕН-290705

Екатеринбург 2021

**Оглавление**

[Вступление 3](#_tyjcwt)

[Решение поставленной задачи 4](#_3dy6vkm)

[Условие задачи 4](#_1t3h5sf)

[Решение 5](#_4d34og8)

[Участок АВ: 5](#_2s8eyo1)

[Участок ВС: 8](#_17dp8vu)

[Участок СE: 10](#_3rdcrjn)

[Описание программы 12](#_26in1rg)

[Приложение](#_515xmstqq7mi) 14

[Файл MainWindow.xaml.сs 1](#_35nkun2)4

[Файл MainWindow.xaml:](#_1ksv4uv) 18

[Подведение итогов работы](#_643db6ua07pk) 24

[Список литературы](#_2jxsxqh) 25

# **Вступление**

Данная научно-исследовательская работа посвящена изучению механики, математическому моделированию и программированию. В отличие от работ по теоретической механике данная работа даст нам возможность увидеть траекторию движения нашего тела, а изменения некоторый значения увидеть изменения в ней.

Были поставлены следующие цели и задачи.

Цель – создание модели движения тела по полученным уравнениям движения

Задачи:

1. Ознакомиться с порядком оформления и написания научно-исследовательской работы
2. Изучить теоретическую часть для решения задачи
3. Решить поставленную задачу
4. Написать и объяснить программу:
   1. изучить язык программирования C# и язык разметки XAML
   2. спроектировать интерфейс с помощью языка разметки XAML
   3. добавить ввод данных с помощью языков XAML и C#
   4. с помощью полученных уравнений движений нарисовать анимированную траектории в окне с помощью языков XAML и C#
5. Сделать выводы о проделанной работе
6. Отчитаться за проделанную работу в письменной форме.

# **Решение поставленной задачи**

## **Условие задачи**

Тело массой *т*, получив в точке *А* начальную скорость V*0,* движется по поверхности *АВС*, расположенной в вертикальной плоскости. В точке *С* тело покидает поверхность и в точке *Е* падает на берег рва. Движение тела разбито на три участка.

1. На **участке *АВ*** на тело, кроме силы тяжести, действуют движущая сила *Q,* и сила сопротивления среды *R.* Трением тела о поверхность на участке *АВ* пренебречь.

2. В точке *В* тело, не изменяя величины своей скорости движения, переходит на **участок *ВС***, где на него действуют сила тяжести, сила трения (коэффициент трения скольжения груза о поверхность *f =*0,2) и переменная сила *F.* Время движения груза по участку *ВС* составляет *t =* 4 *с*.

3. Движение тела на **участке *СЕ*** происходит под действием силы тяжести; сопротивлением воздуха пренебречь.

Требуется, считая тело материальной точкой и зная расстояние *AB= L*,

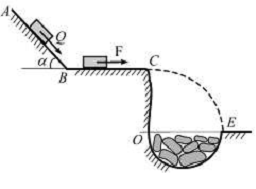
или время движения тела на участке АВ (*τ),* вычислить:

1. скорость движения тела в точке *В* – *V* B;
2. уравнение движения тела на участке *ВС* – *x= f* (*t*);
3. скорость движения тела в точке *С* (*V*C) в момент времени *4 c*;
4. уравнение траектории движения тела на участке *СЕ*, т. е. *y3= f3(x).*

Начальные условия

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| m,  кг | V0,  м/c | Q,  Н | R,  Н | μ | L,  м | Fx,  Н | α,  (град) | CO,  м |
| 3 | 22 | 9 | μ V | 0,5 | 3 | 30t | 45 | 5 |

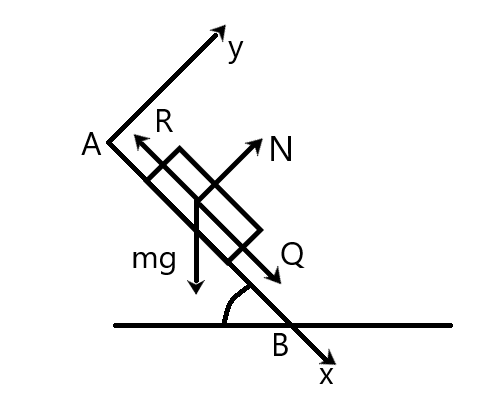
Схема



## **Решение**

### ***Участок АВ:***

При решении задачи мы пренебрегаем трением на этом участке. Принимая тело за материальную точку, покажем действующие на него силы: вес , нормальную реакцию , движущую силу и силу сопротивления среды . Возьмем начало координат в точке А, ось направим вдоль прямой AB.



Составим дифференциальное уравнение движения тела на участке АВ:

Разделим правую и левую часть на :

По условию сила сопротивления среды равна или тоже . Таким образом:

Проинтегрируем. Получили:

Пусть тогда:

Найдем нашу :

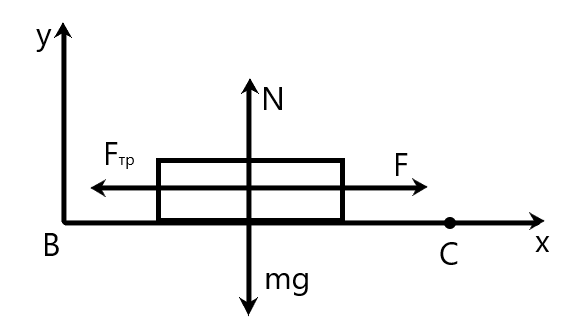
Еще раз проинтегрируем и получим:

Чтобы нарисовать мой график в программе, нужно повернуть его на , для этого воспользуемся формулой поворота координат:

В нашем случае координата «у» не изменяется, т. е. равна нулю, тогда получаем:

### **Участок ВС:**

Рассмотрим движение камня от точки В до точки С. Покажем силу тяжести G, действующую на тело, силу трения и силу F. Составим дифференциальное уравнение его движения:



*, где*

Разделим правую и левую часть равенства на :

Проинтегрируем, т. к. F зависит от t, то обозначим F как :

Найдем постоянные интегрирования:

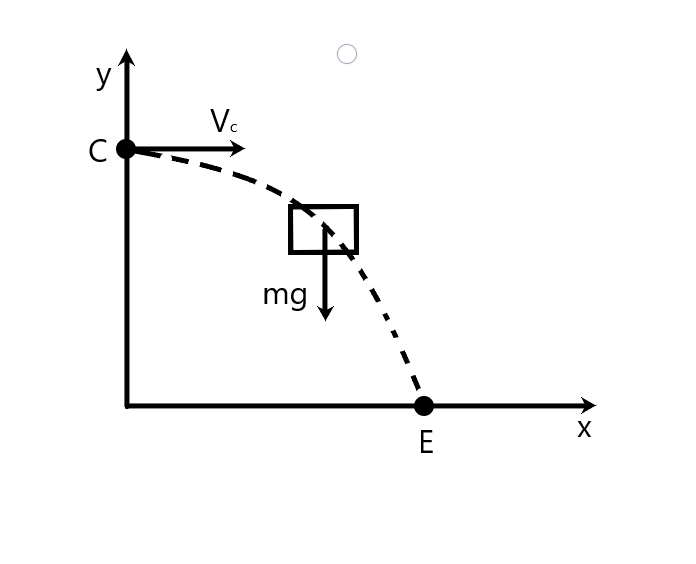
Подставив начальные условия , :

А при будет равна 0:

### 

### **Участок СE:**

Рассмотрим движение камня от точки C до точки E. Покажем силу тяжести G и скорость . Возьмём начало системы отсчета в точке С и направим ось Оу вниз. Составим дифференциальное уравнение его движения:



По оси X:

Т. к. на тела не действуют никакие силы. Разделим правую и левую часть на :

Найдем постоянную интегрирования , подставим начальные данные, :

:

При

Получаем уравнение движения по оси Ox:

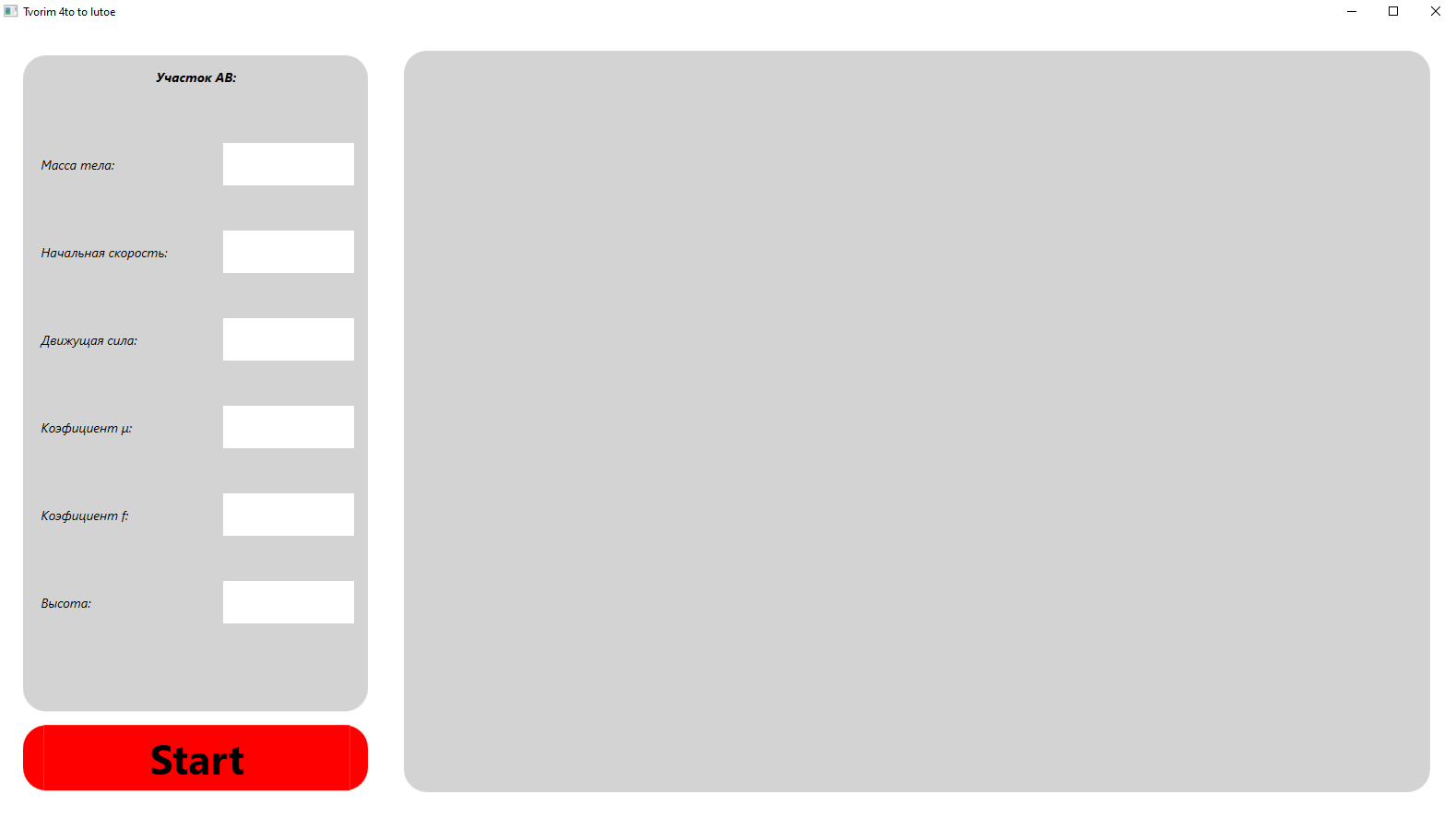
По оси Oy:

Разделим правую и левую часть на m:

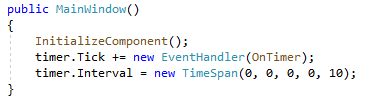
Получаем уравнения движения:

# **Описание программы**

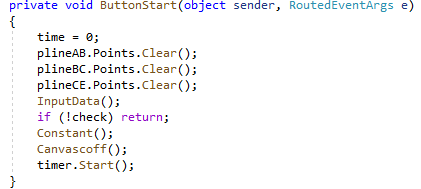
При запуске программы мы видим главное окно программы на котором: 6 строчек для ввода значений, кнопка Start, и зона для нашей траектории



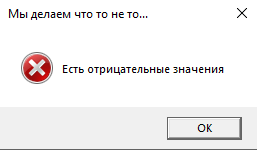
При запуске программы вызывается метод MainWindow, в котором запускается стандартный метод InitializeComponent, также у переменной timer появляется обработчик событий, который будет вызываться по истечению времени, дальше таймеру присваиваем интервал, равный 10 миллисекундам:



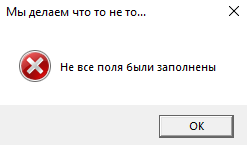
Дальше ожидается нажатие на кнопку Start, по нажатию, вызывается метод ButtonStart, который имеет вид:



В этом методе сначала осуществляется очистка данных, таких как траектория polyline, time. Переменным присваивается ноль. Дальше вызывается метод по вводу данных пользователем InputData(), также этот метод проверяет на “глупости” пользователя, например: отрицательные значения, не все заполненные поля ввода. Если мы попались на такую “глупость” то мы увидим маленькое окно с описанием ошибки.

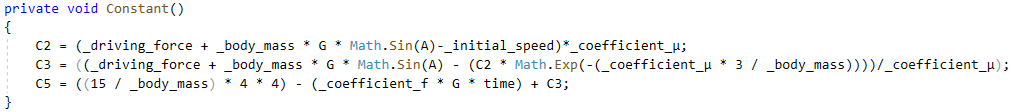


или



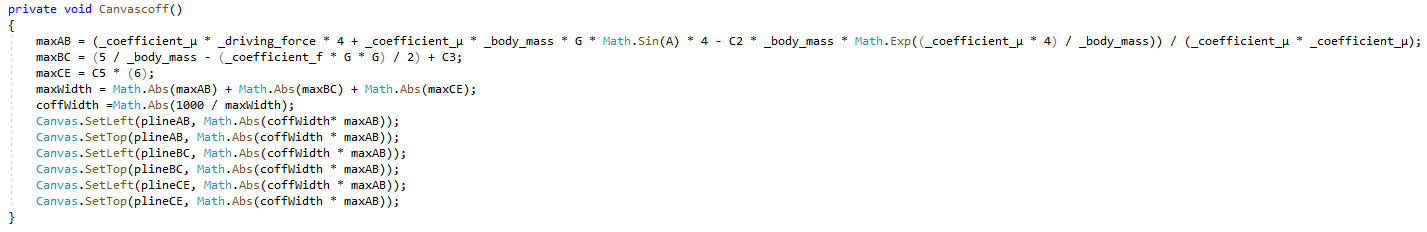
Сам код метода InputData() имеет вид:



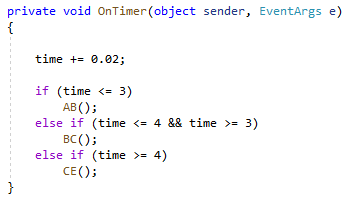
Если программа продолжила выполняться и не было никаких ошибок, то вызывается метод Constant(), который вычисляет 3 наших константы():

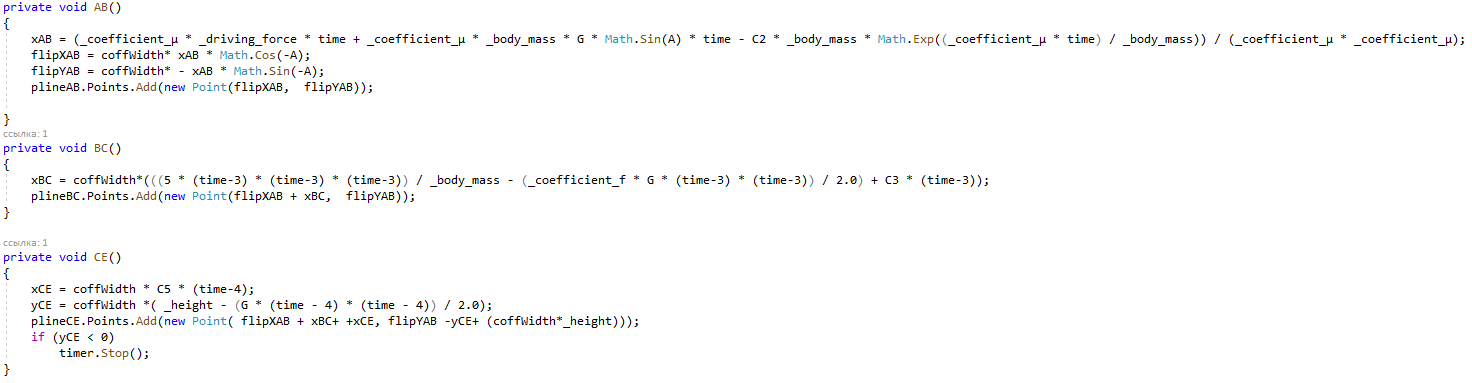
или в другом виде их можно записать

Далее вызывается метод Canvascoff(). Данный метод приблизительно вычисляет максимальные значения нашей траектории и далее выводит некоторый коэффициент масштабирования и выставляет начало координатной плоскости. С его помощью наша траектория постарается остаться и поместиться в центре окна.

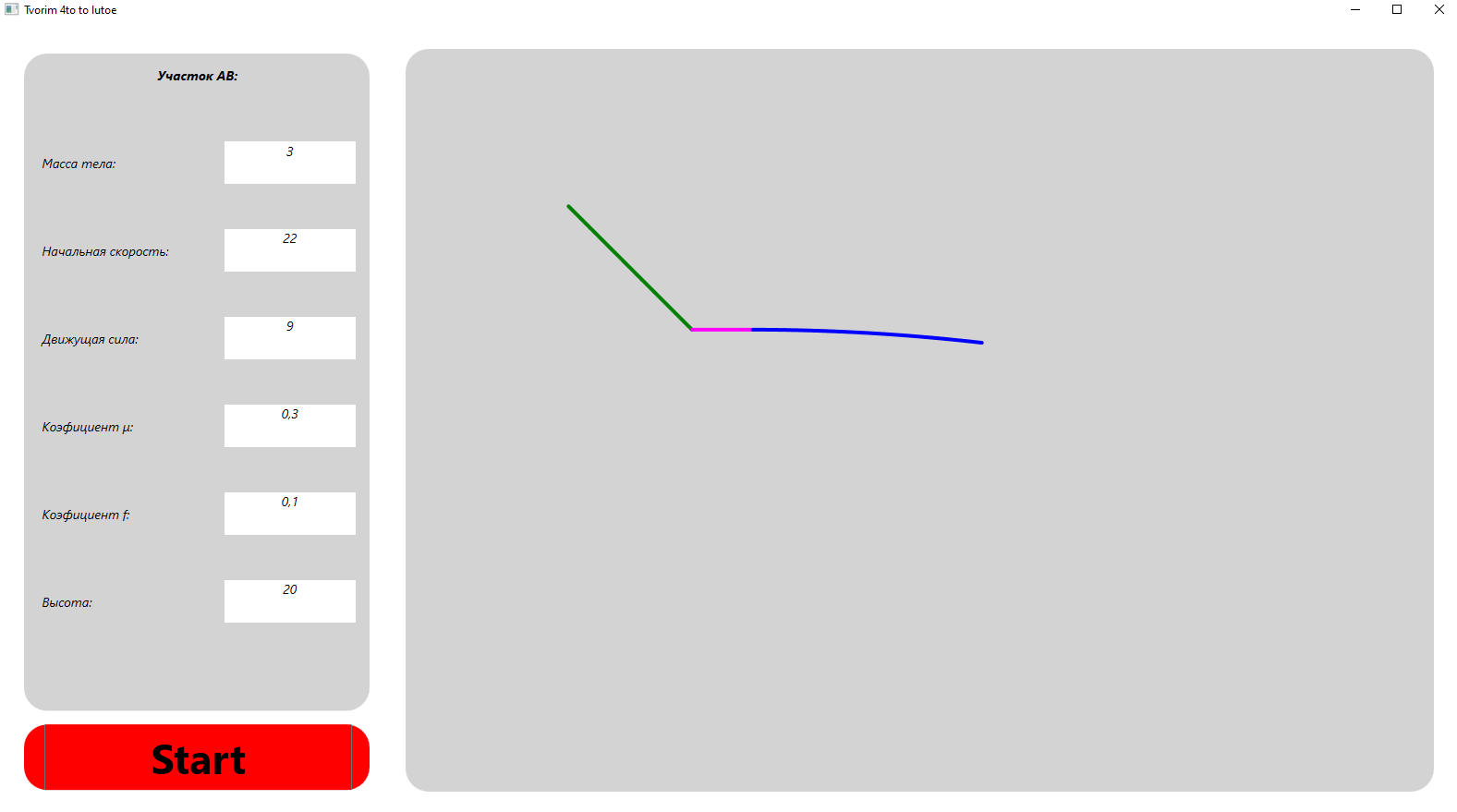


Дальше мы запускаем таймер, вызывается метод OnTimer который по течению времени вызывает методы: метод AB (в период времени от 0с до 3с), метод BC (в период времени от 3с до 4с), (в период времени от 3с до времени когда тело коснется “пола”).Эти методы в свою очередь производят расчеты координат по выведенным ранее формулам и по точкам рисуют нашу траекторию (у каждого метода свой цвет траектории(зеленый, розовый, синий соответственно). Так же метод CE производит остановку таймера.





В результате мы получаем траекторию:



# 

# 

# 

# **Приложение**

## **Файл MainWindow.xaml.сs**:

**using** **System**;

**using** **System.Windows**;

**using** **System.Windows.Controls**;

**using** **System.Windows.Threading**;

**namespace** **практика\_механика**

{

**public** **partial** **class** **MainWindow** : Window

{

**private** DispatcherTimer timer = **new** DispatcherTimer();

**private** **double** time; //наша переменная t

**private** **double** maxWidth; //максимальная ширина графика

**private** **double** coffWidth;//коэффициент увеличения/уменьшения графика

**private** **double** maxAB;//максимальное значение по отрезку AB

**private** **double** maxBC;//максимальное значение по отрезку BC

**private** **double** maxCE;//максимальное значение по отрезку CE

**private** **double** \_body\_mass; //масса тела

**private** **double** \_initial\_speed;// начальная скорость

**private** **double** \_driving\_force; //принудительная сила

**private** **double** \_coefficient\_μ; //коэффициент μ

**private** **double** \_coefficient\_f; //коэффициент f

**private** **double** \_height; //высота

**private** **double** C2;//константа 2

**private** **double** C3;//константа 3

**private** **double** C5;//константа 5

**private** **double** xAB;//координата х отрезка АВ

**private** **double** flipXAB;// измененная координата х отрезка АВ

**private** **double** flipYAB;//измененная координата у отрезка АВ

**private** **double** xBC;//координата х отрезка ВС

**private** **double** xCE;//координата х отрезка СЕ

**private** **double** yCE;//координата х отрезка СЕ

**private** **bool** check = **true**; //булевое значение для проверок

**const** **double** G = **9.80665**;// константа G

**const** **double** A = **0.25**\*Math.PI;//угол в 45 градусов

**public** **MainWindow**()

{

InitializeComponent();

timer.Tick += **new** EventHandler(OnTimer);

timer.Interval = **new** TimeSpan(**0**, **0**, **0**, **0**, **10**);

}

**private** **void** **InputData**()

{

check = **true**;

**if** (body\_mass.Text == "" | coefficient\_μ.Text == "" | driving\_force.Text == "" |

initial\_speed.Text == "" | coefficient\_f.Text=="" | height.Text=="")

{

MessageBox.Show("Не все поля были заполнены ", "Мы делаем что то не то...",

MessageBoxButton.OK, MessageBoxImage.Error);

check = **false**;

}

**else** **if** (check)

{

\_body\_mass = Convert.ToDouble(body\_mass.Text);

\_coefficient\_μ = Convert.ToDouble(coefficient\_μ.Text);

\_driving\_force = Convert.ToDouble(driving\_force.Text);

\_initial\_speed = Convert.ToDouble(initial\_speed.Text);

\_coefficient\_f = Convert.ToDouble(coefficient\_f.Text);

\_height = Convert.ToDouble(height.Text);

}

**if** (\_body\_mass<**0** | \_coefficient\_μ<**0** | \_driving\_force<**0** | \_initial\_speed<**0** | \_coefficient\_f<**0** | \_height<**0** )

{

MessageBox.Show("Есть отрицательные значения ", "Мы делаем что то не то...",

MessageBoxButton.OK, MessageBoxImage.Error);

check = **false**;

}

}

**private** **void** **Constant**()

{

C2 = (\_driving\_force + \_body\_mass \* G \* Math.Sin(A)-\_initial\_speed)\*\_coefficient\_μ;

C3 = ((\_driving\_force + \_body\_mass \* G \* Math.Sin(A) - (C2 \* Math.Exp(-(\_coefficient\_μ \* **3** / \_body\_mass))))/\_coefficient\_μ);

C5 = ((**15** / \_body\_mass) \* **4** \* **4**) - (\_coefficient\_f \* G \* time) + C3;

}

**private** **void** **Canvascoff**()

{

maxAB = (\_coefficient\_μ \* \_driving\_force \* **4** + \_coefficient\_μ \* \_body\_mass \* G \* Math.Sin(A) \* **4** - C2 \* \_body\_mass \* Math.Exp((\_coefficient\_μ \* **4**) / \_body\_mass)) / (\_coefficient\_μ \* \_coefficient\_μ);

maxBC = (**5** / \_body\_mass - (\_coefficient\_f \* G \* G) / **2**) + C3;

maxCE = C5 \* (**6**);

maxWidth = Math.Abs(maxAB) + Math.Abs(maxBC) + Math.Abs(maxCE);

coffWidth =Math.Abs(**1000** / maxWidth);

Canvas.SetLeft(plineAB, Math.Abs(coffWidth\* maxAB));

Canvas.SetTop(plineAB, Math.Abs(coffWidth \* maxAB));

Canvas.SetLeft(plineBC, Math.Abs(coffWidth \* maxAB));

Canvas.SetTop(plineBC, Math.Abs(coffWidth \* maxAB));

Canvas.SetLeft(plineCE, Math.Abs(coffWidth \* maxAB));

Canvas.SetTop(plineCE, Math.Abs(coffWidth \* maxAB));

}

**private** **void** **AB**()

{

xAB = (\_coefficient\_μ \* \_driving\_force \* time + \_coefficient\_μ \* \_body\_mass \* G \* Math.Sin(A) \* time - C2 \* \_body\_mass \* Math.Exp((\_coefficient\_μ \* time) / \_body\_mass)) / (\_coefficient\_μ \* \_coefficient\_μ);

flipXAB = coffWidth\* xAB \* Math.Cos(-A);

flipYAB = coffWidth\* - xAB \* Math.Sin(-A);

plineAB.Points.Add(**new** Point(flipXAB, flipYAB));

}

**private** **void** **BC**()

{

xBC = coffWidth\*(((**5** \* (time-**3**) \* (time-**3**) \* (time-**3**)) / \_body\_mass - (\_coefficient\_f \* G \* (time-**3**) \* (time-**3**)) / **2.0**) + C3 \* (time-**3**));

plineBC.Points.Add(**new** Point(flipXAB + xBC, flipYAB));

}

**private** **void** **CE**()

{

xCE = coffWidth \* C5 \* (time-**4**);

yCE = coffWidth \*( \_height - (G \* (time - **4**) \* (time - **4**)) / **2.0**);

plineCE.Points.Add(**new** Point( flipXAB + xBC+ +xCE, flipYAB -yCE+ (coffWidth\*\_height)));

**if** (yCE < **0**)

timer.Stop();

}

**private** **void** **OnTimer**(**object** sender, EventArgs e)

{

time += **0.02**;

**if** (time <= **3**)

AB();

**else** **if** (time <= **4** && time >= **3**)

BC();

**else** **if** (time >= **4**)

CE();

}

**private** **void** **ButtonStart**(**object** sender, RoutedEventArgs e)

{

time = **0**;

plineAB.Points.Clear();

plineBC.Points.Clear();

plineCE.Points.Clear();

InputData();

**if** (!check) **return**;

Constant();

Canvascoff();

timer.Start();

}

}

}

## **Файл MainWindow.xaml:**

<Window x:Class="практика\_механика.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:практика\_механика"

mc:Ignorable="d"

x:Name="window" Title="Tvorim 4to to lutoe" Height="900" Width="1600" Background="White">

<Grid Margin="0,0,0,0">

<Grid.ColumnDefinitions>

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

<ColumnDefinition />

</Grid.ColumnDefinitions>

<Grid.RowDefinitions>

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

<RowDefinition />

</Grid.RowDefinitions>

<Border x:Name="regionAB" Grid.Row="1" Grid.Column="1" Grid.ColumnSpan="16" Grid.RowSpan="27" CornerRadius="25" Background="LightGray" Margin="5,5,5,10">

<Grid >

<Grid.RowDefinitions>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

<RowDefinition Height="45\*"></RowDefinition>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

<ColumnDefinition></ColumnDefinition>

</Grid.ColumnDefinitions>

<Label Content="Участок АВ:" Grid.Column="9" Grid.ColumnSpan="8" Grid.Row="0" Grid.RowSpan="1" HorizontalAlignment="Center" VerticalAlignment="Center" FontSize="14" FontStyle="Italic" FontWeight="Bold" Margin="9,0,7,0" Width="97"/>

<Label Content="Масса тела:" Grid.Column="1" Grid.ColumnSpan="7" Grid.Row="2" Grid.RowSpan="1" HorizontalAlignment="Left" VerticalAlignment="Center" FontSize="14" FontStyle="Italic" Width="90" />

<Label Content="Начальная скорость:" Grid.Column="1" Grid.ColumnSpan="11" Grid.Row="4" Grid.RowSpan="1" HorizontalAlignment="Left" VerticalAlignment="Center" FontSize="14" FontStyle="Italic" Width="147" />

<Label Content="Движущая сила:" Grid.Column="1" Grid.ColumnSpan="8" Grid.Row="6" Grid.RowSpan="1" HorizontalAlignment="Left" VerticalAlignment="Center" FontSize="14" FontStyle="Italic" Width="114" />

<Label Content="Коэффициент μ:" Grid.Column="1" Grid.ColumnSpan="9" Grid.Row="8" Grid.RowSpan="1" HorizontalAlignment="Left" VerticalAlignment="Center" FontSize="14" FontStyle="Italic" Width="119" Margin="0,10" />

<Label Content="Коэффициент f:" Grid.Column="1" Grid.ColumnSpan="8" Grid.Row="10" Grid.RowSpan="1" HorizontalAlignment="Left" VerticalAlignment="Center" FontSize="14" FontStyle="Italic" Width="114" Margin="0,9" />

<Label Content="Высота:" Grid.Column="1" Grid.ColumnSpan="5" Grid.Row="12" Grid.RowSpan="1" HorizontalAlignment="Left" VerticalAlignment="Center" FontSize="14" FontStyle="Italic" Width="64" />

<TextBox x:Name="body\_mass" Grid.Column="15" Grid.Row="2" Grid.RowSpan="1" BorderBrush="LightGray" MaxLength="10" MaxLines="10" Background="White" FontSize="14" FontStyle="Italic" HorizontalContentAlignment="Center" VerticalAlignment="Center" Grid.ColumnSpan="10" Height="48" />

<TextBox x:Name="initial\_speed" Grid.Column="15" Grid.Row="4" Grid.RowSpan="1" BorderBrush="LightGray" MaxLength="10" MaxLines="10" Background="White" FontSize="14" FontStyle="Italic" HorizontalContentAlignment="Center" VerticalAlignment="Center" Grid.ColumnSpan="10" Height="48" />

<TextBox x:Name="driving\_force" Grid.Column="15" Grid.Row="6" Grid.RowSpan="1" BorderBrush="LightGray" MaxLength="10" MaxLines="10" Background="White" FontSize="14" FontStyle="Italic" HorizontalContentAlignment="Center" VerticalAlignment="Center" Grid.ColumnSpan="10" Height="48" />

<TextBox x:Name="coefficient\_μ" Grid.Column="15" Grid.Row="8" Grid.RowSpan="1" BorderBrush="LightGray" MaxLength="10" MaxLines="10" Background="White" FontSize="14" FontStyle="Italic" HorizontalContentAlignment="Center" VerticalAlignment="Center" Grid.ColumnSpan="10" Height="48" />

<TextBox x:Name="coefficient\_f" Grid.Column="15" Grid.Row="10" Grid.RowSpan="1" BorderBrush="LightGray" MaxLength="10" MaxLines="10" Background="White" FontSize="14" FontStyle="Italic" HorizontalContentAlignment="Center" VerticalAlignment="Center" Grid.ColumnSpan="10" Height="48" />

<TextBox x:Name="height" Grid.Column="15" Grid.Row="12" Grid.RowSpan="1" BorderBrush="LightGray" MaxLength="10" MaxLines="10" Background="White" FontSize="14" FontStyle="Italic" HorizontalContentAlignment="Center" VerticalAlignment="Center" Grid.ColumnSpan="10" Height="48" />

</Grid>

</Border>

<Border Grid.Row="1" Grid.RowSpan="30" Grid.Column="18" Grid.ColumnSpan="47" CornerRadius="25" Background="LightGray" Margin="10,0,5,0" Height="804" VerticalAlignment="Top">

<Grid >

<Grid.RowDefinitions>

<RowDefinition Height="800"></RowDefinition>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="1100"/>

</Grid.ColumnDefinitions>

<Canvas x:Name="canvas" HorizontalAlignment="Center" VerticalAlignment="Center" Margin="16,10,0,10" Height="781" Width="1085" RenderTransformOrigin="0.49,0.554">

<Polyline x:Name="plineAB" Stroke="Green" StrokeThickness="4" StrokeEndLineCap="Round" StrokeStartLineCap="Round" />

<Polyline x:Name="plineBC" Stroke="Fuchsia" StrokeThickness="4" StrokeEndLineCap="Round" StrokeStartLineCap="Round" ></Polyline>

<Polyline x:Name="plineCE" Stroke="Blue" StrokeThickness="4" StrokeEndLineCap="Round" StrokeStartLineCap="Round"></Polyline>

</Canvas>

</Grid>

</Border>

<Border Grid.Row="28" Grid.RowSpan="3" Grid.Column="1" Grid.ColumnSpan="16" CornerRadius="25" Background="Red" Margin="5">

<Button x:Name="buttonStart" Content="Start" FontWeight="Bold" FontSize="45" Click="ButtonStart" Background="Red" VerticalAlignment="Center" Margin="22,0,19,0" RenderTransformOrigin="0.5,0.434" Height="73" />

</Border>

</Grid>

</Window>

# **Подведение итогов работы**

Задача поставленная передо мной оказалась непростой, но очень интересной. Благодаря ей я улучшил свои знания в «Теоретической механике» и программировании на языках C# и Xaml. Также мной было освоено написание и оформление самой работы.

# **Список литературы**

1. Бухгольц Н.Н. Основной курс теоретической механики.
2. Ч. Петцольд «Microsoft Windows Presentation Foundation: базовый курс» https://flylib.com/books/en/4.266.1/