**Міністерство освіти і науки України**

**Чернівецький національний університет**

**імені Юрія Федьковича**

Інститут фізико-технічних та комп’ютерних наук

Кафедра інформаційних технологій та комп’ютерної фізики

**Курсова робота**

**на тему**

**Створення середовища для моделювання різних**

**процесів у фізиці**

Студента 3 курсу 411 групи

спеціальності «Фізика та астрономія»

Кас’янчука О. І.

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів\_\_\_\_Оцінка ECTS\_\_\_\_\_

Зав. Кафедри ІТКФ\_\_\_\_\_\_\_\_Борча М.Д.

Чернівці – 2021

**АНОТАЦІЯ**

Швидкий розвиток науки, техніки та технологій дає можливість застосовувати під час проведення занять з фізики та астрономії нових інформаційних технологій, що дозволяє підвищити інтерес до вивчення предметів природничо-математичного циклу, активізувати їх пізнавальну діяльність, сприяє формуванню наукового світогляду.

Щорічне зростання кількості інформації, що має бути засвоєна студентами, потребує від викладачів ВНЗ розробки і впровадження таких засобів, форм і методів навчання, які зменшують непродуктивні витрати часу, прискорюють темп і забезпечують високу якість навчання [4].

Тому комп’ютерне моделювання реальних та віртуальних фізичних процесів дає можливість більш доступного розкриття фізичних явищ та закономірностей особливо в умовах сучасного світу, зокрема з появою дистанційного навчання.

**Зміст**

Вступ. 4

Розділ 1. Використання комп’ютерних технологій

у навчальному процесі. 5

§1.1. Роль інформаційно-комп’ютерних технологій

у навчальному процесі 5

§1.2. Деякі зауваження до комп’ютерного

моделювання фізичних процесів 6

§1.3. Фізичні моделі маятників 7

Розділ 2. Віртуальні фізичні моделі маятників 11

Порядок створення веб-сайту 11

2.1.1. Розробка робочого простору у веб-сайті 11

2.1.2. Створення моделі простого математичного маятника 12

2.1.2. Створення моделі подвійного маятника 15

2.1.3. Створення моделі пружинного маятника 19

Висновки ……………………………………………………… ..23

Список літератури………………………………………………….. 24

**Вступ**

З швидким розвитком сучасного світу, зокрема технологій, програмних продуктів та сучасної техніки, а також використання дистанційного навчання як одного з виду форм навчання, є необхідним використання у навчальному процесі сучасних технічних засобів.

Використання новітніх комп’ютерних технологій роблять даний напрямок актуальним та привабливим при читанні лекцій, проведенні практичних та лабораторних занять із всіх дисциплін, у тому числі і з курсу загальної фізики та астрономії.

Крім того, використання сучасних технічних гаджетів засновано на всебічному врахуванні психологічних та фізіологічних можливостей організму людини, дозволяє підвищити вплив викладача на аудиторію та підсилити ефективність засвоєння і творчого відтворення інформації студентами, покращити якість навчального процесу та студентської наукової роботи [4].

**Розділ 1. Використання комп’ютерних технологій**

**у навчальному процесі**

**§1.1. Роль інформаційно-****комп’ютерних технологій у навчальному процесі**

Інформаційно-комп’ютерні технології – це досить потужні механізми, які мають багато можливостей. Вони проникли в усі сфери життя, формують нові системи потреб, спосіб життя, демократизують процес навчання, роблять процес пізнання творчим, стимулюють заняття самоосвітою [6].

Традиційні форми навчання поступово змінюються новітніми технологіями, спрямованими на формування навичок та умінь, що відповідають вимогам суспільства.

Сучасні вимоги суспільства до освіти примушують фахівців у багатьох країнах світу переглянути якість і рівень шкільної освіти, що зумовило необхідність її реформування. Змінюються цілі та завдання, що постали перед сучасною освітою в інформаційному суспільстві, поступово на зміну традиційній системі навчання приходить особистісно-орієнтована, традиційні методи змінюються інноваційними, що передбачають зміщення акцентів у навчальній діяльності, її спрямування на інтелектуальний розвиток учнів/студентів за рахунок зменшення долі репродуктивної діяльності. Навчальний процес сьогодні повинен бути орієнтований на особистість учня/студента і враховувати його індивідуальні особливості та здібності.

Але вони не заміняють викладача, а можуть бути тільки інструментом у руках викладача, причому таким інструментом, який є потужним у своїх функціях, і має дуже великий ресурс використання.Отже, і ставитися до цих технологій треба лише як до інструменту, зробленого для полегшення праці, а не до генератора команд та ідей.

**§1.2. Деякі зауваження до комп’ютерного**

**моделювання фізичних процесів**

На даний час існує чимало готових програмних продуктів, які викладачі можуть використовувати під час проведення як лекційних, практичних так лабораторних занять. Але сучасний світ вимагає від викладача використовувати власні методичні розробки, які напрямлені на розвиток інтересу не тільки до свого предмету, але і надати учням/студентам можливість застосувати свої знання предмету разом із програмуванням. Це дає нові можливості для створення віртуального простору, в якому стає можливим демонстрація процесів, що не можливо реалізувати в реальних умовах, або для покращення та поглиблення знань вже існуючих явищ. Крім того, це дає можливість працювати на стику двох напрямків - фізики та програмування, що має зацікавити студентів для подальшого навчання в умовах як очного так і дистанційного.

Основними етапами поєднання таких двох напрямків є:

1. Визначення та формулювання теми дослідження.
2. Визначення об’єкта, предмета, мети та завдань дослідження.
3. Ретельна робота з різними джерелами інформації, щодо обґрунтування явища, яке має запрограмуватися та знаходження коректного математичного та фізичного запису даного явища.
4. Вивчення попередніх робіт та існуючих програм. Робота зі спеціалізованими та тематичними сайтами.
5. Вибір мови програмування.
6. Створення і використання персональної програми, сайту і т. д.

Розглянемо коротко деякі кроки для вирішення цієї проблеми. Для цього необхідно застосовувати методи математичного моделювання.

Це відбувається в декілька етапів [5]:

1. Побудова для досліджуваного об'єкта фізичної моделі. Формулюючи допущення, умови і межі застосування створеної фізичної моделі, її описують системою рівнянь і таким чином замість фізичної створюють математичну модель.
2. Розробляється метод розрахунку сформульованої математичної задачі у вигляді обчислювального алгоритму, що складається із сукупності математичних формул і послідовності їх застосування.
3. Розробляється алгоритм і програма рішення задачі на комп'ютері.
4. На комп'ютері проводяться розрахунки та їх результати представляються у вигляді деякої цифрової інформації.
5. Проводиться обробка отриманих результатів розрахунків і робляться відповідні висновки.

Такого роду робота може проводитися спільно з викладачем та студентом. Це покращує розуміння та комунікацію студента та викладача, а також сприяє розвитку вмінь і навичок застосовувати набуті на заняттях знання в майбутньому.

Але, слід зауважити, що такого роду дії потребують високої кваліфікації викладача та високого рівня знань студента.

**§1.3. Фізичні моделі маятників.**

Розглянемо деякі приклади застосування програмування для опису деяких фізичних явищ на прикладі моделей математичного та пружинного маятників.

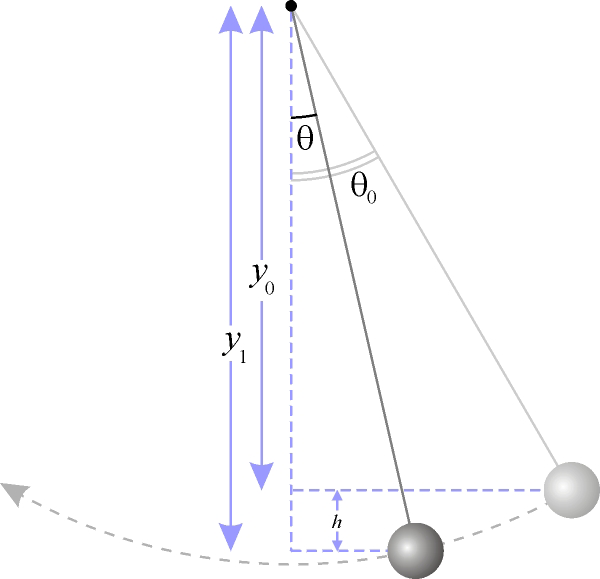
Спочатку коротко розглянемо, що являють собою ці маятники.

* + 1. **Математичний маятник**.

***Математичний маятник*** - теоретична модель маятника, в якій матеріальна точка масою *m* підвішена на невагомій нерозтяжній нитці або легкому стрижні, і здійснює коливання у вертикальній площині під впливом сили тяжіння (рис.1).

Математичний маятник являє собою граничний випадок фізичного маятника, вся маса якого зосереджена в його центрі інерції. Модель нехтує розмірами тіла, деформацією підвісу та [тертям](https://uk.wikipedia.org/wiki/%D0%A1%D0%B8%D0%BB%D0%B0_%D1%82%D0%B5%D1%80%D1%82%D1%8F) в точці підвісу. Зазвичай розглядають коливання маятника в одній площині. При малих відхиленнях нитки від положення рівноваги коливання математичного маятника є гармонічними з періодом коливань:

.



**Рис.1**. Модель математичного маятника [7]

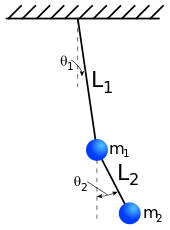
Якщо початкове відхилення є великим, то коливання маятника періодичні, але не гармонічні. Більш детально коливання маятника можна розглянути у літературі [1, 2, 3].

На основі моделі простого матиматичного маятника ми можемо вирахувати прискорення вільного падіння. Для цього знаючи формулу періоду коливання матиматичного маятника

* + 1. **Подвійний маятник**

У фізиці і математиці, у галузі динамічних систем, подвійний маятник це маятник з іншим маятником прикріпленим до його кінця, і є простою фізичною системою, яка проявляє різноманітну динамічну поведінку зі значною залежністю від початкових умов (рис.2).

Рух маятника керується пов'язаними [звичайними диференціальними рівняннями](https://uk.wikipedia.org/wiki/%D0%97%D0%B2%D0%B8%D1%87%D0%B0%D0%B9%D0%BD%D1%96_%D0%B4%D0%B8%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D1%96%D0%B0%D0%BB%D1%8C%D0%BD%D1%96_%D1%80%D1%96%D0%B2%D0%BD%D1%8F%D0%BD%D0%BD%D1%8F). Для деяких [енергій](https://uk.wikipedia.org/wiki/%D0%95%D0%BD%D0%B5%D1%80%D0%B3%D1%96%D1%8F) його рух є [хаотичним](https://uk.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BE%D1%80%D1%96%D1%8F_%D1%85%D0%B0%D0%BE%D1%81%D1%83). Можна розглядати декілька варіантів подвійних маятників; два члени можуть бути однакові чи різні завдовжки та за вагою, вони можуть бути [простими маятниками](https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%8F%D1%82%D0%BD%D0%B8%D0%BA) або [маятниками](https://uk.wikipedia.org/wiki/%D0%A4%D1%96%D0%B7%D0%B8%D1%87%D0%BD%D0%B8%D0%B9_%D0%BC%D0%B0%D1%8F%D1%82%D0%BD%D0%B8%D0%BA) і рух може бути у трьох вимірах або обмежений вертикальною площиною [1, 2, 3].



**Рис.2** Приклад простого подвійного маятника [8].

* + 1. **Пружинний маятник**

**Пружинний маятник** – це тіло масою *m* закріплене на пружині жорсткістю *k*, і яке може рухатися без тертя під дією сили пружності [1, 2, 3].

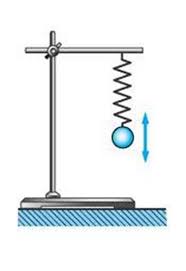
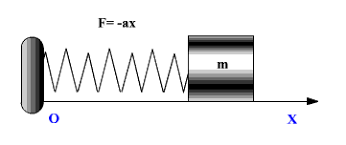
а) б)

Рис. 3. Приклади пружинних маятників [9].

Коливання такої системи можуть відбуватися як в горизонтальній так і *вертикальних* площинах (рис.3, а, б). Період коливань такого маятника залежить запишеться як:

В нашому випадку замість пружини ми обрали резинку. Коливання будуть здійснюватися у вертикальній площині.

Коефіцієнт жорсткості в нашому випадку буде розраховуватися за формулою:

,

де *E* – Модуль Юнга, *S* – площа поперечного перерізу, *l0* – початкова довжина резинки.

Детальніше з особливостями коливань маятників можна ознайомитися з загального курсу фізики: Механіка [1, 2, 3].

**Розділ 2. Віртуальні фізичні моделі маятників**

Наш експеримент ґрунтується на створенні програми, яка дозволить будь-кому виконувати лабораторні та практичні завдання із загального курсу фізики в дистанційному режимі, завдяки віртуальним фізичним моделям.

Для моделювання фізичних процесів та створення програмного середовища було вибрано мову програмування «JavaScript» стандарту «ES6». Також було застосовано наступні допоміжні інструменти:

* **«HTML»** –для написання структури веб-сайту;
* **«СSS»** – для стилізації цієї структури;
* **«Canvas API»,** як допоміжний інструмент накладання анімації та створення геометричних форм.

Ключовий вибір мови програмування «JavaScript»**,** полягає у тому, що вона є однією з трьох основ створення веб-програм. Це дозволяє відкрити програму в будь-якому браузері на будь-якому пристрої.

**Порядок створення веб-сайту**

**2.1.1. Розробка робочого простору у веб-сайті**

Першим кроком було створення «HTML» файлу в якому ми підключили наперед розроблені стилі «CSS» та «JavaScript» файл .Всі види фізичної симуляції було прийнято писати в <canvas></canvas> елементі. Наступним кроком була розробка бокового меню, яке можна відкрити натиснувши на іконку  в правому верхньому куті. Це меню є достатньо важливою частиною програми. Воно являє собою вузол між всіма підпрограмами фізичної симуляції. В даному випадку ми маємо дві готові підпрограми. Вони симулюють коливання простого, подвійного та «пружного» маятників.

Детальна робота цього меню полягає в заміні значення стилю «canvas{*display*: none;}–приховує <canvas></canvas> елемент», на «canvas {*display*: block;}– показує як блок».

**2.1.2. Створення моделі простого математичного маятника**

Для створення моделі простого математичного маятника, необхідно на початку задати всі змінні та константи задачі *let*-змінні *const*-постійні

*let* x, y; //Координати кола

*let* ang = 0; //Кут маятника

*const* len = 350; //Довжина лінії

*const* r = 40; //Радіус кола

*let* aVel = 0; //Кутова швидкість

*let* aAcc; //Кутове прискорення

*const* gravity = 9.81; //Прискорення вільного падіння

*const* damp = 0.996; //Коефіцієнт затухання

Задавши змінні ми створили функцію для малювання під назвою draw()

*const* draw = *function* () {

// Для стирання попереднього малюнку;

    ctx.clearRect(0, 0, innerWidth, innerHeight);

//Малюємо точку

    ctx.beginPath();

    ctx.arc(600, 150, 5, 0, 2 \* Math.PI, false);

    ctx.fillStyle = "#000";

    ctx.fill();

    ctx.stroke();

// Через кут визначаємо положення маятника

    x = 600 + len \* Math.sin(ang);

    y = 150 + len \* Math.cos(ang);

//Малюємо лінію

    ctx.beginPath();

    ctx.moveTo(600, 150);

    ctx.lineTo(x, y);

    ctx.stroke();

//Малюємо коло

    ctx.beginPath();

    ctx.arc(x, y, r, 0, 2 \* Math.PI, false);

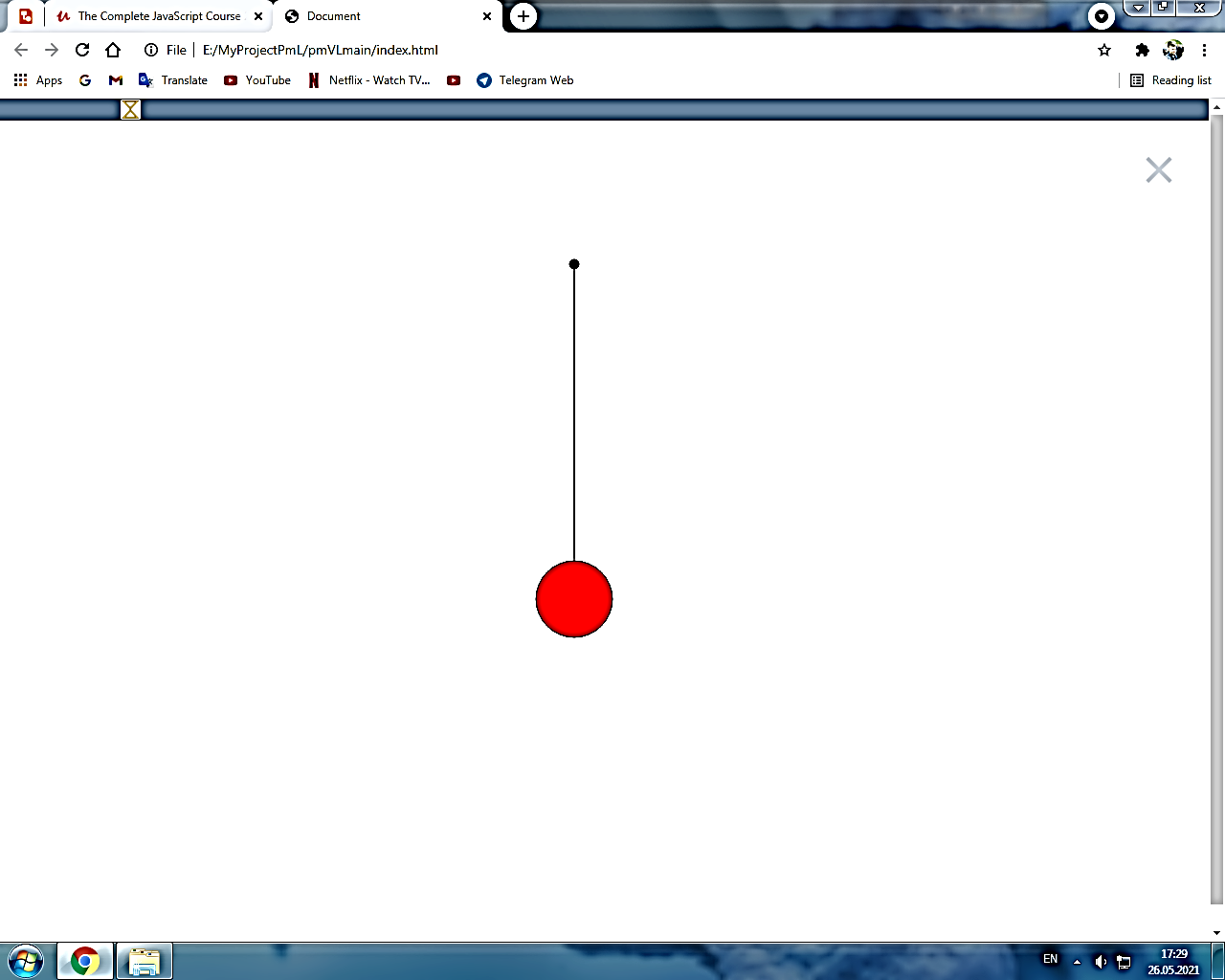
    ctx.fillStyle = "red";

    ctx.fill();

    ctx.stroke();

  };

Викликавши функцію та перейшовши на сайт ми бачимо графічний вигляд математичного маятника (рис.4).



**Рис.4.** Модель математичного маятника

Тепер коли ми запрограмували функцію для малювання залишилось тільки створити анімацію і на цьому етапі ми враховуємо фізично-математичний опис маятника. [10]

В функції для малювання ми виразили наші координати через кут *ang* назвемо його *θ* за даними формулами



Також ми використали рівняння руху в якому знаходимо кутове прискорення



Анімацію маятника ми розробили у вигляді функції *animate*(), вигляд якої представлений наступним чином:

*const* animate = *function* () {

      requestAnimationFrame(animate); // Працює за принципом нескінченого циклу який викликає функцію animate()

      draw();//Викликаємо функцію для малювання

      aAcc = (-gravity / len) \* Math.sin(ang); //Рівняння руху

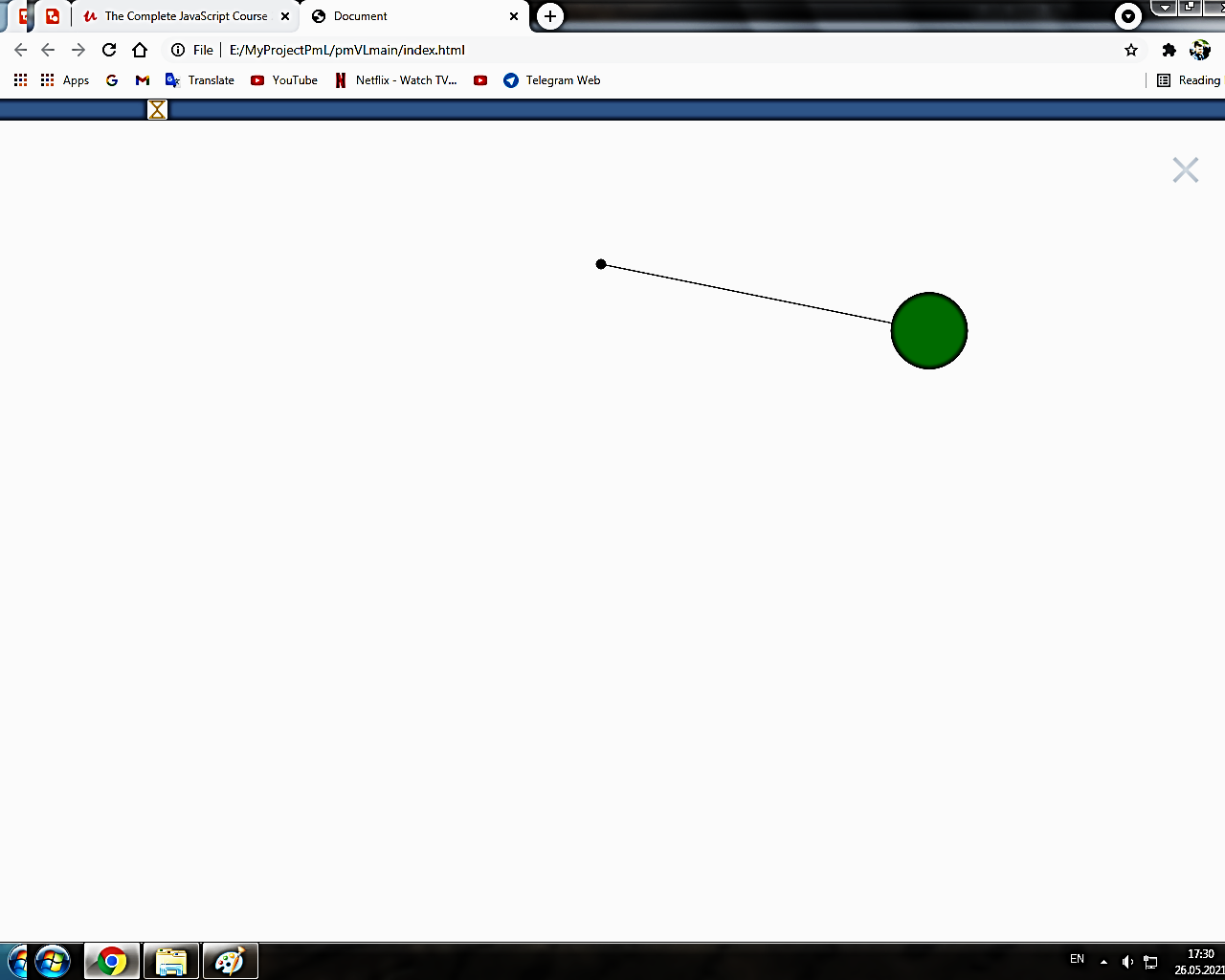
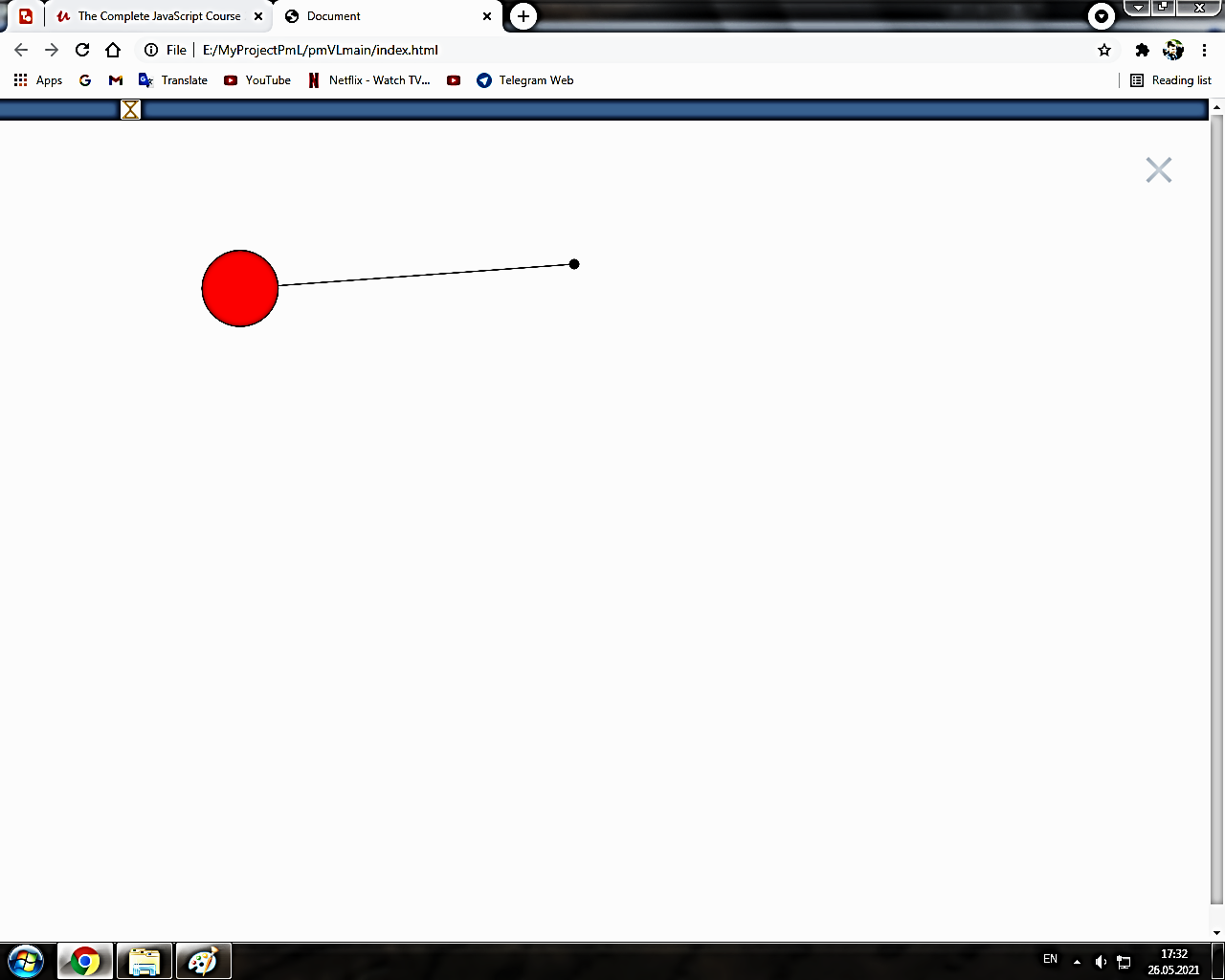
      aVel += aAcc; //Додаємо кут. швид. до кут .приск. та зберігаємо кут. швид.

      ang += aVel; //Так само робимо з кутом

      aVel \*= damp; //кут. швидкість множимо на коеф. затух. Зберігаємо швидкість

  };

За цією програмою ми отримали віртуальну модель коливань простого математичного маятника.

**Рис.5**. Віртуальна модель математичного маятника (анімація).

Тепер розробимо вікно у якому ми зможемо змінювати параметри маятника такі як довжина і прискорення вільного падіння для цього на

На основі моделі ми можемо вирахувати прискорення вільного падіння. Для цього з формули періоду коливання математичного маятника

.

Виразимо g

.

У формулу входить довжина маятник l та період коливання T. Довжина маятника задана, а період ми знайдемо за формулою

**2.1.2. Створення моделі подвійного маятника**

Застосовуючи алгоритм створення математичного маятника, була розроблена програма для опису коливань подвійного маятника. На початку роботи так само задаються змінні та константи.

*let* x1, y1, x2, y2; // Координати колів

*const* l1 = 200; //Довжина верхньої лінії

*const* l2 = 200; //Довжина нижньої лінії

*const* m1 = 20; //Маса та радіус верхньої лінії

*const* m2 = 20; //Маса та радіус нижньої лінії

*let* ang1 = 0; //Кут верхнього маятника

*let* ang2 = 0; //Кут нижнього маятника

*let* a1V = 0; //Швидкість верхнього

*let* a2V = 0; //Швидкість нижнього

*let* a1A = 0; //Кутове прискорення верхнього

*let* a2A = 0; //Кутове прискорення нижнього

*const* g = 0.8; //Прискорення вільного падіння вибрано для зручності

*const* damp1 = 0.996; //Коефіцієнти затухання

*const* damp2 = 0.996;

Цього разу нам потрібні координати верхнього та нижнього кіл, а тому нам необхідно знайти кутове прискорення обох маятників, щоб задати природній рух маятників. Але спершу було розроблено функцію для малювання draw()

*const* draw = *function* () {

//Очищує екран і малю закріплену точку

    ctx.clearRect(0, 0, innerWidth, innerHeight);

    ctx.beginPath();

    ctx.arc(600, 150, 5, 0, 2 \* Math.PI, false);

    ctx.fillStyle = "#000";

    ctx.fill();

    ctx.stroke();

    //Переводить кут в координати першого кола

    x1 = 600 + l1 \* Math.sin(ang1);

    y1 = 150 + l1 \* Math.cos(ang1);

    //Малює лінії від точки до першого кола

    ctx.beginPath();

    ctx.moveTo(600, 150);

    ctx.lineTo(x1, y1);

    ctx.stroke();

    //Переводить кут в координати другого кола

    x2 = x1 + l2 \* Math.sin(ang2);

    y2 = y1 + l2 \* Math.cos(ang2);

    //малює лінію від першого до другого кола

    ctx.beginPath();

    ctx.moveTo(x1, y1);

    ctx.lineTo(x2, y2);

    ctx.stroke();

    //Малює перше коло

    ctx.beginPath();

    ctx.arc(x1, y1, m1, 0, 2 \* Math.PI, false);

    ctx.fillStyle = "red";;

    ctx.fill();

    ctx.stroke();

    //Малюємо друге коло

    ctx.beginPath();

    ctx.arc(x2, y2, m2, 0, 2 \* Math.PI, false);

    ctx.fillStyle = "red";

    ctx.fill();

    ctx.stroke();

  };

Тепер ми використали рівняння в якому знаходимо кутове прискорення, щоб задати природній рух маятників [11]:





Для того щоб змогти коректно запрограмувати рівняння нам потрібно було розбити його на частини і створити функції. в яких ми ці вирази порахували [12]:

  // Задаємо параметри для функції і повертаємо пораховані 4 частини рівняння та знаменник

*const* getAng1AccPart = *function* (*ang1*, *ang2*, *a1V*, *a2V*) //Наші параметри {

*let* prt1\_1 = -g \* (2 \* m1 + m2) \* Math.sin(*ang1*);

*let* prt1\_2 = -m2 \* g \* Math.sin(*ang1* - 2 \* *ang2*);

*let* prt1\_3 = -2 \* Math.sin(*ang1* - *ang2*) \* m2;

*let* prt1\_4 = *a2V* \* *a2V* \* l2 + *a1V* \* *a1V* \* l1 \* Math.cos(*ang1* - *ang2*);

*let* den1 = l1 \* (2 \* m1 + m2 - m2 \* Math.cos(2 \* *ang1* - 2 \* *ang2*));

//Створюєм масив де зберігається всі 4 частини та знаменник

*let* arrExpr = [prt1\_1, prt1\_2, prt1\_3, prt1\_4, den1];

    return arrExpr; //Повертаємо пораховані чотири частини

  };

//Аналогічно створюємо функцію getAng2AccPart() для другого виразу

Створивши функції ми помістили їх у функцію для анімації маятника animate()

//Задаємо змінні для зберігання частин рівняння

*let* arrExpr1;

*let* arrExpr2;

*const* animate = *function* () {

      requestAnimationFrame(animate);

      draw();

      arrExpr1 = getAng1AccPart(ang1, ang2, a1V, a2V); //Присвоюємо частини рівн.

//Повністю вираховуємо рівняння кутових прискорень

      a1A =

        (arrExpr1[0] + arrExpr1[1] + arrExpr1[2] \* arrExpr1[3]) / arrExpr1[4];

      arrExpr2 = getAng2AccPart(ang1, ang2, a1V, a2V);

      a2A =

        (arrExpr2[0] \* (arrExpr2[1] + arrExpr2[2] + arrExpr2[3])) / arrExpr2[4];

//Аналогічна з простим маятником послідовність присвоєнь

a1V += a1A;

      a2V += a2A;

      ang1 += a1V;

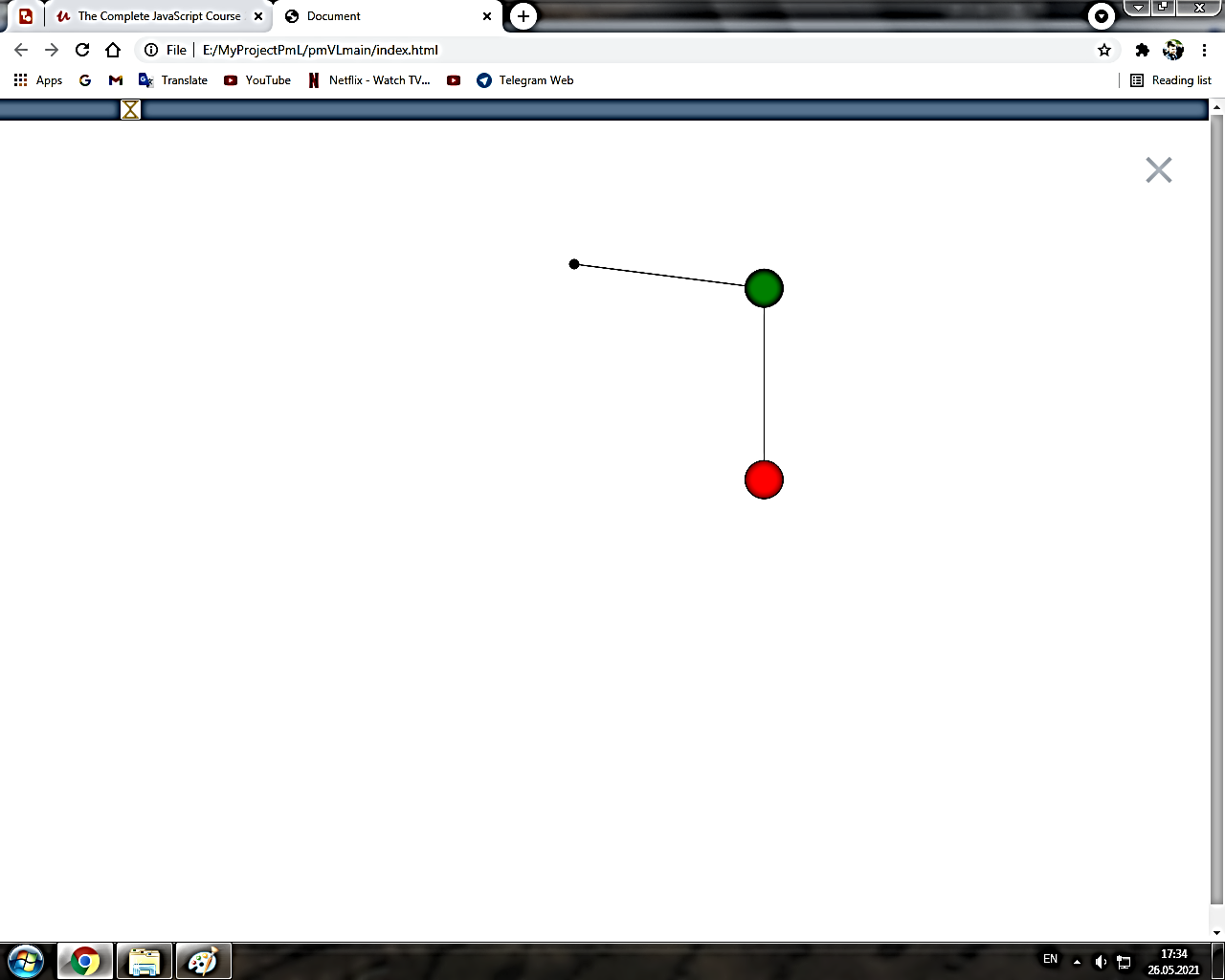
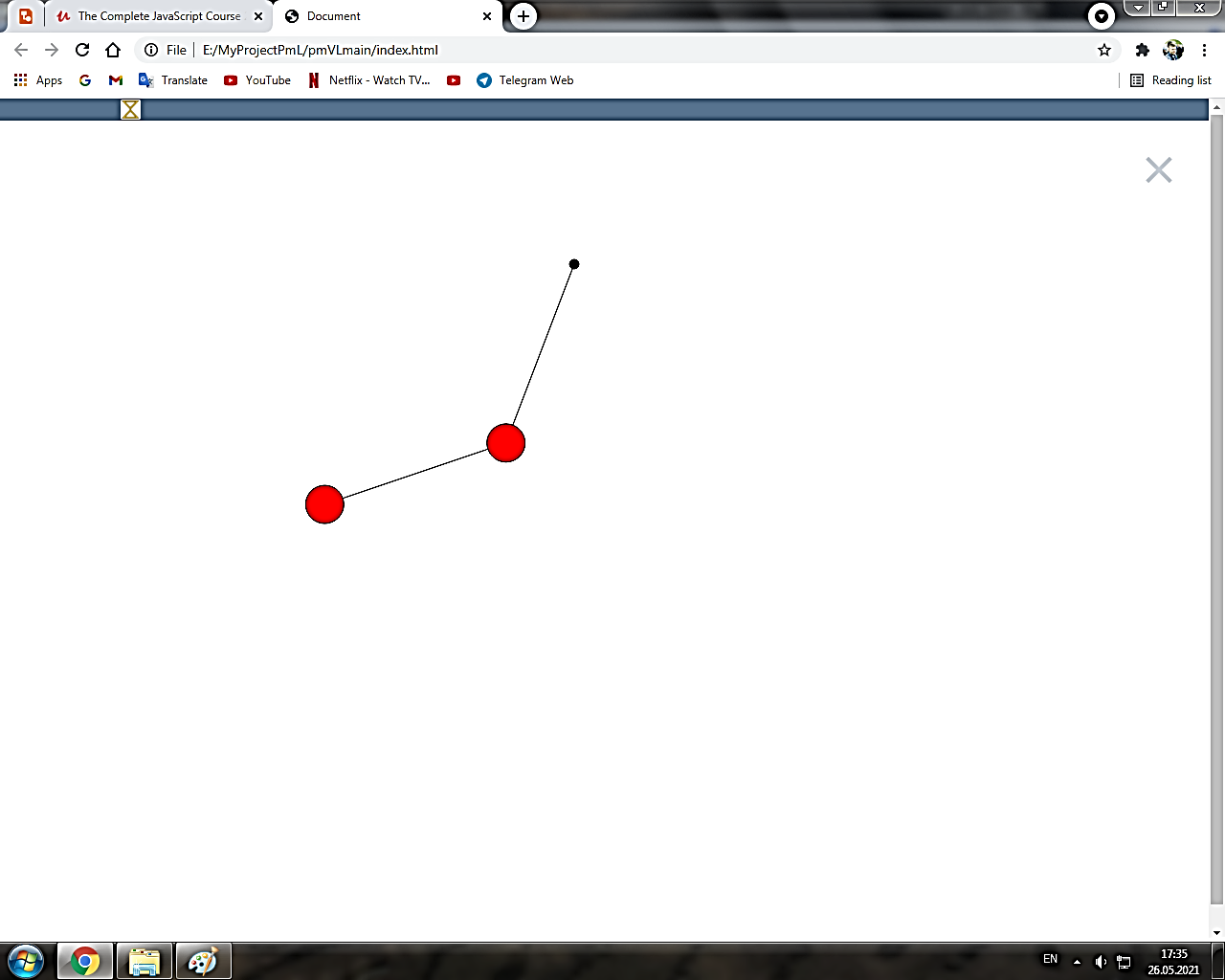
      ang2 += a2V;

      a1V \*= damp1;

      a2V \*= damp2;

  };

В результаті послідовного написання коду, ми отримали модель коливання подвійного маятника.

**Рис. 6**. Модель коливання подвійного маятника (анімація)

**2.1.3. Створення моделі пружинного маятника**

Набравшись досвіду створення простого та подвійного маятників. Ми захотіли створити маятник, у якому крім сили тяжіння та дисипативної сили ще й добавлена пружна сила. Це означає, що наша лінія відіграє роль пружини і змінюється за законом руху. Розпочали ми як завжди за нашим алгоритмом задаючи змінні та константи

//координати кола

*let* BobX = 600;

*let* BobY = 400;

// швидкість маятника

*let* velBobX = 0;

*let* velBobY = 0;

//прискорення маятника

*let* accBobX = 0;

*let* accBobY = 0;

//довжина пружньої лінії,деформація пружини,кут відхилення маятника

*let* l, x, ang;

//координати закріпленої точки

*const* AnchorX = 600;

*const* AnchorY = 150;

//Константи моделі

*const* restLenSpr = 250;

*const* k = 0.1;

*const* m = 1;

*const* r = 40;

*const* damp = 0.996;

*const* g = 9.81;

Знову створили функцію малювання draw()

*const* draw = *function* () {

    ctx.clearRect(0, 0, innerWidth, innerHeight);

    //Малює точку та пружню лінію

    ctx.beginPath();

    ctx.arc(AnchorX, AnchorY, 5, 0, 2 \* Math.PI, false);

    ctx.fillStyle = "#000";

    ctx.fill();

    ctx.moveTo(AnchorX, AnchorY);

    ctx.lineTo(BobX, BobY);

    ctx.stroke();

    //Малює коло

ctx.beginPath();

ctx.arc(BobX, BobY, r, -Math.PI / 2, 2 \* Math.PI + Math.PI / 2, false);

ctx.fillStyle = color;

ctx.fill();

ctx.stroke();

  };

І як завжди анімація, тільки тепер у формулі додана сила пружності [13]:

*const* animate = *function* () {

      requestAnimationFrame(animate);

//Формула для визначення довжини маятника

      l = Math.sqrt(

        (BobX - AnchorX) \* (BobX - AnchorX) +

          (BobY - AnchorY) \* (BobY - AnchorY)

      );

//Дефомація пружини

      x = l - restLenSpr;

//Визначає кут маятника

      ang = Math.atan2(BobX - AnchorX, BobY - AnchorY);

      //Рівняння для прискорення в якому присутня сила пружності

      accBobX = (-k / m) \* x \* Math.sin(ang);

      accBobY = g - (k / m) \* x \* Math.cos(ang);

      velBobX += accBobX / 5;

      velBobY += accBobY / 5;

      BobX += velBobX / 5;

      BobY += velBobY / 5;

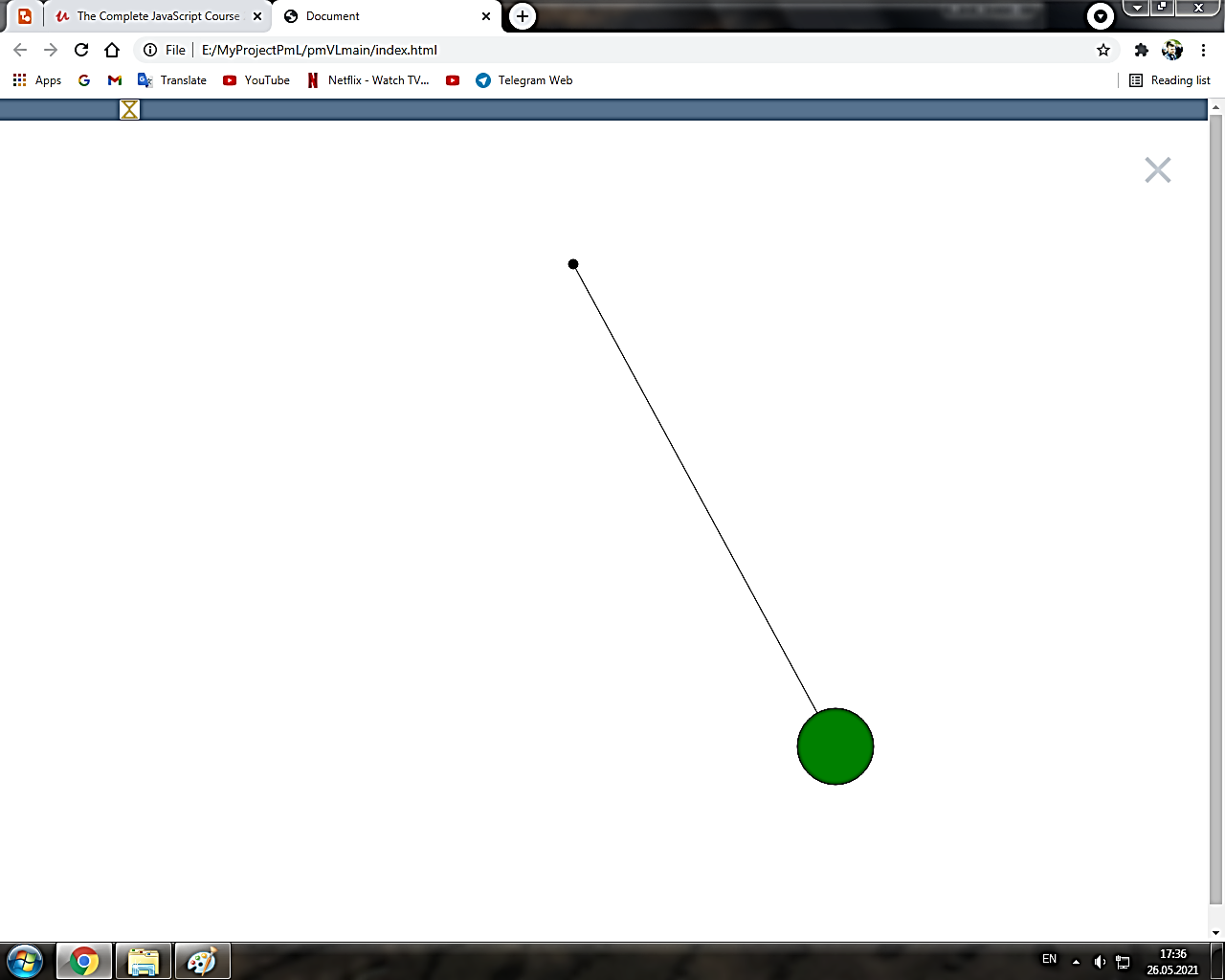
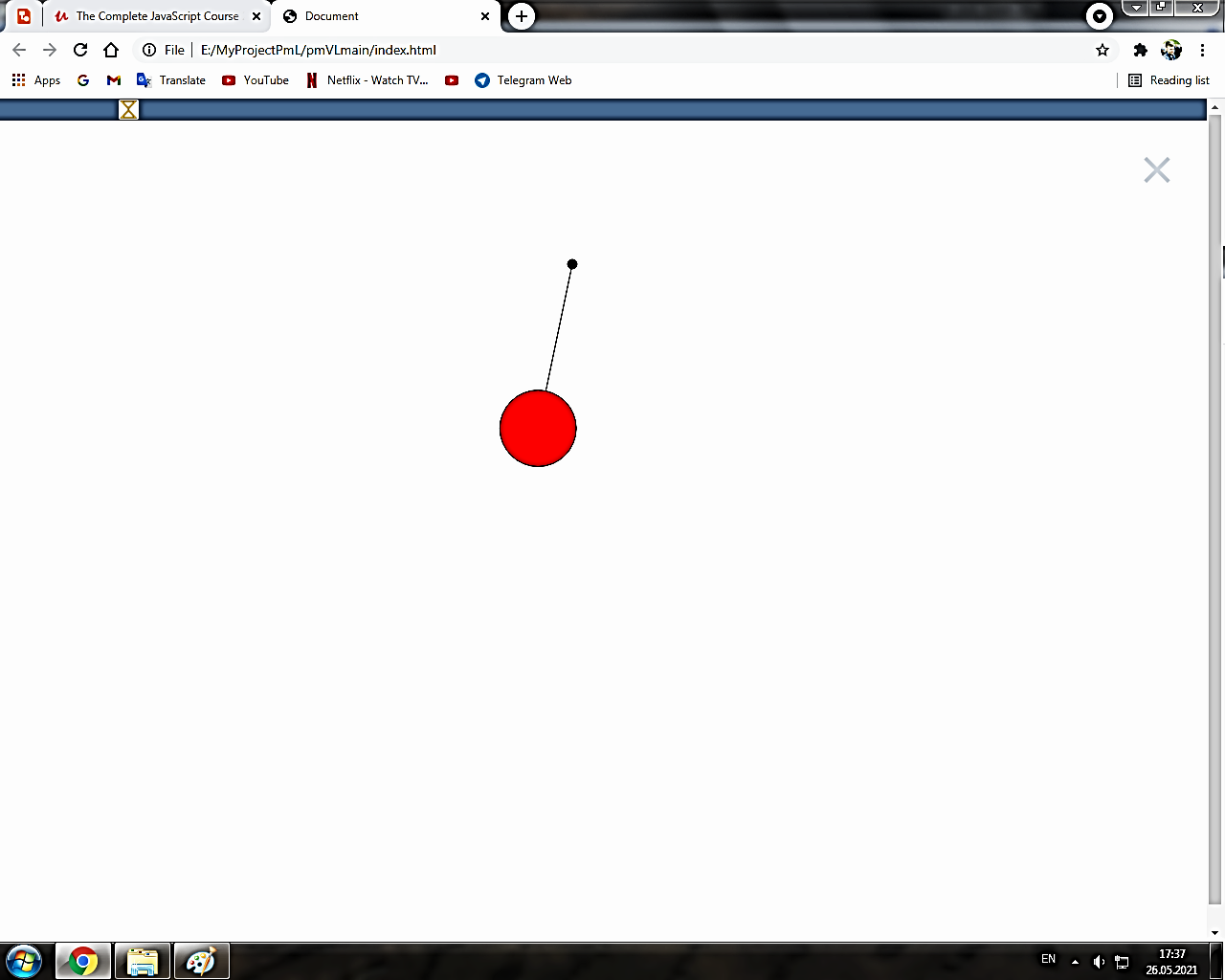
      velBobX \*= damp;

      velBobY \*= damp;

      draw();

    }

Модель коливання такого маятника представлена на рис. 7.

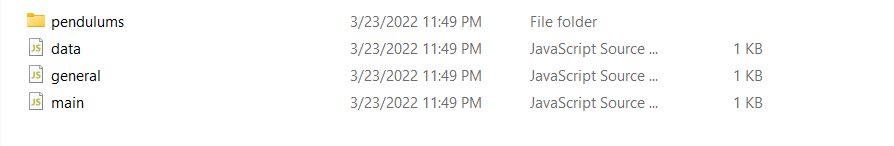
**Рис. 7**. Модель пружинного маятника (анімація)

При розробці та написанні програм слід зазначити їх переваги та недоліки цієї програми.

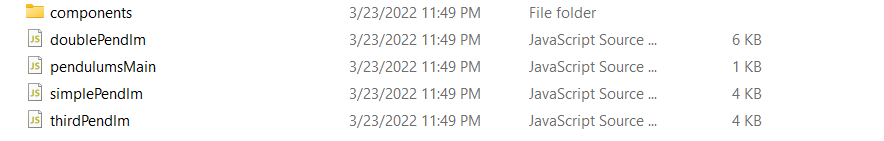
Ця програма дає можливість нам на основі розглянутих моделей відтворити лабораторні умови, розробивши допоміжні вікна, в яких матимемо процес виконання лабораторних робіт. Також цю програму можна відкрити в будь якому браузері який встановлений на комп’ютері.

А ось з мінусів програма повільна, через її функціонал, який займає достатню частину пам’яті, що для такої програми є завелика. Хоча для вирішення цієї проблеми програма буде переписана із застосування методів ООП та модульності для «JavaScript»

**Вдосконалення структури програми**

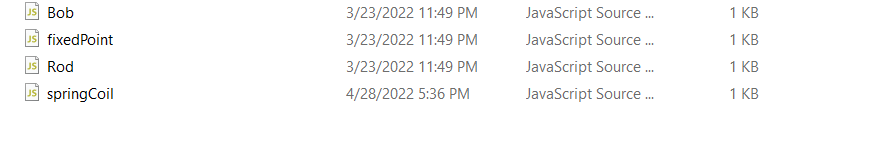
Першим кроком для структуризації програми це розділити код з одного файлу на декілька інших файлів, так як, один файл не може містити всю програму цілком. Це доволі не зручно для подальшої розробки. 

Ми розбили файл на 3 файли та одну папку, яка містить маятники. У файлі “data ” містяться всі необхідні дані програми з якими можна взаємодіяти. У файлі “general” міститься функціонал програми, у якому не має ніякої фізики а ось щодо файлу “main” то він дозволяє поєднати “general” із “main” і саме він запускає всі файли.

Тепер розглянемо папку “pendulums”

В кожному файлі знаходяться усі три маятники, файл який їх об’єднює і пака з компонентами до маятників

В папці компоненти ми використали об’єктно-орієнтоване програмування, щоб розділити маятник на окремі об’єкти



Таким чином вся програма стала структуризованою і набагато зручнішою для написання.

**Висновки:**

1. Уміле поєднання комп'ютерних технологій, методів програмування з традиційними методами викладання фізики та астрономії підвищують рівень засвоєння знань з цих предметів та усвідомлення їх практичного застосування.
2. Використання сучасних методів програмування є засобом підтримки не тільки зацікавленості предметом та його осучасненням, але роблять його більш близьким та наочним, особливо у випадках, коли не має можливості продемонструвати такі фізичні процеси або явища, які неможливо спостерігати візуально в лабораторних умовах.
3. Моделювання різних явищ ні в якому разі не замінює «живих» дослідів, та в поєднанні з ними дозволяє на більш високому рівні пояснити зміст того чи іншого навчального матеріалу.

**Список використаної літератури:**

# Матвєєв О.М. Механіка і теорія відносності/Матвєєв О.М. – К.: Вища школа, 1993.-288с.

# Сивухін Д.В. Общий курс физики. Т.1. [Механика]/Д. В. Сивухин – М.: Наука, 1989.-576 с.

# Яворский Б.М., Детлаф А.А. Справочник по физике/ Яворский Б.М.– М.: Наука, 1985.-512 с.

# Інтернет ресурси:

# <http://enpuir.npu.edu.ua/bitstream/123456789/25425/1/26.pdfО.І>.Кошарний, Л.Ю. Мельничук, О.В. Мельничук Комп’ютерне моделювання фізичних процесів у курсі загальної фізики

# : <https://sites.google.com/site/navcalnapraktikakitvoin/lekciie/lekcia-1>

# <http://kabfiz-roippo.at.ua/Seminar/Book_AVV.pdf>

# https://uk.wikipedia.org/wiki/%D0%9C%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%BD%D0%B8%D0%B9\_%D0%BC%D0%B0%D1%8F%D1%82%D0%BD%D0%B8%D0%BA#/media/%D0%A4%D0%B0%D0%B9%D0%BB:Simple\_pendulum\_height.png

# <https://uk.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D0%B2%D1%96%D0%B9%D0%BD%D0%B8%D0%B9_%D0%BC%D0%B0%D1%8F%D1%82%D0%BD%D0%B8%D0%BA>.

1. https://www.google.com/url?sa=i&url=https%3A%2F%2Fnaukozavr.info%2Ffizuka%2Fpruzhynnyj-mayatnyk%2F&psig=AOvVaw0RgIiYGt7\_zP2yTQa\_vx9O&ust=1622151583207000&source=images&cd=vfe&ved=0CAQQtaYDahcKEwigkJXZp-jwAhUAAAAAHQAAAAAQAg

# http://www.heuristic.su/effects/catalog/tech/byId/description/1214/index.htm

# 10. Поїзд кодування. Виклик кодування # 159: Моделювання простого маятника [Електронний ресурс]: [Відео з списка відтворень Виклик кодування] YouTube. – Режим доступу: https://www.youtube.com/watch?v=NBWMtlbbOag&t=203s – Назва з екрана. – Дата публікації: 20.02.2021. – Дата перегляду : 07.04.2021

# 11. Моя фізична лабораторія .Подвійний маятник [Електронний ресурс] : [Веб-сайт] Режим доступу: https://www.myphysicslab.com/pendulum/double-pendulum-en.html

# 12. Поїзд кодування. Виклик кодування # 93: Подвійний маятник [Електронний ресурс] : [Відео з списка відтворень Виклик кодування] YouTube. – Режим доступу: <https://www.youtube.com/watch?v=uWzPe_S-RVE203s> – Назва з екрана. – Дата публікації: 13.02.2018. – Дата перегляду : 18.04.2021

# 13. Поїзд кодування. Виклик кодування # 160: Пружня сила [Електронний ресурс] : [Відео з списка відтворень Виклик кодування] YouTube. – Режим доступу: <https://www.youtube.com/watch?v=Rr-5HiXquhw&t=807s> – Назва з екрана. – Дата публікації: 22.02.2021. – Дата перегляду : 28.04.2021