МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«ГЛАЗОВСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ ИНСТИТУТ
им. В. Г. Короленко»

О. Е. Данилов, А. Ю. Трефилова

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ КОЛЕБАТЕЛЬНОГО ДВИЖЕНИЯ

Численные методы решения физических задач

Учебно-методическое пособие

Глазов
ГГПИ
2012

УДК 004.942
ББК 22.311
Д18

Д 18 ***Данилов О. Е., Трефилова А. Ю.***
Компьютерное моделирование колебательного движения. Численные методы решения физических задач: учебно-методическое пособие. – Глазов: Глазов. гос. пед. ин-т, 2012. – 36 с.

Р е ц е н з е н т ы :
А. В. Проказов, канд. физ.-мат. наук, доцент кафедры физики и дидактики физики ФГБОУ ВПО «Глазовский государственный педагогический институт им. В. Г. Короленко»,
О. Н. Щепин, канд. пед. наук, доцент кафедры естественно-научных и гуманитарных дисциплин ФГБОУ ВПО «ГЭИИ» (филиал ИжГТУ)

Учебно-методическое пособие предназначено для преподавателей и студентов физико-математических факультетов педагогических институтов.

ISBN 978-5-93008-153-4 УДК 004.942
ББК 22.311

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Любое исследование можно представить как взаимодействие теории и эксперимента. Физический эксперимент – это в какой-то степени некоторая модель действительности. Она может быть очень сложна. Тогда мы упрощаем ее, отбрасывая несущественные факторы. Причиной упрощения исследуемой экспериментальной модели могут являться ее дороговизна, времязатратность и т. д., что приводит к невозможности проведения эксперимента вообще или проведения необходимое число раз. В этом случае натурный эксперимент может быть заменен численным. Этот эксперимент также нужно спланировать, оценить его погрешность. Кроме того, численный эксперимент можно рассматривать как средство, позволяющее упростить используемый математический аппарат или полностью заменить аналитическое решение численным. Он может подтверждать или опровергать гипотезы, прогнозировать. Проверка численной модели осуществляется посредством натурного эксперимента.

Процесс решения физической задачи на компьютере включает ряд последовательных этапов [6, с. 6–10]: 1) изучение и постановка задачи; 2) выбор физической модели; 3) разработка математической модели; 4) численное решение задачи; 5) анализ результатов эксперимента; 6) усовершенствование численной модели; 7) реализация эксперимента. Более подробно остановимся на численном решении. Его составные части: выбор численного метода, разработка программы, анализ и обработка входной информации, проведение математического эксперимента на компьютере, обработка и предварительный анализ информации [6, с. 8–9].

Цель данного пособия – познакомить читателей с конкретной реализацией численного решения задач физики, связанных с колебательным движением математического маятника. При описании процесса моделирования мы несколько отступили от его классического представления, считая первоочередными образовательные цели и задачи. Программы, моделирующие колебания, написаны на языке программирования Borland Pascal (Версия 7.0). Выбор языка обусловлен тем, что он является одним из наиболее распространенных алгоритмических языков в системе российского обра-

зования. Он удобен для решения вычислительных и других задач, способствует внедрению современной технологии программирования, позволяет записывать сложные алгоритмы в компактной форме, может выступать в роли базового языка для обучения программированию, лаконичен, относительно прост и доступен [7, с. 5].

Скажем несколько слов о том, как нужно работать с представленными в пособии компьютерными программами. Обычно такая программа состоит из блока констант, блока переменных, подпрограмм и основной части программы. Для удобства работы в блок констант помещены те величины, которые пользователь программы может менять перед каждым новым запуском программы для изменения условий вычислительного эксперимента. Это освобождает его от рутинного ввода значений величин по запросу программы после ее запуска, что часто приходится делать при работе с такими программами. Текст всех программ имеет такую структуру, чтобы было понятно, где начинается и где заканчивается та или иная часть программы. Авторы пособия не стремились к компактности записей, а старались сделать их максимально понятными для тех, кто с ними будет работать. Перед тем, как использовать программы, убедитесь в том, что на вашем компьютере установлены все необходимые драйверы, что они находятся именно там, где указано в программе.

Глава 1
ТЕОРИЯ МАТЕМАТИЧЕСКОГО МАЯТНИКА

1.1. МЕХАНИЧЕСКИЕ КОЛЕБАНИЯ. КРУГОВОЙ МАТЕМАТИЧЕСКИЙ МАЯТНИК

Колебания – это движения или процессы, обладающие той или иной степенью повторяемости во времени [14, с. 293]. Изучение колебаний составляет основу многих областей физики и техники. Мы будем рассматривать колебания механической системы, поэтому они называются *механическими.* Хотя колебания, рассматриваемые в механике, радиотехнике, акустике и т. д., отличаются по своей физической природе, законы этих колебаний остаются одними и теми же. По этой причине изучение механических колебаний важно не только потому, что такие колебания часто имеют место в технике, но и вследствие того, что результаты исследований могут быть использованы и для выяснения колебательных законов в других областях.

Маятник – тело, совершающее под действием приложенных сил (обычно силы тяжести) колебания около неподвижной точки или вокруг оси [14, с. 399]. Простейший маятник состоит из груза, подвешенного на нити. Если считать нить нерастяжимой и пренебречь размерами груза по сравнению с длиной нити, а массой нити по сравнению с массой груза, то такую систему можно рассматривать как материальную точку массой m, находящуюся на неизменном расстоянии l от точки подвеса O. Этот идеализированный маятник называется *математическим.*

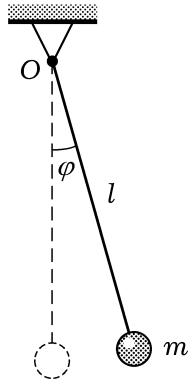

Рис. 1.1.1

Если маятник, отклоненный от положения равновесия, отпустить, то он будет совершать колебания в одной вертикальной плоскости, а его центр тяжести будет двигаться по дуге окружности. В этом случае маятник называют *плоским,* или *круговым математическим,* и его положение в пространстве определяется одной координатой, например, углом φ отклонения от положения равновесия (рис. 1.1.1). Такие системы называют *системами с одной степенью свободы*.

Когда в силу наложенных связей материальная точка вынуждена двигаться по заданной кривой, говорят, что движение точки является *несвободным.*

1.2. УРАВНЕНИЕ СВОБОДНЫХ КОЛЕБАНИЙ МАТЕМАТИЧЕСКОГО МАЯТНИКА

1. Для определения закона колебаний кругового математического маятника воспользуемся основным уравнением вращательного движения (рис. 1.2.1):

$$\vec{M}_{mg} + \vec{M}_T = J\vec{\varepsilon}, \qquad (1.2.1)$$

где $\vec{M}_{mg} = [\vec{l}\, m\vec{g}]$ – момент силы тяжести $m\vec{g}$ относительно точки подвеса, $\vec{M}_T = [\vec{l}\,\vec{T}] = \vec{0}$ – момент силы натяжения $\vec{T}$ относительно этой же точки, $J = ml^2$ – момент инерции материальной точки m относительно оси, перпендикулярной плоскости рисунка и проходящей через точку подвеса, $\vec{\varepsilon}$ – угловое ускорение маятника. Учитывая то, что проекция момента силы тяжести

$$M_{mg_z} = -mgl\sin\varphi,$$

так как вектор $\vec{\varphi}$ направлен в сторону, противоположную оси z (положительный отсчет угла φ ведется при вращении маятника от положения равновесия против часовой стрелки), и угловое ускорение маятника

$$\vec{\varepsilon} = \frac{\partial^2\vec{\varphi}}{\partial t^2},$$

получаем

$$-mgl\sin\varphi = ml^2\frac{\partial^2\varphi}{\partial t^2},$$

$$\frac{\partial^2 \varphi}{\partial t^2} + \frac{g}{l} \sin \varphi = 0. \qquad (1.2.2)$$

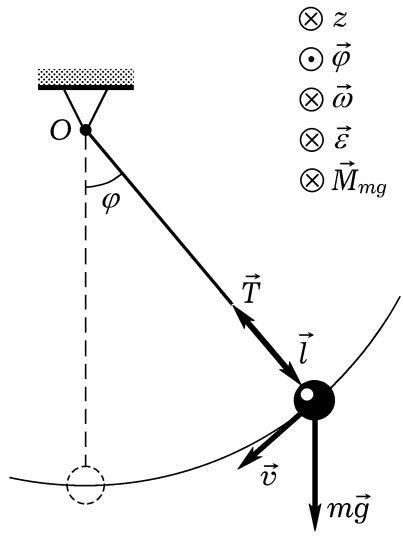

Рис. 1.2.1

Полученное нелинейное дифференциальное уравнение второго порядка в обычных функциях не интегрируется [8, с. 326]. Как правило, ограничиваются рассмотрением малых колебаний маятника, считая, что угол φ мал. В этом случае, если разложить $\sin \varphi$ в ряд

$$\sin \varphi = \varphi - \frac{\varphi^3}{3!} + \frac{\varphi^5}{5!} - \ldots$$

и пренебречь членами разложения порядка выше первого, то получаем линейное однородное дифференциальное уравнение второго порядка с постоянными коэффициентами:

$$\frac{\partial^2 \varphi}{\partial t^2} + \frac{g}{l} \varphi = 0, \qquad (1.2.3)$$

которое имеет аналитическое решение. Для его нахождения решаем соответствующее характеристическое уравнение

$$\lambda^2 + \frac{g}{l} = 0,$$

корни которого равны

$$\lambda_{1,2} = \pm\sqrt{\frac{g}{l}}.$$

Тогда решение уравнения запишется в виде:

$$\varphi = C_1 \cos\left(\sqrt{\frac{g}{l}}t\right) + C_2 \sin\left(\sqrt{\frac{g}{l}}t\right), \qquad (1.2.4)$$

где постоянные интегрирования C_1 и C_2 определяются из начальных условий движения. Например, если $\varphi=\varphi_0$ при $t = 0$, то

$$\varphi_0 = C_1 \cos 0 + C_2 \sin 0,$$
$$C_1 = \varphi_0,$$

и

$$\varphi = \varphi_0 \cos\left(\sqrt{\frac{g}{l}}t\right),$$

где φ_0 – амплитуда колебаний. Такое движение называется *гармоническим колебанием.*

Период колебаний в рассматриваемом случае

$$T_1 = 2\pi\sqrt{\frac{l}{g}}. \qquad (1.2.5)$$

При нахождении периода из точного дифференциального уравнения (1.2.2) имеем [1, с. 239]:

$$T_2 = 2\pi\sqrt{\frac{l}{g}}\left(1 + \frac{1}{4}\sin^2\frac{\varphi_0}{2} + \frac{9}{64}\sin^4\frac{\varphi_0}{2} + \ldots\right), \qquad (1.2.6)$$

где φ_0 – амплитуда колебаний. Таким образом, в общем случае колебания маятника свойством изохронности не обладают – его период зависит от амплитуды φ_0 колебаний.

2. Если на маятник будет действовать еще сила сопротивления движению (рис. 1.2.2), пропорциональная скорости $\vec{v}$ маятника,

$$\vec{F} = -k\vec{v},$$

то уравнение (1.2.1) нужно привести к виду:

$$\vec{M}_{mg} + \vec{M}_T + \vec{M}_F = J\vec{\varepsilon}, \qquad (1.2.7)$$

где момент силы сопротивления

$$\vec{M}_F = [\vec{l}\vec{F}] = -[\vec{l}k\vec{v}],$$

k – коэффициент сопротивления. Проекция

$$M_{F_z} = -kvl\sin\frac{\pi}{2} = -kvl.$$

Скорость равна

$$v = \omega l = \frac{\partial\varphi}{\partial t}l,$$

где ω – угловая скорость маятника.

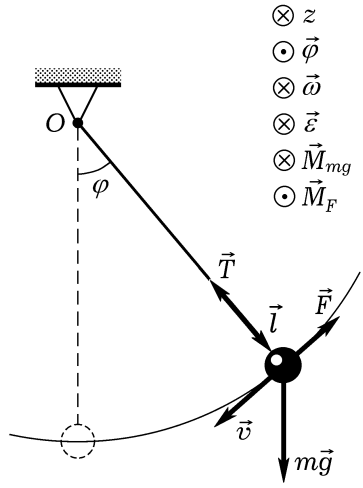

Рис. 1.2.2

Тогда

$$-mgl\sin\varphi - kl^2\frac{\partial\varphi}{\partial t} = ml^2\frac{\partial^2\varphi}{\partial t^2},$$

$$\frac{\partial^2\varphi}{\partial t^2} + \frac{k}{m}\frac{\partial\varphi}{\partial t} + \frac{g}{l}\sin\varphi = 0. \qquad (1.2.8)$$

В случае малых значений ωуравнение (1.2.8) можно записать так:

$$\frac{\partial^2\varphi}{\partial t^2} + \frac{k}{m}\frac{\partial\varphi}{\partial t} + \frac{g}{l}\varphi = 0. \qquad (1.2.9)$$

Характеристическое уравнение

$$\lambda^2 + \frac{k}{m}\lambda + \frac{g}{l} = 0$$

имеет корни

$$\lambda_{1,2} = -\frac{k}{2m} \pm \sqrt{\frac{k^2}{4m^2} - \frac{g}{l}}.$$

Здесь возможны три случая.

Во-первых,

$$\frac{k^2}{4m^2} - \frac{g}{l} > 0,$$

то есть

$$\frac{k}{2m} > \sqrt{\frac{g}{l}}.$$

Тогда общее решение уравнения имеет вид

$$\varphi = C_1 e^{-\frac{k}{2m} + \sqrt{\frac{k^2}{4m^2} - \frac{g}{l}}} + C_2 e^{-\frac{k}{2m} - \sqrt{\frac{k^2}{4m^2} - \frac{g}{l}}}. \qquad (1.2.10)$$

Движение, соответствующее этому решению, является *апериодическим.*

Во-вторых,

$$\frac{k^2}{4m^2} - \frac{g}{l} = 0,$$

или

$$\frac{k}{2m} = \sqrt{\frac{g}{l}}.$$

Тогда корни характеристического уравнения совпадают:

$$\lambda_1 = \lambda_2 = -\frac{k}{2m} < 0,$$

и общим решением будет

$$\varphi = (C_1 + C_2 t) e^{-\frac{k}{2m}t}. \qquad (1.2.11)$$

Это движение также является апериодическим.

В-третьих,

$$\frac{k^2}{4m^2} - \frac{g}{l} < 0,$$

то есть

$$\frac{k^2}{4m^2} < \frac{g}{l}.$$

В этом случае характеристическое уравнение имеет сопряженные комплексные корни с отрицательной действительной частью:

$$\lambda_{1,2} = -\frac{k}{2m} \pm i\sqrt{\frac{k^2}{4m^2} - \frac{g}{l}}.$$

Общее решение

$$\varphi = \left(C_1 \cos\left(\sqrt{\frac{k^2}{4m^2} - \frac{g}{l}} \right) t + C_2 \sin\left(\sqrt{\frac{k^2}{4m^2} - \frac{g}{l}} \right) t \right) e^{-\frac{k}{2m}t}. \qquad (1.2.12)$$

В этом случае тело совершает *затухающие гармонические колебания* с периодом

$$T = \frac{2\pi}{\sqrt{\frac{k^2}{4m^2} - \frac{g}{l}}}. \qquad (1.2.13)$$

В отличие от гармонических колебаний (1.2.4) здесь амплитуда непостоянна. Величина

$$\beta = \frac{k}{2m} \qquad (1.2.14)$$

называется *коэффициентом затухания.* Быстроту затухания характеризует величина $e^{-\beta t}$.

Однородные дифференциальные уравнения (1.2.3) и (1.2.9) описывают колебания маятника, которые называют *свободными,* или *собственными.* Это повторяющееся движение системы, предоставленной самой себе, в отсутствие внешних воздействий.

1.3. ПОСТАНОВКА ЗАДАЧИ

Постановка задачи включает определение целей решения. Как правило, задача формулируется в виде системы дифференциальных уравнений относительно искомой функции совместно с граничными и начальными условиями. Ниже приведена математическая модель, представляющая собой математическую формулировку задачи.

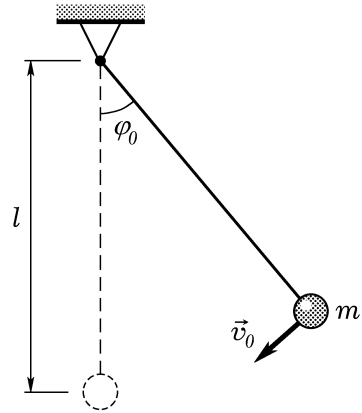

Рис. 1.3.1

Пусть маятник представляет собой систему из материальной точки массой m и нити длиной l. Маятник находится в гравитационном поле $\vec{g}$, коэффициент сопротивления среды k. Тогда смещение маятника относительно положения равновесия описывается уравнением (1.2.7). Начальные условия: угловая координата маятника и его угловая скорость, или линейная скорость материальной точки, в момент времени $t = t_1$ являются известными величинами $\varphi(t_1)=\varphi_0$ и $\omega(t_1)=\omega_0$, или $\vec{v}(t_1)=\vec{v}_0$. Граничные условия в явном виде формулировать пока не будем. Требуется найти углы $\varphi(x,t)$ отклонений маятника в последующие моменты времени.

Переход от математической модели к компьютерной показан в главе 2.

Глава 2. МОДЕЛИРОВАНИЕ КОЛЕБАНИЙ МАТЕМАТИЧЕСКОГО МАЯТНИКА

2.1. КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ

Модель – это упрощенное подобие реального объекта. Она может быть *материальной* и *информационной.* Информационная модель – это в той или иной форме описание объекта моделирования. Любая наука занимается информационным моделированием, так как любые знания носят приближенный, то есть модельный

характер. Эти знания уточняются, углубляются, но все равно остаются приближенными. Процесс уточнения моделей является бесконечным.

Построение компьютерной модели начинается с анализа объекта моделирования (рис. 2.1.1). Он анализируется как *система,* или объект, состоящий из взаимосвязанных частей и существующий как единое целое. На основании анализа строится теоретическая информационная модель и осуществляется построение компьютерной информационной модели. Важным этапом любого моделирования является проверка построенной модели на соответствие реальной системе, в результате которой могут быть выявлены и устранены недостатки модели.

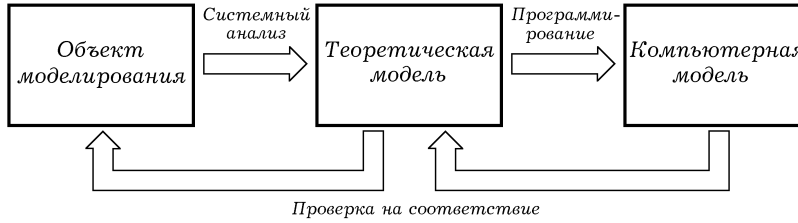

Рис. 2.1.1

2.2. ДИСКРЕТИЗАЦИЯ. ЧИСЛЕННОЕ ИНТЕГРИРОВАНИЕ

Компьютер не может выполнять операции с функциями, определенными на непрерывных множествах. Он выполняет алгебраические операции с конечным числом переменных, в качестве которых могут быть взяты значения непрерывных функций в конечном числе точек. Говорят, что в этом случае осуществляется дискретизация с использованием конечно-разностных методов, один из которых рассмотрим ниже.

Обратимся к нашей задаче, сформулированной в параграфе 1.3. Непрерывную область переменной времени t заменим дискретным множеством: $t_i = i\tau,\ \ i = 1, 2\ldots n$. Вместо непрерывной функции $\varphi\ (t)$ рассмотрим числовую последовательность значений φ_i, соответствующих моментам времени t_i. Представим формулу (1.2.8) в виде:

$$\varepsilon + \frac{k}{m}\omega + \frac{g}{l}\sin\varphi = 0, \qquad (2.2.1)$$

отсюда

$$\varepsilon = -\frac{k}{m}\omega - \frac{g}{l}\sin\varphi. \qquad (2.2.2)$$

Угловая скорость

$$\omega = \int_0^t \varepsilon dt, \qquad (2.2.3)$$

угол отклонения маятника

$$\varphi = \int_0^t \omega dt. \qquad (2.2.4)$$

В случаях неизменных значений ω и φ формулы (2.2.3) и (2.2.4) выглядят так:

$$\omega = \omega_0 + \varepsilon t, \quad \varphi = \varphi_0 + \omega t,$$

где ω_0 и φ_0 – соответственно значения угловой скорости и угла отклонения маятника от положения равновесия в момент времени $t = 0$.

Применяемый ниже способ дискретизации уравнений не является наилучшим, но он достаточно прост. Формулы (2.2.2) – (2.2.4) преобразуем к виду

$$\varepsilon^i = -\frac{k}{m}\omega^{i-1} - \frac{g}{l}\sin\varphi^{i-1},$$
$$\omega^i = \omega^{i-1} + \varepsilon^i \tau,$$
$$\varphi^i = \varphi^{i-1} + \omega^i \tau,$$

где верхний индекс указывает номер шага по времени.

Мы решаем уже несколько упрощенную задачу, которая является приближением исходной. Существует погрешность численного метода. Она возникает, потому что мы считаем ускорение ε и скорость ω на шаге интегрирования постоянными величинами и заменяем интегралы (2.2.3) и (2.3.4) суммами с конечным числом членов. Эта погрешность регулируема, так как можно изменять шаг τ по времени t.

При вычислениях на компьютере возникают также погрешности округления, связанные с ограничением количества разрядов чисел, которыми оперирует компьютер.

2.3. АЛГОРИТМ ЧИСЛЕННОГО РЕШЕНИЯ

Задача сводится к решению системы уравнений:

$$\begin{cases} \varepsilon^i = -\dfrac{k}{m}\omega^{i-1} - \dfrac{g}{l}\sin\varphi^{i-1} \\ \omega^i = \omega^{i-1} + \varepsilon^i \tau \\ \varphi^i = \varphi^{i-1} + \omega^i \tau \end{cases} \qquad (2.3.1)$$

с известными начальными условиями (φ^1, ω^1).

Система уравнений (2.3.1) решается для данного момента времени t_i. Состояние системы выводится на экран компьютера. Затем происходит переход к следующему моменту времени t_{i+1}, новое изображение маятника выводится на экран и т. д. (рис. 2.3.1).

Задача должна ставиться корректно, то есть для любых входных данных решение должно существовать, быть единственным и устойчивым по этим данным. Устойчивость задачи состоит в том, что решение непрерывно зависит от входных данных. При отсутствии устойчивости даже небольшие погрешности в исходных данных могут приводить к большим погрешностям решения или неверному результату. Предлагаем читателю определить условие устойчивости самостоятельно с помощью компьютерной модели.

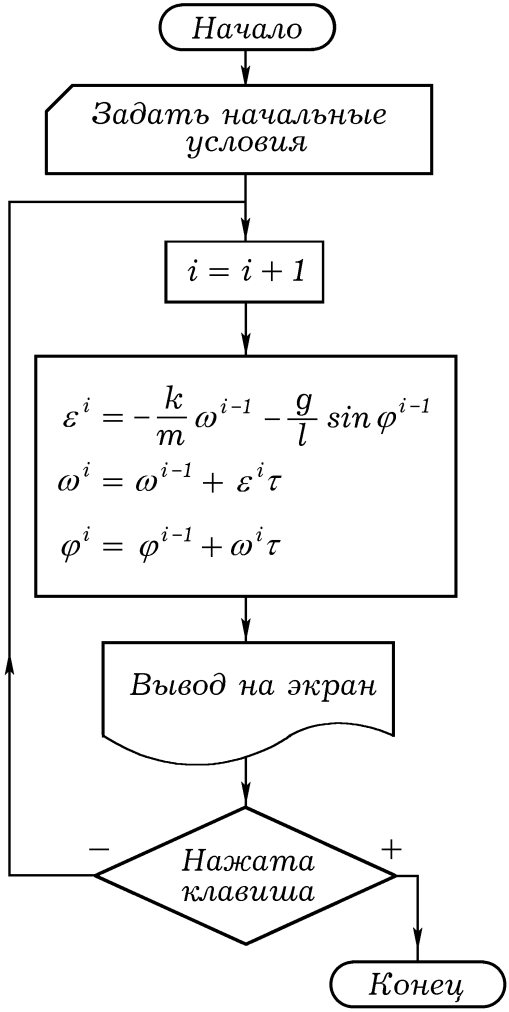
Начало
Задать начальные условия
$i = i + 1$
$\varepsilon^i = -\frac{k}{m}\omega^{i-1} - \frac{g}{l}\sin\varphi^{i-1}$
$\omega^i = \omega^{i-1} + \varepsilon^i\tau$
$\varphi^i = \varphi^{i-1} + \omega^i\tau$
Вывод на экран
Нажата клавиша
−
+
Конец

Рис. 2.3.1

Глава 3
ПРОГРАММИРОВАНИЕ В BORLAND PASCAL

3.1. СОСТАВЛЕНИЕ ПРОГРАММЫ

Рассмотрим программу `Math_01`. В ее начале происходит подгружение стандартных модулей CRT.TPU и GRAPH.TPU. Первый содержит процедуры ввода и вывода, обеспечивающие контроль через клавиатуру, второй – пакет стандартных графических процедур. Далее объявляются константы – величины, которые не будут изменяться в течение выполнения программы, но пользователь может изменять их перед каждым новым запуском, варьируя некоторые параметры компьютерной модели. В следующем блоке находятся переменные `x, y` – горизонтальная и вертикальная координаты груза, `xx, yy` – переменные, хранящие значения координат груза на предыдущем временном шаге, `eps, w, phi` – переменные, обозначающие угловое ускорение, угловую скорость и угол отклонения маятника.

Процедура `OpenGraph` инициализирует графический режим работы монитора с разрешением 640 × 480. Для этого используется стандартный драйвер EGAVGA.BGI, который расположен в папке `c:\bp\bgi`. Процедура `Glimmer` устраняет мерцание экрана при смене изображений, обеспечивая необходимую задержку. В процедуре `Display` предусмотрена очистка экрана от предыдущего изображения с последующим выводом на экран следующего за ним изображения маятника на данном временно**м** шаге. Строки программы:

```
SetColor(Black);
SetFillStyle(SolidFill, Black);
Line(x0, y0, Round(xx),
Round(yy));
FillEllipse(Round(xx),
Round(yy), r, r);
```

можно заменить одной строкой:

```
ClearDevice;
```

однако это приводит к тому, что на экран не выводится верхняя, хоть и незначительная, часть изображения маятника.

В основной части программы сначала инициализируется графический режим. После чего выводится на экран изображение

подвеса. Затем задаются начальные условия: значение угла отклонения, соответствующие ему координаты груза (начальная угловая скорость считается равной нулю). Далее в программе предусмотрен расчет угла отклонения на следующем временном шаге и соответствующих этому углу координат груза, а также вывод изображения маятника с помощью процедуры `Display`. Этот процесс повторяется до тех пор, пока не будет нажата любая клавиша. В случае, когда клавиша будет нажата, произойдет закрытие текущего графического режима и выход из программы.

При моделировании мы не осуществляли приведение величин к безразмерному виду, что обычно делают в этом случае. Также не проведено масштабирование, то есть значения констант и переменных приведены в некоторых условных единицах, чтобы читателю было легче понять логику программирования. В дальнейшем можно легко изменить программу, введя необходимые масштабные коэффициенты для физических величин.

```
{Программа
СВОБОДНЫЕ КОЛЕБАНИЯ КРУГОВОГО МАТЕМАТИЧЕСКОГО МАЯТНИКА}
Program Math_01;

{Подключение модулей подпрограмм}
Uses Crt, Graph;

{Описание констант}
Const phi0=30; {Начальный угол наклона маятника}
      w0=0;    {Начальная угловая скорость маятника}
      l=400;   {Длина нити}
      r=8;     {Радиус груза}
      light=2; {Радиус блика}
      k=0.02;  {Коэффициент сопротивления}
      m=1;     {Масса груза}
      xp=320;  {Координата x точки подвеса}
      yp=10;   {Координата y точки подвеса}
      g=9.8;   {Ускорение свободного падения}
      tau=0.4; {Шаг по времени}

{Описание переменных}
Var x, y, xx, yy, w, eps, phi: Real;

{Процедура инициализации графического режима 640 x 480}
Procedure OpenGraph;
Var Driver, Mode, ErrorCode: Integer;
Begin
  Driver:=Detect;
  InitGraph(Driver, Mode, 'c:\bp\bgi');
  ErrorCode:=GraphResult;
  If ErrorCode <> grOK Then Halt(1);
```

```
End;
{Процедура исключения мерцаний изображений}
Procedure Glimmer;
Begin
  Repeat
  Until Port[$3da] And 8 <> 0;
End;

{Процедура вывода изображения маятника на экран}
Procedure Display;
Begin
  Glimmer;
  SetColor(Black);
  SetFillStyle(SolidFill, Black);
  Line(xp, yp, Round(xx), Round(yy));
  FillEllipse(Round(xx), Round(yy), r, r);
  SetLineStyle(SolidLn, 0, NormWidth);
  SetColor(DarkGray);
  Line(xp, yp, Round(x), Round(y));
  SetLineStyle(SolidLn, 0, NormWidth);
  SetColor(DarkGray);
  SetFillStyle(SolidFill, DarkGray);
  FillEllipse(Round(x), Round(y), r, r);
  SetColor(LightGray);
  SetFillStyle(SolidFill, White);
  FillEllipse(Round(x)-3, Round(y)-3, light, light);
End;
{Основная часть программы}
Begin
  OpenGraph;
  SetFillStyle(SolidFill, DarkGray);
  Bar(xp-20, yp-5, xp+20, yp-1);
  phi:=phi0*Pi/180;
  w:=w0;
  x:=xp+l*sin(phi);
  y:=yp+l*cos(phi);
  Repeat
    Begin
      xx:=x;
      yy:=y;
      eps:=-k*w/m-g*sin(phi)/l;
      w:=w+eps*tau;
      phi:=phi+w*tau;
      x:=xp+l*sin(phi);
      y:=yp+l*cos(phi);
      Display;
    End;
  Until KeyPressed;
  CloseGraph;
End.
```

3.2. ИССЛЕДОВАНИЕ КОМПЬЮТЕРНОЙ МОДЕЛИ

В программе `Math_01` положительным направлением отсчета угла отклонения маятника от положения равновесия считается вращение по часовой стрелке (начало отсчета – линия, соответствующая положению равновесия).

Результаты моделирования движения маятника в течение полупериода представлены на рис. 3.1.1. Подумайте, как надо изменить программу `Math_01`, чтобы получить на экране одновременно такие изображения маятника в разные моменты времени, а не его движение в динамике. Представленные последовательные положения маятника следуют одно за другим с интервалом времени 2 при шаге по времени `tau=1`.

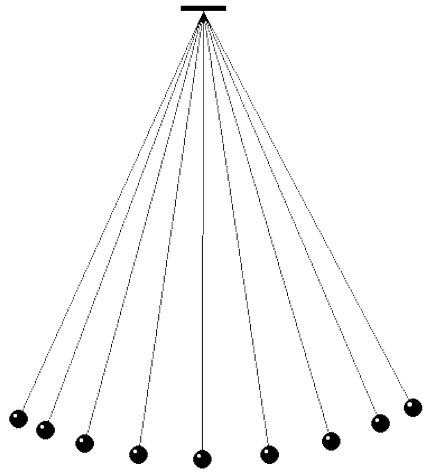

Рис. 3.2.1

Задания для самостоятельного исследования

1. Промоделируйте колебания кругового математического маятника при различных значенях `tau`, `k`, `m`, `l` с помощью программы `Math_01`. Определите условие устойчивости схемы численного моделирования.

2. Меняя начальные условия `phi0`, `w0`, пронаблюдайте свободные колебания.

3. Измените программу так, чтобы рядом с изображением маятника на экран выводился график *φ (t)*. Объясните результаты, полученные при различных значениях параметров `k`, `m`, `l`, `phi0`, `w0`. Меняется ли при этом период колебаний маятника?

4. Внесите в программный код изменения, чтобы рядом с изображением маятника на экран выводился график *ω (t)*. Объясните результаты, полученные при различных значениях параметров `k`, `m`, `l`, `phi0`, `w0`.

5. Измените программу так, чтобы рядом с изображением маятника на экран выводился график *ε (t)*. Объясните результаты, полученные при различных значениях параметров `k`, `m`, `l`, `phi0`, `w0`.

6. Измените программу так, чтобы рядом с изображением маятника на экран выводился график *ω (φ)*. Такие графики называются *интегральными кривыми,* или *фазовыми диаграммами.* Рассмотрите случаи с различными значениями параметров `k`, `m`, `l`, `phi0`, `w0`. Оцените влияние затухания на вид интегральных кривых.

7. Исследуйте свободные колебания маятника, считая, что модуль силы сопротивления его движению пропорционален квадрату скорости материальной точки: $F = kv^2$. Получите графики *ε (t), ω (t), φ (t), ω (φ).*

8. Исследуйте вынужденные колебания маятника. В этом случае уравнение его колебаний будет иметь вид:

$$\frac{\partial^2 \varphi}{\partial t^2} + \frac{k}{m}\frac{\partial \varphi}{\partial t} + \frac{g}{l}\varphi = \frac{F_B}{m}\cos \omega_B t,$$

где величина $F_B \cos \omega_B t$ определяет *внешнюю,* или *возмущающую* силу. Пронаблюдайте явление резонанса.

3.3. ИССЛЕДОВАНИЕ ВЛИЯНИЯ ВНЕШНИХ ТЕЛ НА КОЛЕБАНИЯ КРУГОВОГО МАТЕМАТИЧЕСКОГО МАЯТНИКА

Если на пути маятника поместить массивную пластину, то маятник, испытывая удар о нее, будет получать толчок, а период его колебаний станет меньше, чем в отсутствие пластины. Исследуйте такую колебательную систему, используя программу `Math_02`, которая моделирует ситуацию, когда удар маятника о пластину происходит при прохождении им положения равновесия.

```
{Программа
СТОЛКНОВЕНИЕ МАТЕМАТИЧЕСКОГО МАЯТНИКА С ВЕРТИКАЛЬНОЙ ПРЕГРАДОЙ}
Program Math_02;

{Подключение модулей подпрограмм}
Uses Crt, Graph;

{Описание констант}
Const phi0=40; {Начальный угол отклонения маятника}
      w0=0;    {Начальная угловая скорость маятника}
      l=360;   {Длина нити}
      r=8;     {Радиус тела}
      light=2; {Радиус блика}
      k=0.002; {Коэффициент сопротивления}
      m=1;     {Масса груза}
      c=0.9;   {Коэффициент восстановления}
      xp=320;  {Координата x точки подвеса}
      yp=55;   {Координата y точки подвеса}
      g=9.8;   {Ускорение свободного падения}
      tau=0.2; {Шаг по времени}
      a=140;   {Высота преграды}
      b=5;     {Толщина преграды}

{Описание переменных}
Var x, y, w, eps, phi: Real;

{Процедура инициализации графического режима}
Procedure OpenGraph;
Var Driver, Mode, ErrorCode: Integer;
Begin
  Driver:=Detect;
  InitGraph(Driver, Mode, 'c:\bp\bgi');
  ErrorCode:=GraphResult;
  If ErrorCode <> grOK Then Halt(1);
End;

{Процедура исключения мерцаний изображений}
Procedure Glimmer;
Begin
  Repeat
  Until Port[$3da] And 8 <> 0;
End;

{Процедура вывода изображения колеблющегося тела на экран}
Procedure Display;
Begin
  SetColor(DarkGray);
  SetFillStyle(SolidFill, DarkGray);
  Bar(xp-20, yp-5, xp+20, yp-1);
  Bar(xp+r+3, 480-a, xp+r+b+3, 480);
  Line(xp, yp, Round(x), Round(y));
  FillEllipse(Round(x), Round(y), r, r);
  SetColor(LightGray);
```

```
  SetFillStyle(SolidFill, White);
  FillEllipse(Round(x)-3, Round(y)-3, light, light);
  Glimmer;
  ClearDevice;
End;

{Основная часть программы}
Begin
  OpenGraph;
  phi:=phi0*Pi/180;
  w:=w0;
  x:=xp-l*sin(phi);
  y:=yp+l*cos(phi);
  Repeat
    Begin
      If (xp-x)<0 Then
        Begin
          phi:=0;
          w:=-c*w;
        End;
      eps:=-k*w/m-g*sin(phi)/l;
      w:=w+eps*tau;
      phi:=phi+w*tau;
      x:=xp-l*sin(phi);
      y:=yp+l*cos(phi);
      Display;
    End;
  Until KeyPressed;
  CloseGraph;
End.
```

Задания для самостоятельного исследования

1. Промоделируйте колебания кругового математического маятника при различных значениях `tau`, `k`, `m`, `l`, `c` с помощью программы `Math_02`. Определите оптимальные условия численного моделирования.

2. Разместите пластину на пути маятника в другом месте так, чтобы при столкновении маятника с ней направление его скорости менялось на противоположное. Промоделируйте колебания в этом случае, изменяя величину коэффициента восстановления `c`.

3. Исследуйте колебания маятника при различных положениях пластины и параметрах маятника. Сделайте вывод.

4. Изучите программу `Math_03`, моделирующую столкновение маятника с бруском, расположенным на горизонтальной поверхности.

5. Самостоятельно сформулируйте задачу исследования системы, смоделированной в этой программе, и проведите необходимые исследования. Опишите наблюдаемые явления.

6. Изучите программу Math_04, моделирующую столкновение двух математических маятников, движущихся в одной плоскости.

7. Самостоятельно сформулируйте задачу исследования системы, смоделированной в программе Math_04. Проведите необходимые исследования. Напишите заключение.

```
{Программа
СТОЛКНОВЕНИЕ МАТЕМАТИЧЕСКОГО МАЯТНИКА С ТЕЛОМ,
РАСПОЛОЖЕННЫМ НА ГОРИЗОНТАЛЬНОЙ ПОВЕРХНОСТИ}
Program Math_03;

{Подключение модулей подпрограмм}
Uses Crt, Graph;

{Описание констант}
Const phi0=20;  {Начальный угол отклонения маятника}
      w0=0;     {Начальная угловая скорость маятника}
      l=400;    {Длина нити}
      m1=1;     {Масса шарика}
      r=8;      {Радиус шарика}
      light=2;  {Радиус блика}
      k=0.02;   {Коэффициент сопротивления}
      xp=320;   {Координата x точки подвеса}
      yp=10;    {Координата y точки подвеса}
      c=0.9;    {Коэффициент восстановления}
      m2=2;     {Масса бруска}
      mu=0.03;  {Коэффициент трения о горизонтальную поверх-
ность}
      g=20;     {Ускорение свободного падения}
      tau=0.2;  {Шаг по времени}
{Описание переменных}
Var x1, y1, xx1, yy1, x2, y2, xx2, yy2, w, eps, phi, v, a:
Real;
    Blow: Integer;

{Процедура инициализации графического режима}
Procedure OpenGraph;
Var Driver, Mode, ErrorCode: Integer;
Begin
  Driver:=Detect;
  InitGraph(Driver, Mode, 'c:\bp\bgi');
  ErrorCode:=GraphResult;
  If ErrorCode <> grOK Then Halt(1);
End;

{Процедура исключения мерцаний изображений}
Procedure Glimmer;
```

```
Begin
  Repeat
  Until Port[$3da] And 8 <> 0;
End;

{Процедура вывода изображений движущихся тел на экран}
Procedure Display;
Begin
  Glimmer;
  SetColor(Black);
  Line(xp, yp, Round(xx1), Round(yy1));
  SetFillStyle(SolidFill, Black);
  FillEllipse(Round(xx1), Round(yy1), r, r);
  Bar(Round(xx2), Round(yy2), Round(xx2+70), Round(yy2-2*r-
13));
  SetColor(DarkGray);
  Line(xp, yp, Round(x1), Round(y1));
  SetColor(DarkGray);
SetFillStyle(SolidFill, DarkGray);
  FillEllipse(Round(x1), Round(y1), r, r);
  SetColor(LightGray);
  SetFillStyle(SolidFill, White);
  FillEllipse(Round(x1)-3, Round(y1)-3, light, light);
  SetFillStyle(SolidFill, LightGray);
  Bar(Round(x2), Round(y2), Round(x2+70), Round(y2-2*r-13));
End;

{Основная часть программы}
Begin
  OpenGraph;
  SetFillStyle(SolidFill, DarkGray);
  x2:=xp+r;
  y2:=yp+l+2*r-1;
  Bar(xp-20, yp-5, xp+20, yp-1);
  Bar(0, Round(y2+1), 640, Round(y2+6));
  SetFillStyle(SolidFill, LightGray);
  Bar(Round(x2), Round(y2), Round(x2+70), Round(y2-2*r-3));
  a:=0;
  v:=0;
  w:=w0;
  phi:=phi0*Pi/180;
  x1:=xp-l*sin(phi);
  y1:=yp+l*cos(phi);
  Blow:=0;
  Repeat
    Begin
      If ((xp-x1)<=0) And (Blow=0) Then
        Begin
          phi:=0;
          w:=-c*w;
          v:=c*m1*w*l/m2;
          a:=-mu*g;
          Blow:=1;
```

```
        End;
      xx1:=x1;
      yy1:=y1;
      xx2:=x2;
      yy2:=y2;
      eps:=-k*w/m1-g*sin(phi)/l;
      w:=w+eps*tau;
      phi:=phi+w*tau;
      x1:=xp-l*sin(phi);
      y1:=yp+l*cos(phi);
      If v>0.2 Then
        Begin
          v:=v+a*tau;
          x2:=x2+v*tau;
        End;
      Display;
    End;
  Until KeyPressed;
  CloseGraph;
End.


{Программа
СТОЛКНОВЕНИЕ ДВУХ МАТЕМАТИЧЕСКИХ МАЯТНИКОВ}
Program Math_04;

{Подключение модулей подпрограмм}
Uses Crt, Graph;

{Описание констант}
Const phi01=70; {Начальный угол отклонения первого маятника}
      phi02=0;  {Начальный угол отклонения второго маятника}
      w01=0;    {Начальная угловая скорость первого маятника}
      w02=0;    {Начальная угловая скорость второго маятника}
      l=350;    {Длина нитей маятников}
      m1=1;     {Масса первого шарика}
      m2=1;     {Масса второго шарика}
      r=8;      {Радиусы шариков}
      light=3;  {Радиусы бликов}
      k1=0.02;  {Коэффициент сопротивления первого маятника}
      k2=0.01;  {Коэффициент сопротивления второго маятника}
      xp=320;   {Координата x точки подвеса первого маятника}
      yp=40;    {Координата y точки подвеса первого маятника}
      c=0.95;   {Коэффициент восстановления}
      g=9.8;    {Ускорение свободного падения}
      tau=0.2;  {Шаг по времени}

{Описание переменных}
Var x1, y1, w1, eps1, phi1, x2, y2, w2, eps2, phi2, w: Real;
{Процедура инициализации графического режима}
Procedure OpenGraph;
Var Driver, Mode, ErrorCode: Integer;
Begin
```

```
  Driver:=Detect;
  InitGraph(Driver, Mode, 'c:\bp\bgi');
  ErrorCode:=GraphResult;
  If ErrorCode <> grOK Then Halt(1);
End;

{Процедура исключения мерцаний изображений}
Procedure Glimmer;
Begin
  Repeat
  Until Port[$3da] And 8 <> 0;
End;

{Процедура вывода изображений маятников на экран}
Procedure Display;
Begin
  SetColor(DarkGray);
  SetFillStyle(SolidFill, DarkGray);
  Bar(xp-20, yp-5, xp+2*r+20, yp-1);
  Line(xp, yp, Round(x1), Round(y1));
  Line(xp+2*r+1, yp, Round(x2), Round(y2));
  SetColor(DarkGray);
  SetFillStyle(SolidFill, DarkGray);
  FillEllipse(Round(x1), Round(y1), r, r);
  FillEllipse(Round(x2), Round(y2), r, r);
  SetColor(LightGray);
  SetFillStyle(SolidFill, White);
  FillEllipse(Round(x1)-3, Round(y1)-3, light, light);
  FillEllipse(Round(x2)-3, Round(y2)-3, light, light);
  Glimmer;
  ClearDevice;
End;

{Основная часть программы}
Begin
  OpenGraph;
  w1:=w01;
  phi1:=-phi01*Pi/180;
  x1:=xp+l*sin(phi1);
  y1:=yp+l*cos(phi1);
  w2:=w02;
  phi2:=-phi02*Pi/180;
  x2:=xp+2*r+l*sin(phi2)+1;
  y2:=yp+l*cos(phi2);
  Repeat
    Begin
      If ((x2-x1-2*r)<=0) And (c<>0) Then
        Begin
          w:=w1;
          w1:=c*(2*m2*w2+(m1-m2)*w1)/(m1+m2);
          w2:=c*(2*m1*w+(m2-m1)*w2)/(m1+m2);
        End;
```

```
      If ((x2-x1-2*r)<=0) And (c=0) Then
        Begin
          w1:=(m1*w1+m2*w2)/(m1+m2);
          w2:=w1;
        End;
      eps1:=-k1*w1/m1-g*sin(phi1)/l;
      w1:=w1+eps1*tau;
      phi1:=phi1+w1*tau;
      x1:=xp+l*sin(phi1);
      y1:=yp+l*cos(phi1);
      eps2:=-k2*w2/m2-g*sin(phi2)/l;
      w2:=w2+eps2*tau;
      phi2:=phi2+w2*tau;
      x2:=xp+2*r+l*sin(phi2)+1;
      y2:=yp+l*cos(phi2);
      Display;
    End;
  Until KeyPressed;
  CloseGraph;
End.
```

Представляют интерес колебания, возникающие в системе при периодическом изменении ее параметров. Ниже приведена программа Math_05, которая моделирует такой случай – математический маятник с переменной длиной. Более сложным примером являются качели с человеком, старающимся их раскачать. Такой маятник, конечно же, не является математическим, он – физический.

```
{Программа
СВОБОДНЫЕ КОЛЕБАНИЯ МАТЕМАТИЧЕСКОГО МАЯТНИКА
С ИЗМЕНЯЮЩЕЙСЯ ТОЧКОЙ ПОДВЕСА}
Program Math_05;

{Подключение модулей подпрограмм}
Uses Crt, Graph;

{Описание констант}
Const phi0=50; {Начальный угол отклонения маятника}
      w0=0;    {Начальная угловая скорость маятника}
      ls=350;  {Длина нити}
      r=8;     {Радиус шарика}
      light=2; {Радиус блика}
      m=1;     {Масса шарика}
      k=0.01;  {Коэффициент сопротивления}
      xp0=320; {Начальная координата x точки подвеса}
      yp0=50;  {Начальная координата y точки подвеса}
      ys=100;  {Координата y перекладины}
      g=10;    {Ускорение свободного падения}
      tau=0.2; {Шаг по времени}
```

```
{Описание переменных}
Var x, y, w, eps, phi, l: Real;
    xp, yp: Integer;

{Процедура инициализации графического режима}
Procedure OpenGraph;
Var Driver, Mode, ErrorCode: Integer;
Begin
  Driver:=Detect;
  InitGraph(Driver, Mode, 'c:\bp\bgi');
  ErrorCode:=GraphResult;
  If ErrorCode <> grOK Then Halt(1);
End;

{Процедура исключения мерцаний изображений}
Procedure Glimmer;
Begin
  Repeat
  Until Port[$3da] And 8 <> 0;
End;

{Процедура вывода изображения колеблющегося тела на экран}
Procedure Display;
Begin
  SetColor(DarkGray);
  SetFillStyle(SolidFill, DarkGray);
  Bar(xp0-20, yp0-5, xp0+20, yp0-1);
  If yp=ys Then Line(xp0, yp0, xp0, ys);
  Line(xp, yp, Round(x), Round(y));
  FillEllipse(Round(x), Round(y), r, r);
  SetColor(LightGray);
  SetFillStyle(SolidFill, White);
  FillEllipse(Round(x)-3, Round(y)-3, light, light);
  SetFillStyle(SolidFill, LightGray);
  FillEllipse(xp0+2, ys-2, 2, 2);
  Glimmer;
  ClearDevice;
End;

{Основная часть программы}
Begin
  OpenGraph;
  xp:=xp0;
  yp:=yp0;
  l:=ls;
  w:=w0;
  phi:=-phi0*Pi/180;
  x:=xp+l*sin(phi);
  y:=yp+l*cos(phi);
  Repeat
    Begin
      If (Round(phi*180/Pi)=0) And (yp=yp0) And (eps<0) Then
        Begin
```

```
          yp:=ys;
          l:=l+yp0-ys;
        End;
      If (Round(phi*180/Pi)=0) And (yp=ys) And (eps>0) Then
        Begin
          yp:=yp0;
          l:=ls;
        End;
      eps:=-k*w/m-g*sin(phi)/l;
      w:=w+eps*tau;
      phi:=phi+w*tau;
      x:=xp+l*sin(phi);
      y:=yp+l*cos(phi);
      Display;
    End;
  Until KeyPressed;
  CloseGraph;
End.
```

Задания для самостоятельного исследования

1. Определите оптимальные условия моделирования в программе `Math_05`.

2. Выясните, что называют параметрическим резонансом, и попробуйте промоделировать это явление с помощью программы `Math_05`. Опишите результаты этого моделирования. Сделайте вывод.

3.4. КОЛЕБАНИЯ ТЕЛА, ПОДОБНЫЕ КОЛЕБАНИЯМ КРУГОВОГО МАТЕМАТИЧЕСКОГО МАЯТНИКА

```
{Программа
КОЛЕБАНИЯ ШАРИКА В ЖЕЛОБЕ}
Program Ball_01;
{Подключение модулей подпрограмм}
Uses Crt, Graph;
{Описание констант}
Const phi0=80;  {Начальный угол отклонения шарика}
      w0=0;     {Начальная угловая скорость шарика}
      l=300;    {Радиус желоба}
      m=1;      {Масса шарика}
      r=14;     {Радиус тела}
      light=4;  {Радиус блика}
      mu=0.005; {Коэффициент трения}
      xp=320;   {Координата x центра желоба}
      yp=10;    {Координата y центра желоба}
      g=10;     {Ускорение свободного падения}
      tau=0.3;  {Шаг по времени}
{Описание переменных}
```

```
Var x, y, xx, yy, w, eps, phi: Real;
{Процедура инициализации графического режима}
Procedure OpenGraph;
Var Driver, Mode, ErrorCode: Integer;
Begin
  Driver:=Detect;
  InitGraph(Driver, Mode, 'c:\bp\bgi');
  ErrorCode:=GraphResult;
  If ErrorCode <> grOK Then Halt(1);
End;
{Процедура исключения мерцаний изображений}
Procedure Glimmer;
Begin
  Repeat
  Until Port[$3da] And 8 <> 0;
End;

{Процедура вывода изображения колеблющегося тела на экран}
Procedure Display;
Begin
  Glimmer;
  SetColor(Black);
  SetFillStyle(SolidFill, Black);
  FillEllipse(Round(xx), Round(yy), r, r);
  SetColor(DarkGray);
  SetLineStyle(SolidLn, 0, NormWidth);
  SetFillStyle(SolidFill, DarkGray);
  FillEllipse(Round(x), Round(y), r, r);
  SetColor(LightGray);
  SetFillStyle(SolidFill, White);
  FillEllipse(Round(x)-5, Round(y)-5, light, light);
End;

{Основная часть программы}
Begin
  OpenGraph;
  SetFillStyle(SolidFill, DarkGray);
  Bar(0, 0, 640, 480);
  SetColor(Black);
  SetFillStyle(SolidFill, Black);
  FillEllipse(Round(xp), Round(yp), l+r, l+r);
  w:=w0;
  phi:=phi0*Pi/180;
  x:=xp+l*sin(phi);
  y:=yp+l*cos(phi);
  Repeat
    Begin
      xx:=x;
      yy:=y;
```

```
      eps:=-mu*w/m-g*sin(phi)/l;
      w:=w+eps*tau;
      phi:=phi+w*tau;
      x:=xp+l*sin(phi);
      y:=yp+l*cos(phi);
      Display;
    End;
  Until KeyPressed;
  CloseGraph;
End.
```

СПИСОК ЛИТЕРАТУРЫ

1. *Бать, М. И.* Теоретическая механика в примерах и задачах: учебное пособие для втузов: в 3 т. Т. 2. Динамика / М. И. Бать, Г. Ю. Джанелидзе, А. С. Кельзон. – 8-е изд., перераб. – М.: Наука; Гл. ред. физ.-мат. лит., 1991. – 640 с.

2. *Карлов, Н. В.* Колебания, волны, структуры / Н. В. Карлов, Н. А. Кириченко. – М.: Физматлит, 2003. – 496 с.

3. *Малов, Н. Н.* Основы теории колебаний: пособие для учителей / Н. Н. Малов. – М.: Просвещение, 1971. – 200 с.

4. *Матвеев, Н. М.* Дифференциальные уравнения: учебное пособие для студентов педагогических институтов физико-математических специальностей / Н. М. Матвеев. – М.: Просвещение, 1988. – 256 с.

5. *Несис, Е. И.* Методы математической физики / Е. И. Несис. – М.: Просвещение, 1977. – 200 с.

6. *Ращиков, В. И.* Численные методы решения физических задач: учебное пособие / В. И. Ращиков, А. С. Рошаль. – СПб.: Изд-во «Лань», 2005. – 208 с.

7. *Сергиевский, М. В.* Турбо Паскаль 7.0: язык, среда программирования / М. В. Сергиевский, А. В. Шалашов. – М.: Машиностроение, 1994. – 256 с.

8. *Тарг, С. М.* Краткий курс теоретической физики: учебник для вузов. – 11-е изд., испр. – М.: Высш. шк., 1995. – 416 с.

9. *Шелест, А. Е.* Микрокалькуляторы в физике / А. Е. Шелест. – М.: Наука; Гл. ред. физ.-мат. лит., 1988. – 272 с.

10. *Шуп, Т. Е.* Прикладные численные методы в физике и технике / Т. Е. Шуп; под ред. С. П. Меркурьева; пер. с англ. С. Ю. Славянова. – М.: Высш. шк., 1990. – 256 с.

11. *Фараонов, В. В.* Программирование на персональных ЭВМ в среде Турбо Паскаль / В. В. Фараонов. – М.: Высшая школа, 1991. – 580 с.

12. *Фараонов, В. В.* Турбо Паскаль: в 3 кн. Кн. 1. Основы Турбо Паскаля / В. В. Фараонов. – М.: МГТУ, 1992. – 304 с.

13. *Федоров, А.* Особенности программирования на Borland Pascal / А. Федоров. – Киев: Диалектика, 1994. – 144 с.

14. Физика: энциклопедия / под ред. Ю. В. Прохорова. – М.: Большая Российская энциклопедия, 2003. – 944 с.

15. *Хайкин, С. Э.* Физические основы механики / С. Э. Хайкин. – 2-е изд., испр. и доп. – М.: Наука; Гл. ред. физ.-мат. лит., 1971. – 752 с.

16. *Хершель, Р.* Турбо Паскаль 4.0/5.0 / Р. Хершель. – 2-е изд., перераб. – Вологда: МП «МИК», 1991. – 344 с.

17. ЭВМ в курсе общей физики / П. С. Булкин, Б. И. Волков, М. А. Воронцов и др.; под ред. А. Н. Матвеева. – М.: Изд-во МГУ, 1982. – 232 с.

Учебное издание

Олег Евгеньевич Данилов
Анна Юрьевна Трефилова

КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ КОЛЕБАТЕЛЬНОГО ДВИЖЕНИЯ

Численные методы решения физических задач

Учебно-методическое пособие

Редактор *Л. В. Ларионова*
Оригинал-макет: *Е. Н. Шулятьева*
Дизайн обложки: *И. Н. Пермяков*

Подписано в печать 16.05.2012. Напечатано на ризографе. Формат 60×84 $^{1}/_{16}$.
Усл. печ. л. 2,5. Уч.-изд. л. 2,0. Тираж 75 экз. Заказ № 3247 – 2012.

ФГБОУ ВПО «Глазовский государственный педагогический институт имени В. Г. Короленко»

427621, Удмуртская Республика, г. Глазов, ул. Первомайская, д. 25
Тел./факс: 8 (34141) 5-60-09, email: izdat@mail.ru