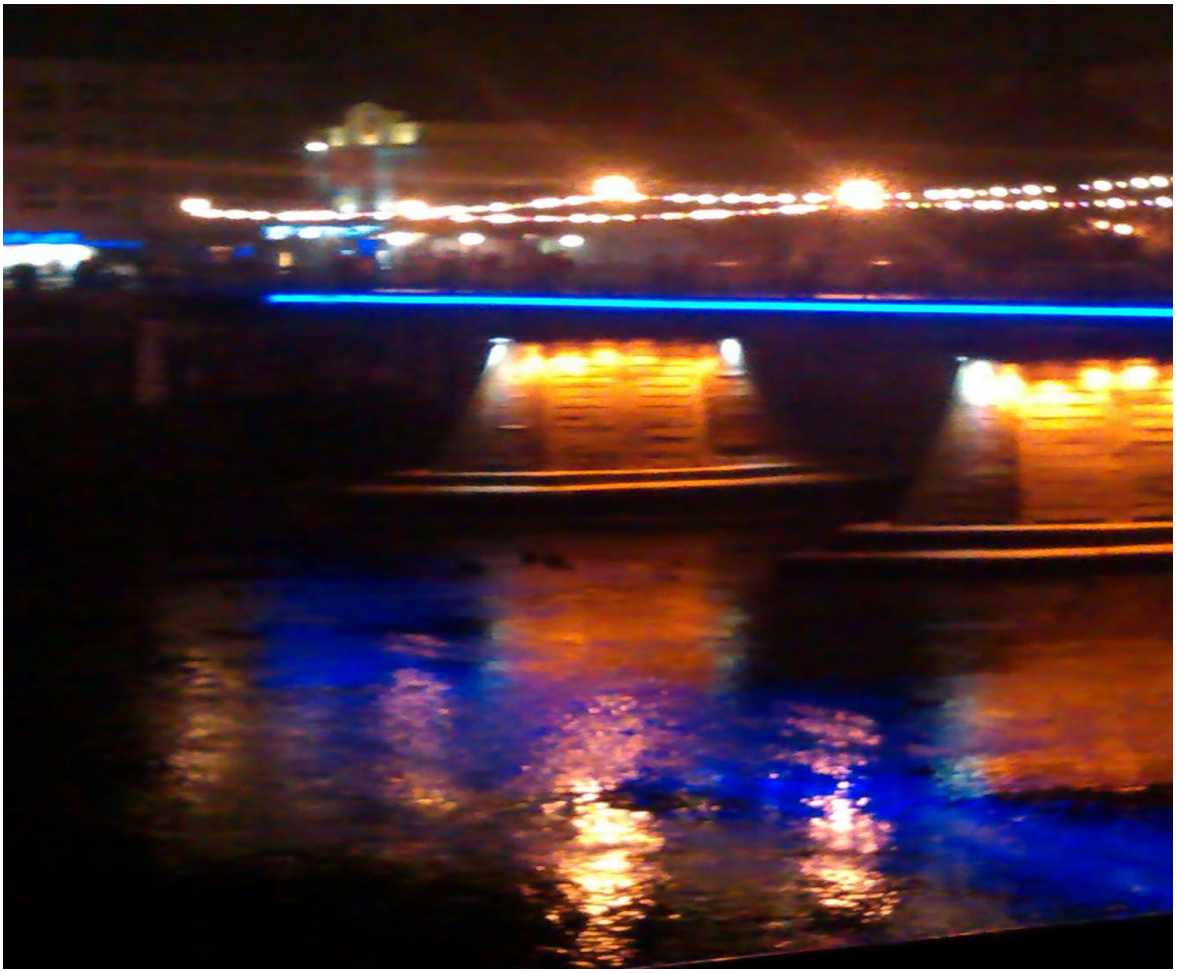
НАТАЛІЯ ЮРКОВИЧ

ВЛАДІМІР ШЕБЕНЬ

МИХАЙЛО МАР'ЯН

КОМП'ЮТЕРНЕ МОДЕЛЮВАННЯ ТА ІННОВАЦІЙНІ ПІДХОДИ В ФІЗИЦІ: ОПТИКА



University of Prešov in Prešov,
Faculty of Humanities and Natural Sciences

Prešovská univerzita v Prešove
Fakulta humanitných a prírodných vied

Title / Názov: **COMPUTER MODELING AND INNOVATIVE APPROACHES IN PHYSICS: OPTICS**

Počítačové modelovanie a inovatívne prístupy vo fyzike: optika

Authors / Autori: **Ass Prof. Nataliya Yurkovych, PhD.**/
Doc. Natalia Jurkovič, PhD.
Dr.h.c. doc. PaedDr. Vladimir Seben, PhD./
Dr.h.c. doc. PaedDr. Vladimír Šebeň, PhD.
Prof. Mykhaylo Mar'yan, DrSc./
Prof. Michajlo Marian, DrSc.

Reviewers / Recenzenti: Doctor of Sciences (Physics), prof. S. Kekenesi, Debrecen University (Debrecen, Hungary);
Doctor of Sciences (Physics), prof. L. Kharkhalis, Uzhgorod National University (Uzhgorod, Ukraine);
RNDr. S. Iľkovič, PhD, *Prešovska univerzita v Prešov*e (Prešov, Slovakia)

.

Vydala: © Prešovská univerzita v Prešove, 2017
Vydanie: Prvé

ISBN: 978-80-555-1770-4

Nataliya Yurkovych, Vladimir Seben, Mykhaylo Mar'yan. ***Computer modeling and innovative approaches in physics: optics.***– Presov: University of Presov Publishing, 2017. – 112 p.

Innovative information products of teaching physics, based on the use of the computer modeling and the synergy are approbated. The expedience of application of the object-oriented programming and visualization of dependencies in mastering the material is shown not only in the sections of physics, but also for adjacent areas. Recommended for teaching physics in high schools, for teachers and students.

Keywords: computer modeling, object-oriented programming, visualization dependencies, synergy of interdisciplinary science, innovative approaches, optical phenomena.

Natalia Jurkovič, Vladimír Šebeň, Michajlo Marian. ***Počítačové modelovanie a inovatívne prístupy vo fyzike: optika***. - Prešov: University of Presov Publishing, 2017. - 112 s.

V publikácii sú prezentované o*svedčené inovatívne informačné produkty výučby fyziky, ktoré sú založené na využití počítačového modelovania a synergie. Je demonštrovaná účelnosť využitia objektovo orientovaného programovania a vizualizácie funkčných závislostí nielen vo fyzike, ale aj v príbuzných odboroch. Publikácia je odporúčaná ako metodický materiál pre učiteľov a študentov fyziky na vysokých školách.*

Kľúčové slová: počítačové modelovanie, objektovo orientované programovanie, vizualizácia závislostí, synergia interdisciplinárne vzťahy, inovatívne prístupy, optické javy.

Наталія Юркович, Владімір Шебень, Михайло Мар'ян. ***Комп'ютерне моделювання та інноваційні підходи в фізиці: оптика.*** - Presov: University of Presov Publishing, 2017. – 112 с.

Апробовані інноваційні інформаційні засоби викладання фізики, які базуються на використанні комп'ютерного моделювання та синергетики. Показана доцільність застосування об'єктно-орієнтованого програмування та візуалізації залежностей при засвоєнні матеріалу не тільки в розділах фізики, але і в суміжних з нею областях. Рекомендовано при викладанні фізики у вузах для викладачів, аспірантів та студентів.

Ключові слова: комп'ютерне моделювання, об'єктно-орієнтоване програмування, візуалізація залежностей, синергія міждисциплінарних наук, інноваційні підходи, оптичні явища.

Reviewers: Doctor of Sciences (Physics), prof. S. Kekenesi, Debrecen University (Debrecen, Hungary);
Doctor of Sciences (Physics), prof. L. Kharkhalis, Uzhgorod National University (Uzhgorod, Ukraine);
RNDr. S. Iľkovič, PhD, *Prešovska univerzita v Prešov*e (Prešov, Slovakia)

Ms. Nataliya Yurkovych, PhD
Department of solid-state electronics, information security
Faculty of Physics
Uzhhorod National University
Voloshyna str. 54
Uzhhorod, Ukraine, 88000
E-mail: yurkovich@ukr.net
Web site: http://www.uzhnu.edu.ua

Mr. Vladimir Seben, PhD.
Department of physics, mathematics and technics
Faculty of Humanities and Natural Science
University of Presov
Ul. 17.Novembra 1
08116 Presov
Slovak republic
E-mail: vladimir.seben@unipo.sk
Web site: http://unipo.sk

Mr. Mykhaylo Mar'yan, Doctor of Sciences (Physics)
Department of solid-state electronics, information security
Faculty of Physics
Uzhhorod National University
Voloshyna str. 54
Uzhhorod, Ukraine, 88000
E-mail: mclmaryan@gmail.com
Web site: http://www.uzhnu.edu.ua

ЗМІСТ

ПЕРЕДМОВА

Комп'ютерне моделювання є інноваційним засобом розв'язку прикладних науково-технічних задач у фізиці. Має важливе використання у навчальному процесі разом з потужними у пізнавальному аспекті інформаційними технологіями. Імплементація комп'ютерного моделювання майбутніми вчителями фізики відкриває хороші можливості використання сучасних технологій у їх науковій та навчальній діяльності для реалізації міжпредметних синергетичних зв'язків інформатики, математики, фізики та інших предметів (Nicolis, Prigogin, 1989). Воно включає покроковий аналіз фізичного явища чи процесу, побудову фізичної моделі (абстрагування від несуттєвих впливів, вибір законів, які описують відповідні процеси), створення математичної моделі, реалізацію її засобами інформаційних технологій, проведення відповідних обчислень та аналіз результатів (Huffman,1997).

Першою проблемою, впроваджуючи елементи комп'ютерного моделювання під час вивченні фізики, є вибір інструментів його реалізації. Сучасні інформаційні технології дозволяють використовувати мови програмування високого рівня. За останні роки опубліковано багато книг і статей, де демонструють розв'язання фізичних задач цим способом. Спеціалізовані програмні продукти дають можливість автоматизації математичних обчислень, що є серйозним кроком вперед у галузі комп'ютерного моделювання (Гулд, Тобочник, 1990). Інформаційні технології мають застосування не лише в дисциплінах, що традиційно застосовують комп'ютери (інформатика, комп'ютерне моделювання, числові методи), а й у класичних навчальних дисциплінах (Мар'ян, Шебень, Юркович, 2016).

Важливим у реалізації комп'ютерних моделей є отримання кінцевої вихідної інформації у графічному представленні. Бо особливості психіки і фізіології людини дозволяють її миттєво проаналізувати, провести власні асоціації і ідентифікувати графічну інформацію на відміну від сухого набору формул і цифр. Можливість проводити аналіз отриманих графічні залежності

між різними величинами – це необхідний елемент фізичної освіти.

Дискутуючии про переваги та недоліки імплементації інноваційних освітніх технологій, справжнім «індикатором» їх застосування є учні і студенти. Педагогічний досвід засвідчує, що якісну освіту можна здобути лише в адекватному діалозі учня з учителем, студента з викладачем. Ось чому думка учня та студента так багато важить в оцінці новітніх інформаційно-освітніх технологій (Kuo, Hull, Gupta, Elby, 2013).

РОЗДІЛ І. ІННОВАЦІЙНІ ТЕХНОЛОГІЇ ТА МЕТОДИ

1.1. Комп'ютерне моделювання електромагнітних явищ.

Всі комп'ютерні моделі реалізуються відкритою або закритою системами (Nicolis, Prigogin, 1989). Імплементація методів моделювання при вивченні фізики визначає цілий ряд характерних їх застосувань. Звертається увага на певні ідеалізації, які застосовуються для опису відповідних процесів (Гулд, Тобочник, 1990). До прикладу, в курсі фізики зустрічаємося з цілим рядом фізичних ідеалізацій: матеріальна точка, інерційна система відліку, ідеальний газ, кристалічна гратка. Багато з них важко уявити. Проте, існують ілюстративні - демонстраційно моделі, які досить точно описують такі процеси та явища. Інші навчальні моделі описуються відповідними математичними співвідношеннями, логічними представленнями. Це навчально-еврістичні моделі, що включають в себе схеми, системи рівнянь, матриці коефіцієнтів та ін. (Cavinato, Giliberti, Perotti, 2015).

Кожна з цих навчальних моделей може описуватися певними математичними представленнями та функціями. Математичні співвідношення добре описують фізичні явища та процеси: статистична фізика описує процеси в термодинамічних системах, які однозначно визначаються математичними операторами; поведінка електромагнітних хвиль описується рівнянням хвильової функції Шредінгера; теорія відносності, опираючись на постулати Ейнштейна, використовує перетворення Лоренца для інерційних систем. У цілому, можна говорити, що математичні моделі несуть як смислове, так і пізнавальне значення та можуть бути складовою комп'ютерної моделі, реалізованої в педагогічних програмних засобах (Leung, Terrana, Jerzak, 2016).

Приклад застосування комп'ютерного моделювання для електромагнітних явищ показано на рис.1.1.

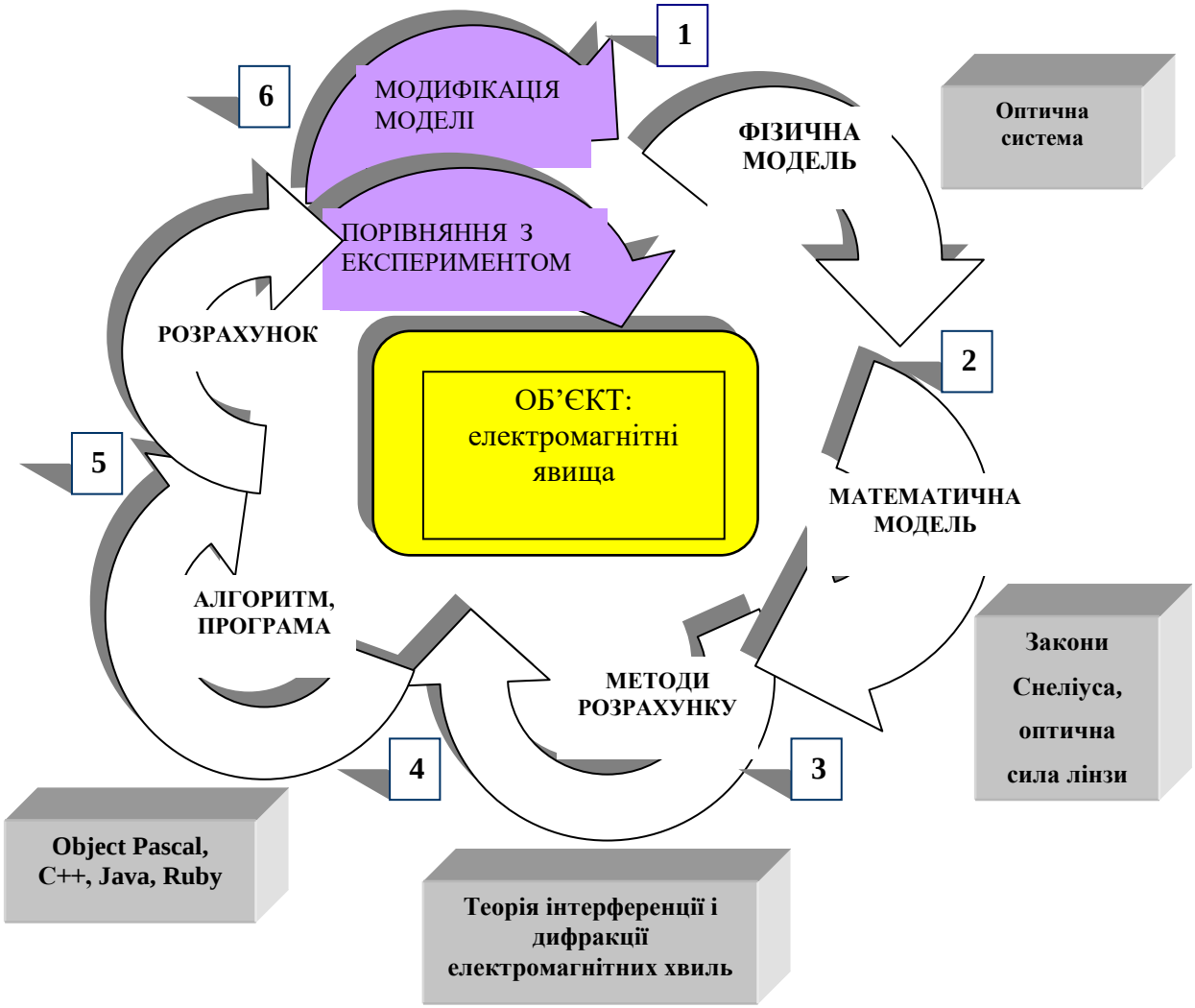


Рис.1.1. Приклад проведення комп'ютерного моделювання для електромагнітних явищ.

Застосовування комп'ютерного моделювання, як видно з рис.1.1, передбачає ітераційність та «формування» моделі в циклі: 1-фізична модель, 2-математична модель, 3-методи розрахунку, 4-алгоритм та програма розрахунку моделі, 5-проведення тестування та дослідження моделі, 6-порівняння результатів розрахунку з експериментальними даними та наступне уточнення моделі. Даний цикл повторюється необхідне число разів, наближаючись до реального об'єкту (явища). Розглянемо кожний з етапів проведення комп'ютерного моделювання (рис.1.1). Перший етап – фізична модель. Фізична модель для електромагнітних явищ – це модель електромагнітних хвиль та

середовищ, в яких вони поширюються. Другий і третій етапи - це математичні моделі та методи їх розрахунку. Четвертий – використання сучасних алгоритмічних мов програмування Object Pascal, C++, Java, Ruby, які побудовані на принципах об'єктно-орієнтованого програмування та новітніх інформаційних технологіях RAD (швидка розробка додатків), VCL (бібліотеки візуальних компонент), DLL (динамічно зв'язаних бібліотек), OLE (взаємопов'язане переміщення додатків). П'ятий та шостий етапи – тестування та відладка програми, проведення розрахунку і порівняння з оптичним експериментом (Мар'ян, Шебень, Юркович, 2016).

1.2. Структура середовищ візуального програмування C++, Delphi.

Концепція об'єктно-орієнтованого програмування має на увазі, що основою управління процесом реалізації програми є передача повідомлень об'єктам. Тому об'єкти визначаються спільно з повідомленнями, на які вони повинні реагувати при виконанні програми (Фаронов, 2012). У цьому полягає головна відмінність об'єктно-орієнтованого програмування від процедурного програмування, де окремо певні структури даних передаються в процедури (функції) як параметри. Таким чином, об'єктно-орієнтована програма складається з об'єктів – окремих фрагментів коду, які взаємодіють один з одним через певні інтерфейси (Bucknall, 2001).

Розробка об'єктно-орієнтованих програм складається з таких послідовних кроків:

- визначення основних об'єктів, необхідних для вирішення даної задачі;
- визначення закритих даних (даних стану) для вибраних об'єктів;
- визначення другорядних об'єктів і їх закритих даних;
- визначення ієрархічної системи класів, що представляють вибрані об'єкти;
- визначення ключових повідомлень, які повинні обробляти об'єкти кожного класу;
- розробка послідовності виразів, які дозволяють вирішити поставлену задачу;

- очищення проекту, тобто усунення всіх допоміжних матеріалів, що використалися при проектуванні;
- кодування, відладка, компоновка і тестування.

Об'єктно-орієнтоване програмування дозволяє моделювати об'єкти певної області шляхом програмування їх змісту і поведінки в межах класу. Конструкція «клас» забезпечує механізм інкапсуляції для реалізації абстрактних типів даних. Інкапсуляція приховує подробиці внутрішньої реалізації типів, зовнішні операції і функції, допустимі для виконання над об'єктами цього типу (Фаронов, 2012).

Зображене на рис.1.2 IDE (Integrated Development Environment)-інтегроване середовище розробки володіє всіма необхідними інструментами програмування і є робочою областю створюваного розробником проекту. Тут створюються елементи управління і компоненти, вводиться код, налаштовується програма, а також можуть задаватися властивості проекту.

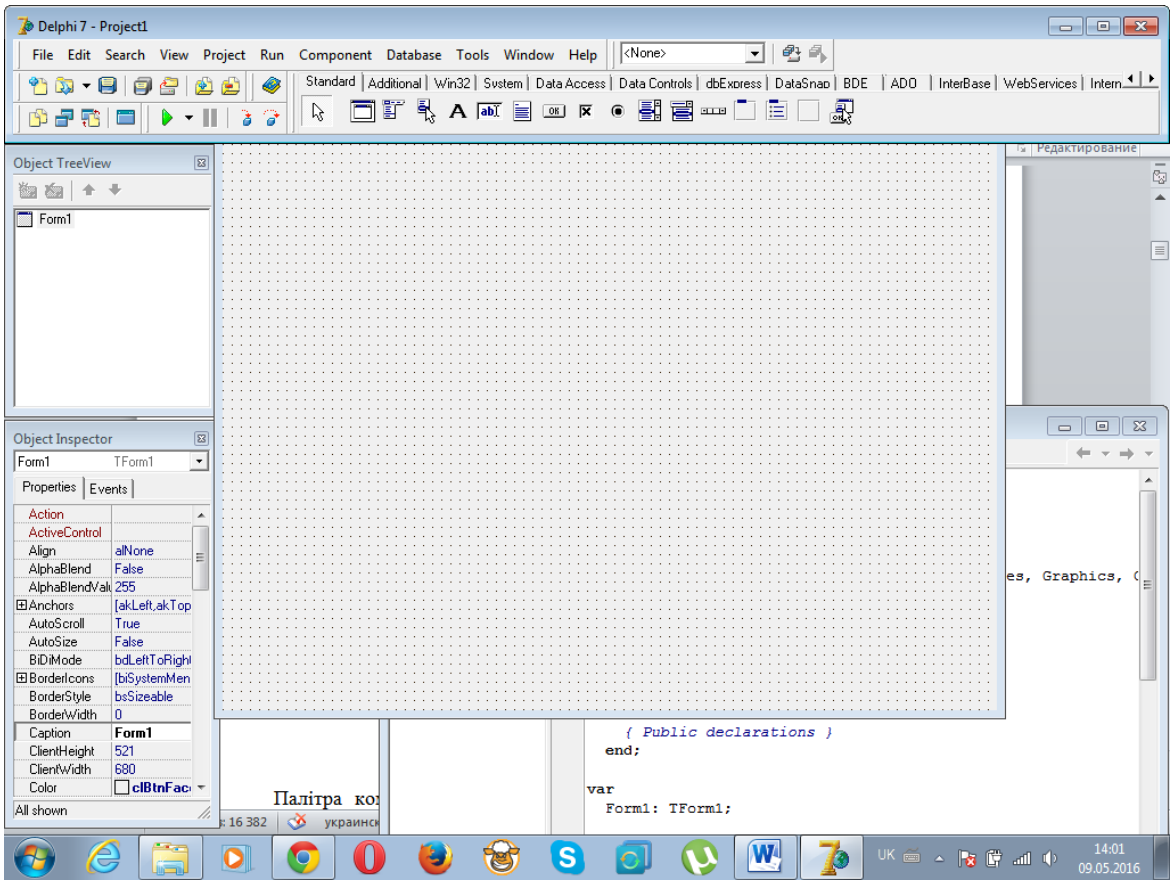


Рис.1.2. Загальне середовище Delphi (Фаронов, 2012).

Інтегроване середовище розробки Delphi складається з таких основних частин: головне меню; панель інструментів; палітра компонентів; редактор форм; редактор коду; інспектор об'єктів. У верхній частині вікна IDE-середовища розміщена панель головного меню. Панель інструментів в Delphi зображена на рис.1.3. Призначення розміщених на панелі кнопок можна дізнатися з ярличків, що з'являються, якщо навести курсор миші на відповідну кнопку і на деякий час затримати його.

Рис.1.3. Панель інструментів.

Палітра компонентів – це вітрина бібліотеки візуальних компонентів (Visual Component Library - VCL), зображена на рис.1.4. Вона дозволяє згрупувати компоненти відповідно до їх призначення. Імена компонентів, які відповідають тій або іншій піктограмі, можна дізнатись з ярличка, що з'являється, якщо затримати над цією піктограмою курсор миші. Якщо вибрати в палітрі компонент і натиснути клавішу F1, то буде показано довідку про тип даного компоненту (Юркович, 2010).

Рис.1.4. Палітра компонентів

Основою майже всіх проектів Delphi є форма, показана на рис.1.5.

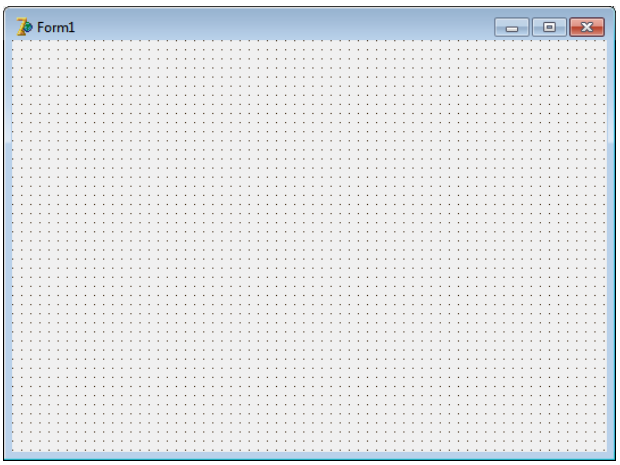


Рис.1.5. Вікно форми

Однією з найважливіших частин середовища Delphi є вікно редактора коду, яке показано на рис.1.6.

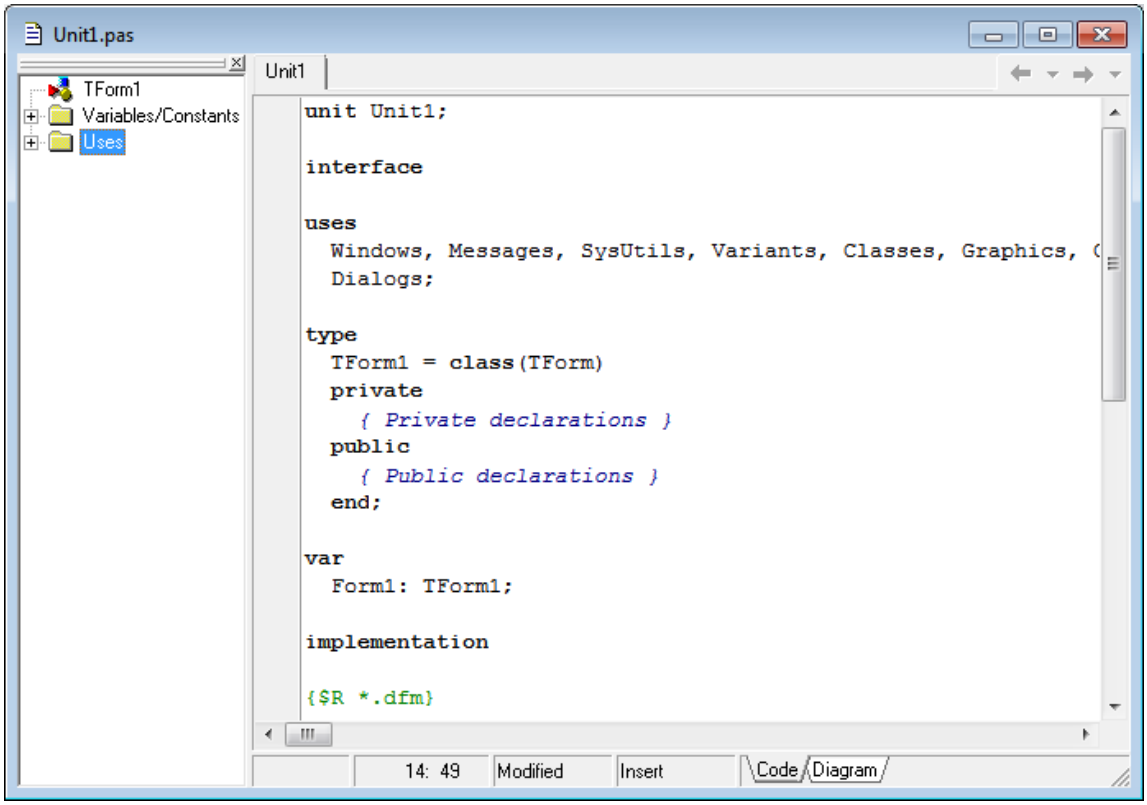
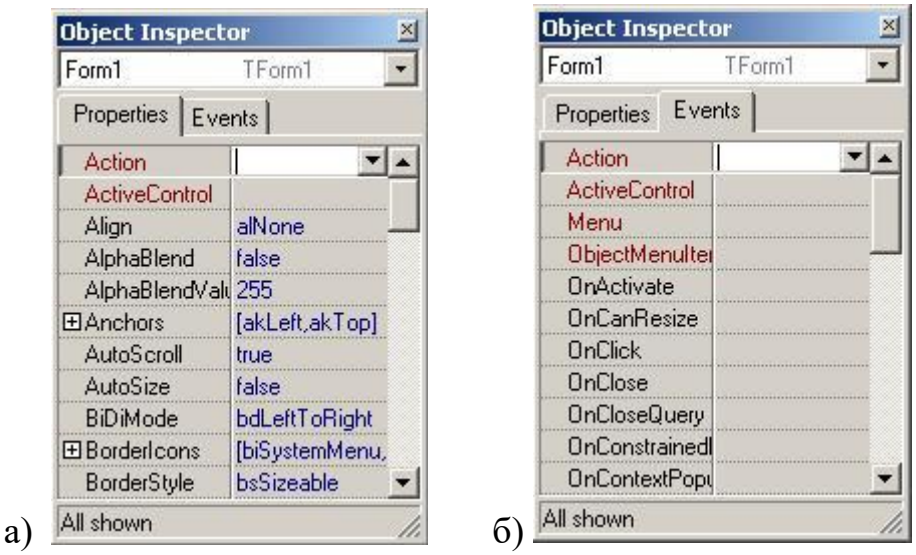


Рис.1.6. Вікно редактора коду.

Інспектор об'єктів забезпечує простий і зручний інтерфейс для зміни властивостей об'єктів Delphi і управління подіями, на які реагує об'єкт.



а) б)

Рис.1.7. Сторінка властивостей (а) та сторінка подій (б) інспектора об'єктів.

Сторінка властивостей (properties) інспектора об'єктів (рис.1.7(а)), показує властивості того об'єкту, який в даний момент виділений. Властивості є атрибутами компоненту, що визначають його зовнішній вигляд і поведінку. При визначенні властивостей компоненту під час проектування потрібно вибрати компонент на формі, відкрити сторінку властивостей в інспекторі об'єктів, вибрати потрібну властивість і змінити її за допомогою редактора властивостей (це може бути порожнє поле для введення тексту або числа, список, що розкривається, діалогова панель і т.д.).

Сторінка подій (events) складає другу частину інспектора об'єктів (рис.1.7(б)). На ній указані всі події, на які може реагувати вибраний об'єкт. Для того, щоб додати обробник подій, потрібно вибрати на формі за допомогою миші компонент, якому необхідний обробник подій, потім відкрити сторінку подій інспектора об'єктів і двічі натиснути лівою клавішею миші на колонці значень поряд з подією, щоб примусити Delphi згенерувати прототип обробника подій і показати його в редакторі коду. При цьому автоматично генерується текст порожньої функції, і редактор відкривається в тому місці, де слід вводити код. Далі потрібно ввести код, який повинен виконуватися при настанні події (Фаронов, 2012).

Система візуального об'єктно-орієнтованого проектування Delphi дозволяє:

- створювати закінчені програми для Windows різної спрямованості;
- швидко створювати професійний віконний інтерфейс для будь-яких додатків, написаних на будь-якій мові;
- створювати свої динамічно приєднуючі бібліотеки (DLL) компонентів, форм, функцій;
- створювати потужні системи роботи з локальними і віддаленими базами даних будь-яких типів;
- формувати і друкувати складні звіти;

- створювати довідкові системи (файли. hlp), як для своїх додатків, так і для будь-яких інших, з якими можна працювати не тільки з додатків, а й просто з Windows;
- створювати професійні програми установки для додатків Windows.

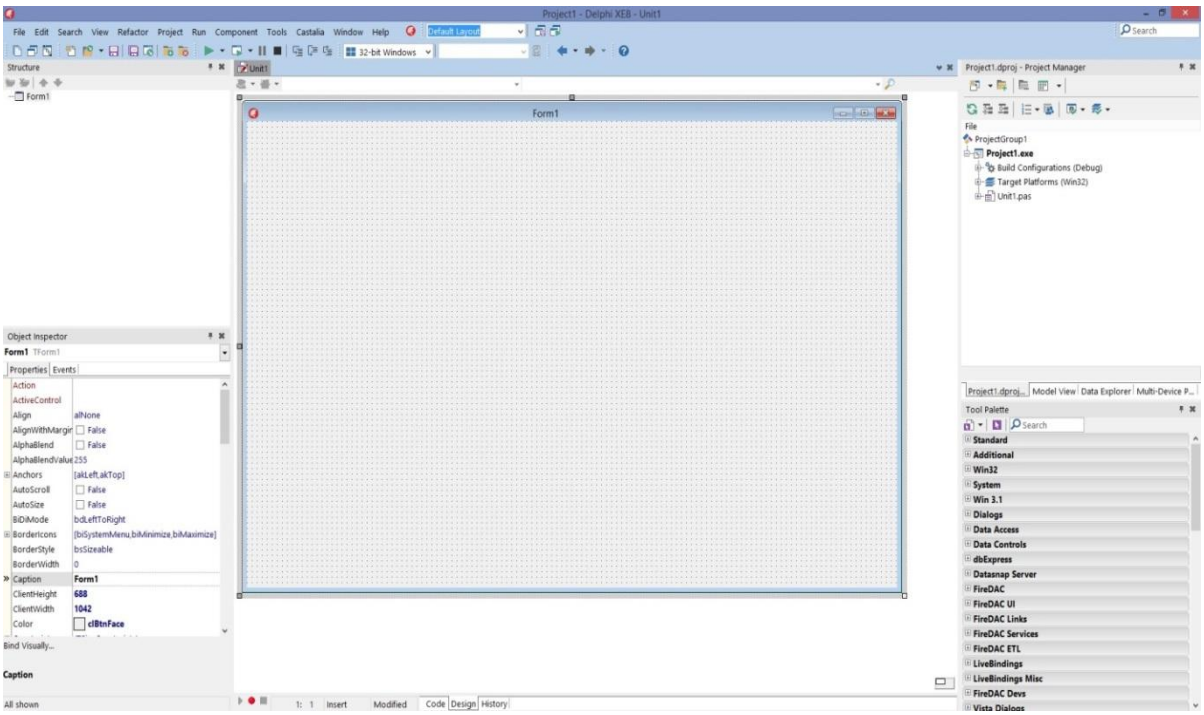


Рис. 1.8. Одна з останній версій середовища візуального програмування на мові Delphi – Embarcadero RAD Studio XE 8.

Компоненти можуть бути візуальними, видимими при роботі додатка, і невізуальними, які виконують деякі службові функції. Візуальні компоненти відразу відображаються на екрані в процесі проектування в такому ж вигляді, в якому їх побачить користувач під час виконання програми. Це дозволяє дуже легко вибрати місце їх розташування і дизайн - форму, розмір, оформлення, текст, колір і т. д. Невізуальні компоненти відображаються на формі в процесі проектування у вигляді значка, але користувачеві під час виконання вони не видимі, хоча і виконують корисну роботу (Cantu, 2008).

1.3. Компонента Chart та графічний редактор в середовищі Delphi. Задання графіків та їх візуалізація.

Objective Chart

Можливості Objective Chart зі створення діаграм і управління ними здатна задовольнити вимоги найвимогливішого розробника.

Перерахуємо основні особливості Objective Chart:

- можливість створення більше 30 типів діаграм;
- можливість швидко створити власні типи діаграм;
- можливість зберігати параметри діаграми в текстовому файлі або у вигляді графічного образу діаграми;
- доступ до даних з різних джерел, просте управління даними, що відображаються, шляхом доступу до колекцій об'єктів певних типів, можливість інтерактивного управління даними діаграми шляхом маніпулювання графічними елементами самої діаграми;
- наявність Chart Editor і Chart Wizard, призначених для використовування кінцевими користувачами і дозволяючих змінювати представлення діаграми і її дані.

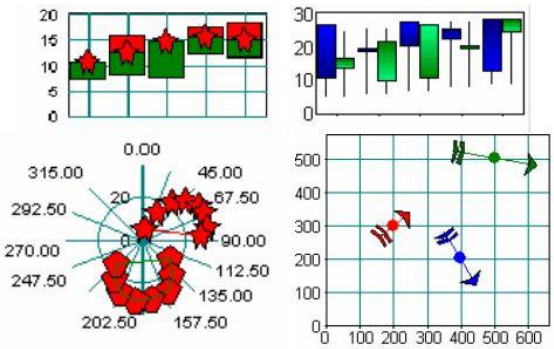


Рис.1.9. Приклади діаграм Objective Chart

Компоненти TeeChart знайомі всім користувачам останніх версій Delphi і C++Builder - вони входять до складу цих продуктів. Версія цих компонентів TeeChart-Pro, що поставляється окремо, володіє рядом додаткових можливостей. Зокрема, при використовуванні професійної версії можливе використовування редактора властивостей графіка не тільки з середовища розробки, але і з готового використовуючого його додатку (Фаронов, 2012).

Крім цього, користувачам професійної версії продукту доступні деякі додаткові типи графіків: «Candle», «ErrorBar», «Volume», «Surface», «Polar» (рис.1.10-1.11).

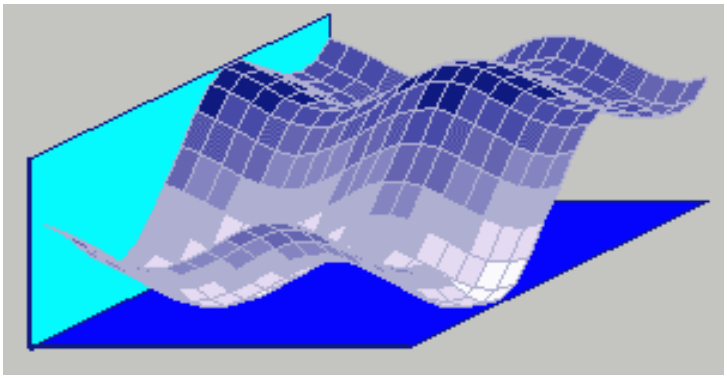


Рис.1.10. Тип серій «Surface» (TeeChart Pro)

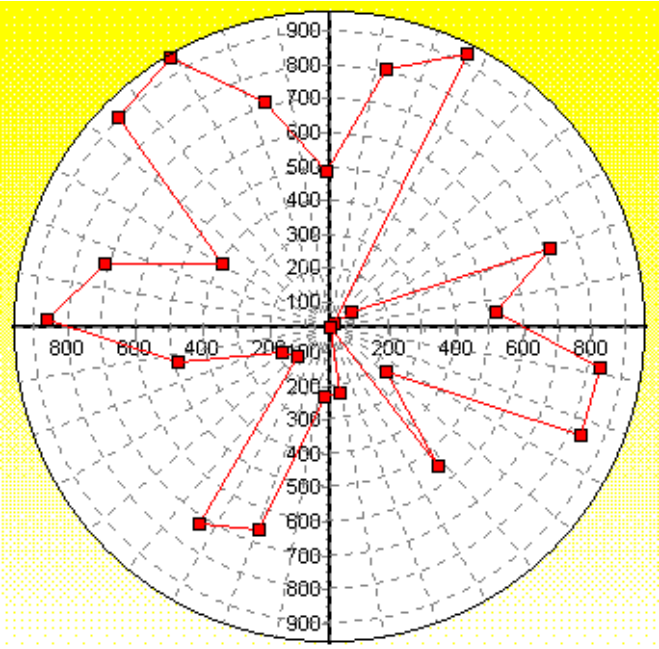


Рис.1.11. Тип серій «Polar» (TeeChart Pro)

До складу професійної версії TeeChart входить докладне керівництво зі створення власних типів серій.

Графічні редактори середовища DelphiTLogGraph

Цей компонент зручно використовувати у разі відображення швидко змінних даних, оскільки він володіє великою швидкістю перемальовування зображення. Дозволяє малювати двовимірні графіки і діаграми з використанням

декількох різних стилів. Зображення декількох серій на одному графіку не підтримує.

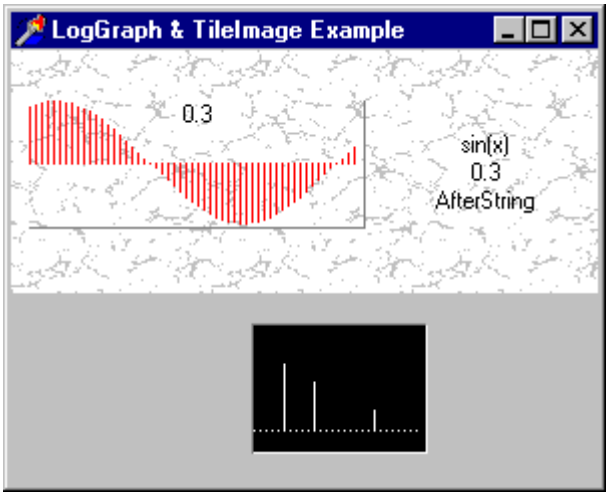


Рис.1.12. Приклад використання компоненту TlogGraph *Ttrend*

TTrend - графічний компонент, призначений для зображення динамічних графіків за даними, що постійно додаються. При додаванні даних відбувається горизонтальний скролінг зображення. Компонент дозволяє використовувати різні стилі для зображення даних: двовимірні і тривимірні стовпчасті діаграми, лінійні діаграми, окремі крапки, лінійні діаграми із заповненням. В загальному випадку відображається одна серія, але можливе відображення двох серій (наприклад, сигналу і відгуку на нього).

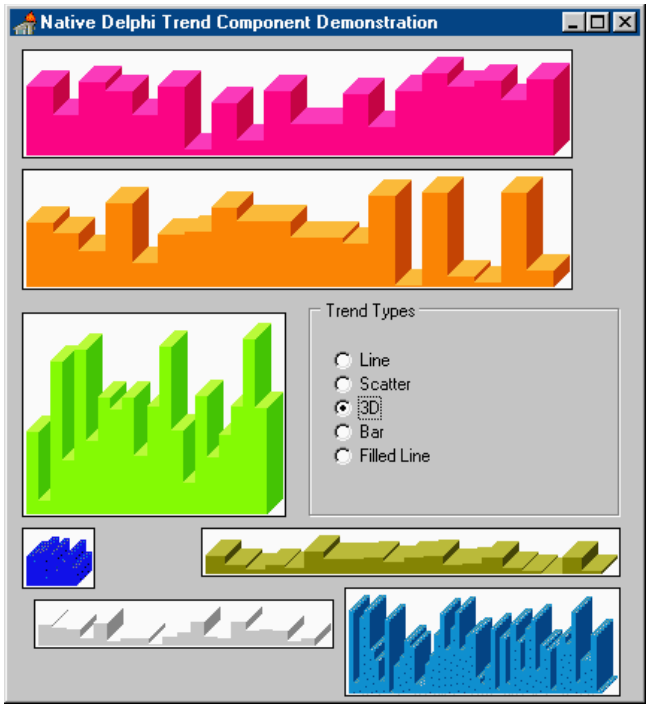


Рис.1.13. Приклад використання компоненту TgraphWin

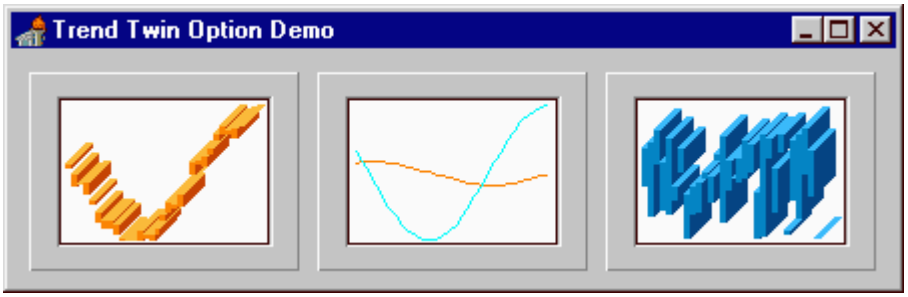


Рис.1.14. Приклад використання компоненту TTrend для зображення двох серій

TRChar

Компонент TRChart дозволяє відображати двовимірні графіки. Він дозволяє автоматично вибирати масштаб, використовувати логарифмічну шкалу, створювати динамічні графіки для відображення швидко змінних значень, використовувати як значення змінні типа TDateTime, змінювати масштаб зображення за допомогою миші, додавати маркери, надписи пояснень, прямокутники, еліпси, які автоматично масштабуються і переміщаються при зміні масштабу графіка. Володіє різноманітними можливостями, пов'язаними з друком графіків (Фаронов, 2012).

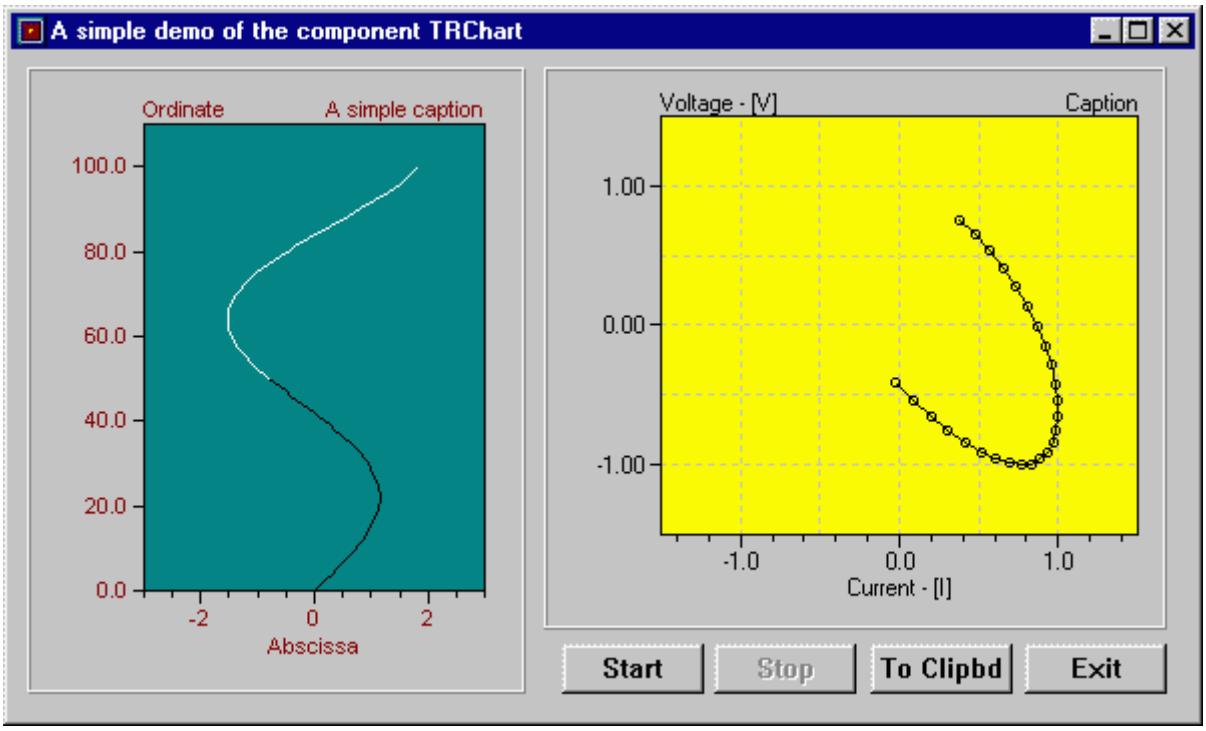


Рис.1.15. Приклад використовування компоненту TRChart

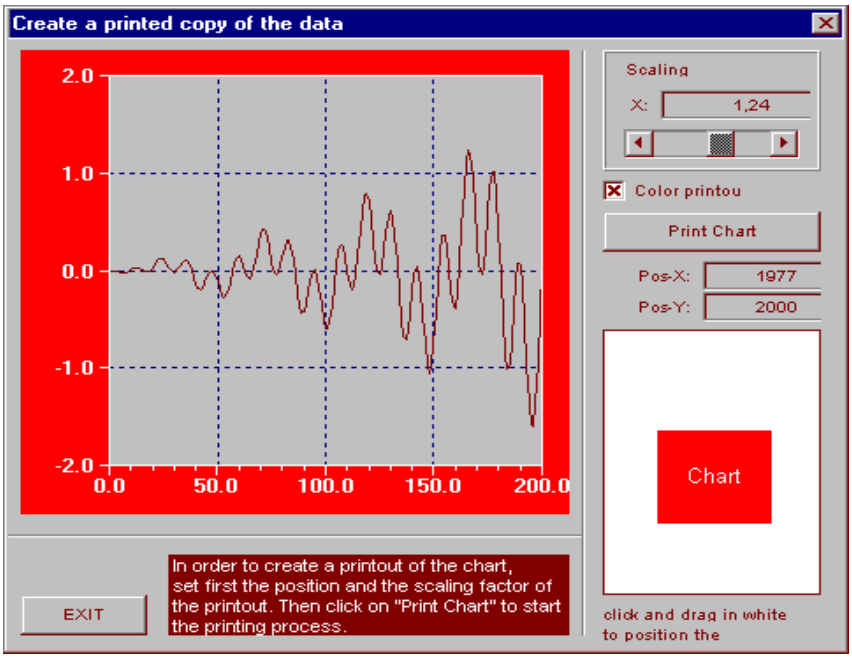


Рис.1.16. Приклад установки параметрів друку при використовуванні компоненту TRChart

PIE

Компонент TPie призначений для відображення кругових діаграм. Дозволяє вибрати кольори секторів діаграми.

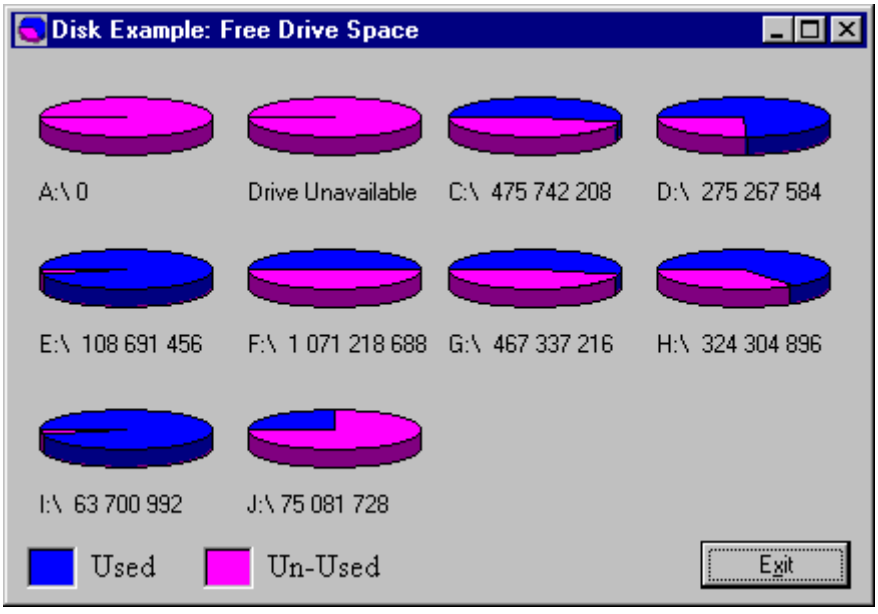


Рис.1.17. Приклад використання компоненту TPie

РОЗДІЛ ІІ. КОМП'ЮТЕРНЕ МОДЕЛЮВАННАЯ ФІЗИЧНИХ ЯВИЩ В ОПТИЦІ

2.1. Комп'ютерне моделювання оптичних параметрів законів відбивання та заломлення світла.

Мета: вивчити фізичний зміст законів геометричної оптики з використанням комп'ютерного моделювання, зокрема показника заломлення, явищ відбивання та заломлення світла.

Теоретичні відомості

Оптика є наукою, яка вивчає і аналізує поширення та поглинання світла, електромагнітні випромінювання, довжина хвилі яких лежить в межах від 1 до 10^5нм: видиме світло (λ=0,4мкм÷0,76мкм), інфрачервоне (λ>0,76мкм) і ультрафіолетове світло λ<0,4мкм.

Погляди на природу світла

У 17 ст. були дві теорії про природу світла: корпускулярна (Ньютон) і хвильова (Гюйгенс). У 1900 р. Планк висунув гіпотезу, що світло випромінюється квантами - окремими порціями, а Ейнштейн припустив, що світло не тільки випромінюється, але й поширюється в просторі і поглинається речовиною також окремими порціями – квантами. У теперішній час ця теорія найбільш поширена.

Основні закони геометричної оптики: 1) *закон прямолінійного поширення світла* – в однорідному прозорому середовищі світло поширюється прямолінійно; 2) *закон незалежності поширення світлових променів* – світлові промені, які розповсюджуються в просторі, при перетині не впливають один на одного; 3) *закон зворотності світлових променів* – якщо промінь світла поширюється з точки 1 в точку 2, то на зворотньому шляху з точки 2 в точку 1 він розповсюджується за тим самим шляхом; 4) *закон відбивання світла* – промінь падаючий, промінь відбитий і перпендикуляр, в точці падіння, лежать в одній площині; при цьому кут падіння дорівнює куту відбивання $\alpha=\beta$

(рис.2.1). Кут між падаючим променем (A*O*) і перпендикуляром (*CO*) у точку падіння називають кутом падіння (α). Кут між відбитим променем (*OB*) і перпендикуляром (C*O*) у точку падіння називають кутом відбивання (β).

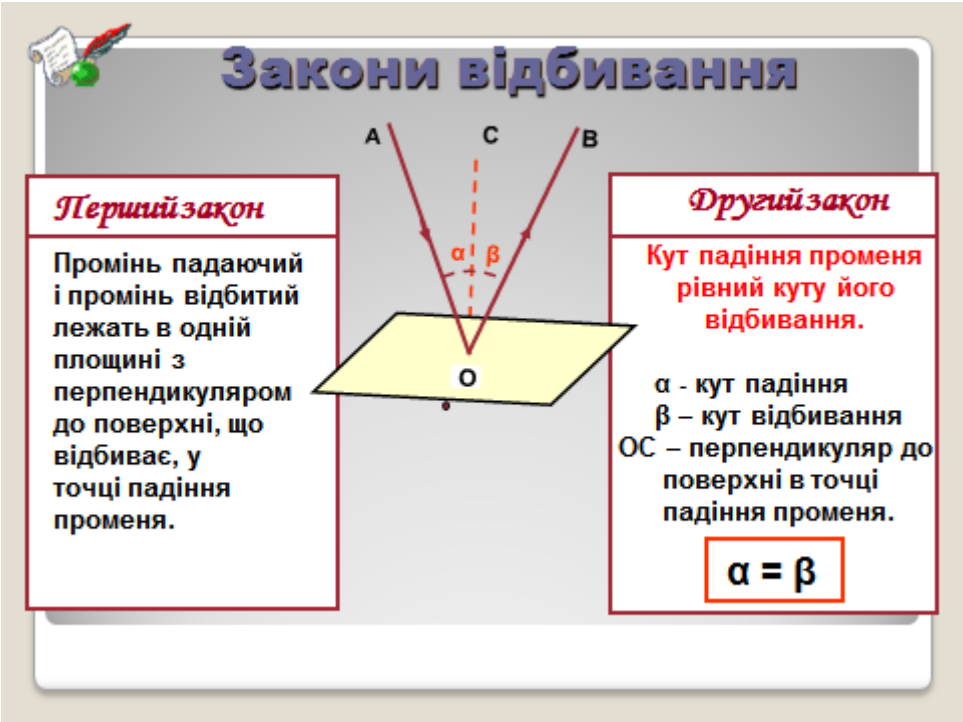


Рис.2.1.Закони відбивання світла.

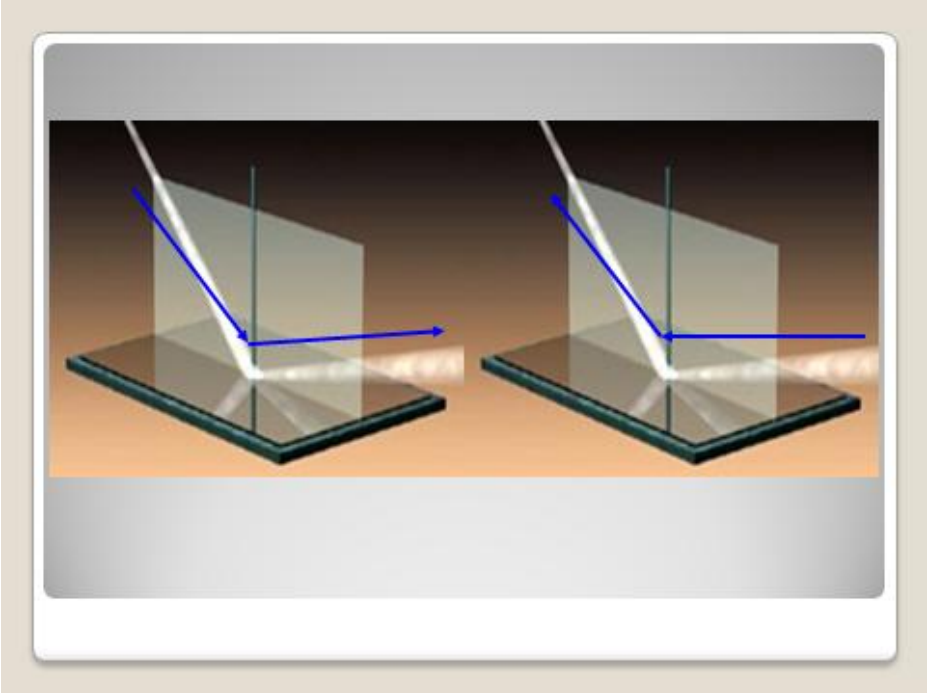


Рис.2.2. Відбивання світлового променя від непрозорої поверхні.

Види відбивання світла

Від різних поверхонь світлові промені відбиваються по-різному. Існують два види відбивання променів: дзеркальне відбивання й дифузне відбивання. Коли промінь світла потрапляє на ідеально плоску поверхню, що відбиває, розміри нерівностей якої не більші довжини світла, має місце дзеркальне відбивання. При цьому промені, що входять у світловий пучок, відбиваючись, залишаються взаємно паралельними. Прикладами поверхонь, що близькі за властивостями до дзеркальних є поверхня гладкого скла, краплі ртуті, відполірована металева поверхня, звичайне дзеркало (Сивухин, 1980).

При попаданні світла на нерівну, шорстку поверхню, що відбиває (розміри нерівностей перевищують довжину світлової хвилі) має місце дифузне відбивання. У цьому випадку відбиті промені направлені хаотично відносно один одного. Явище дифузного відбивання дає нам можливість розрізняти предмети, які самі не здатні випромінювати світло. Предмет є абсолютно невидимим, якщо розсіювання світлових променів дорівнює нулю. Але, навіть ідеально відполіровані дзеркала розсіюють частину світла

а)

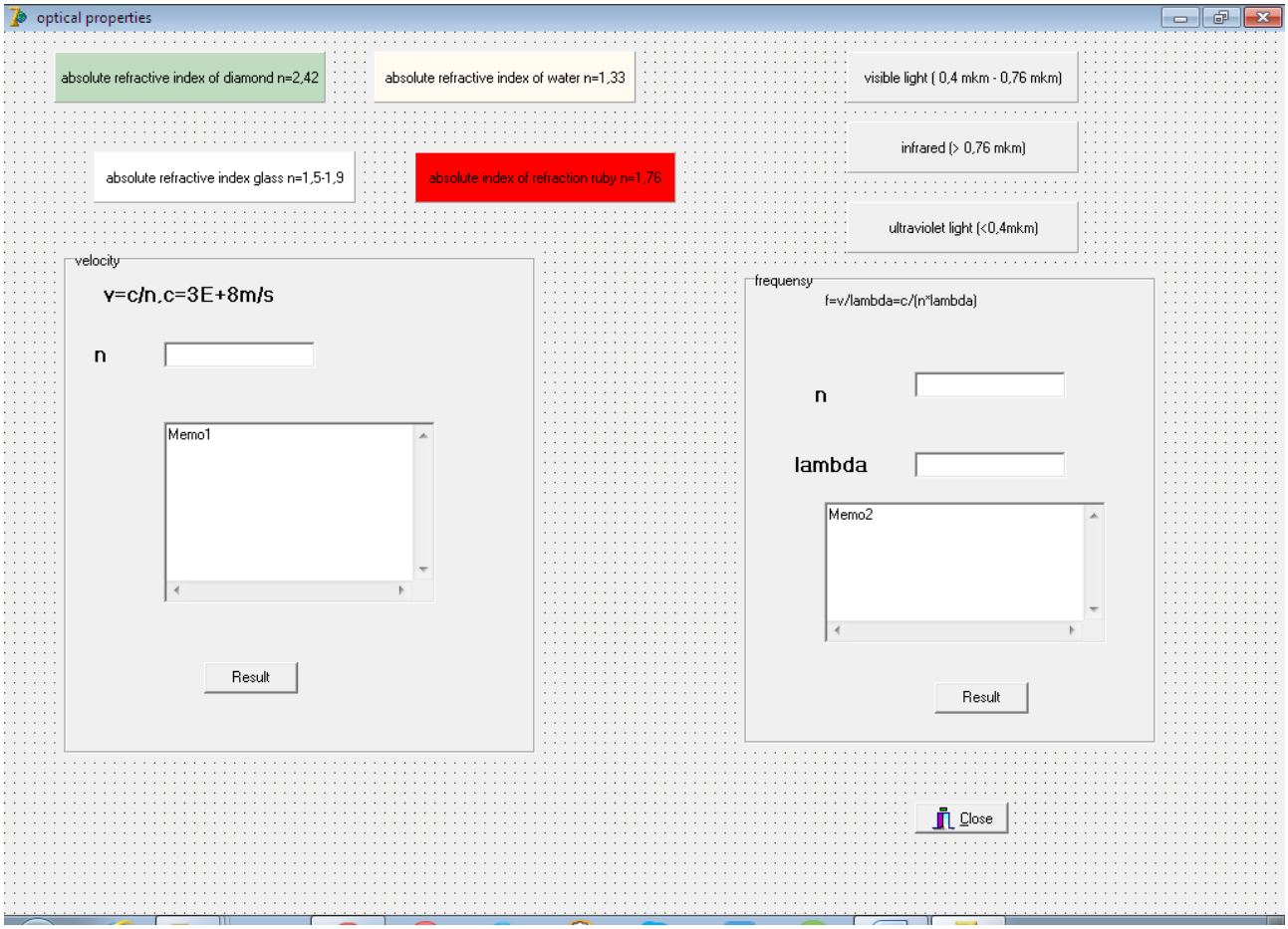
б)

в)

Рис.2.3. Ілюстрації дзеркального та дифузного відбивання у різноманітних середовищах.

Виконання роботи

1. Скласти програму для розрахунку оптичних параметрів законів відбивання і заломлення світла з використанням середовища програмування Delphi за таким алгоритмом:
   - помістити на форму компоненти GroupBox, Label, Panel і записати в них основні формули розрахунку оптичних параметрів законів відбивання і заломлення світла;
   - задати обробку подій для кнопок Button, Bitbtn (див.Лістинг1);
   - введення вихідних значень виконати в Edit;
   - виведення результату виконати в Memo.
2. Зберегти проект і створити файл *.exe.



Закріплення.

1. Проаналізувати закони заломлення світла.

2. Чому заломлюється світло на межі поділу двох прозорих середовищ?
3. Фізичний зміст показника заломлення?
4. Яка різниця між відносним і абсолютним показником заломлення?
5. Коли відносний показник заломлення більший за одиницю, а коли менший за одиницю?

2.2. Комп'ютерне моделювання закону Снеліуса в середовищі Delphi з використанням компоненти PaintBox.

Мета: провести комп'ютерне моделювання закону Снеліуса з використанням графічного інструментарію в Delphi.

Теоретичні відомості

Закони заломлення формулюються так (Сивухин, 1980):

1) промінь, що падає, заломлений промінь і перпендикуляр у точці падіння променя лежать в одній площині.

2) відношення синуса кута падіння і синуса кута заломлення є постійною величиною для розділюваних двох середовищ:

$$\frac{sin\alpha}{sin\beta} = \frac{v_1}{v_2} = n_{21}$$

де v_1 - швидкість світла в першому середовищі; v_2 - швидкість світла в другому середовищі; n_{21} - відносний показник заломлення світла у другому середовищі відносно першого.

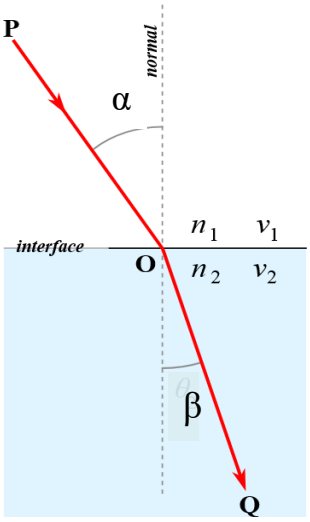


Рис.2.4. Заломлення світла

Якщо першим середовищем є вакуум, то показник заломлення називають абсолютним. Абсолютні показники заломлення визначено для всіх середовищ і занесено до таблиць.

Оскільки $n_1 = \frac{c}{v_1}$, $n_2 = \frac{c}{v_2}$, де c - швидкість світла у вакуумі, то

$n_{21} = \frac{v_2}{v_1} = \frac{n_2}{n_1}$.

Фізичний зміст показника заломлення зформулювали тільки після отримання законів заломлення за принципом Гюйгенса. Відносний показник заломлення показує у скільки разів швидкість світла в одному середовищі є більшою за швидкість світла в другому середовищі.

Середовище є з оптично більш густішим з більшим абсолютним показником заломлення, а з меншим - оптично менш густим. Якщо світло рухається з оптично менш густого середовища у більш густе то промінь буде «наближатись» до перпендикуляра. Якщо ж світло переходить із більш оптично густого середовища в менш густе, то промінь світла буде відхилятись від перпендикуляра (рис.2.5).

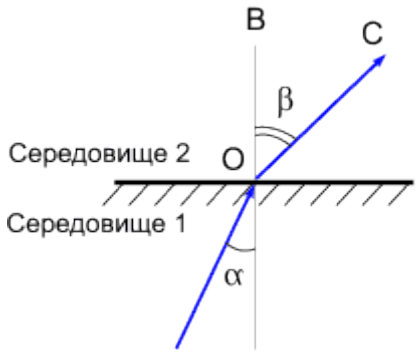


Рис.2.5. Заломлення світла на межі двох середовищ.

Як і для відбивання світла закон заломлення підтверджується при оберненому напрямі ходу світлових променів. Коли світло переходить з одного середовища в інше, змінюється його швидкість і довжина хвилі.

Частота світла у вакуумі:

$$\nu = \frac{c}{\lambda}.$$

Частота світла для середовища:

$$\nu = \frac{\upsilon}{\lambda} ,$$

$$\frac{c}{\lambda} = \frac{\upsilon}{\lambda} \Rightarrow \lambda = \frac{\lambda_0 \upsilon}{c} = \frac{\lambda_0}{n} .$$

Якщо світло іде із оптично більш густого в оптично менш густе середовище, тоді:

$$\frac{\sin\alpha}{\sin 90°} = \frac{n_2}{n_1} ,$$

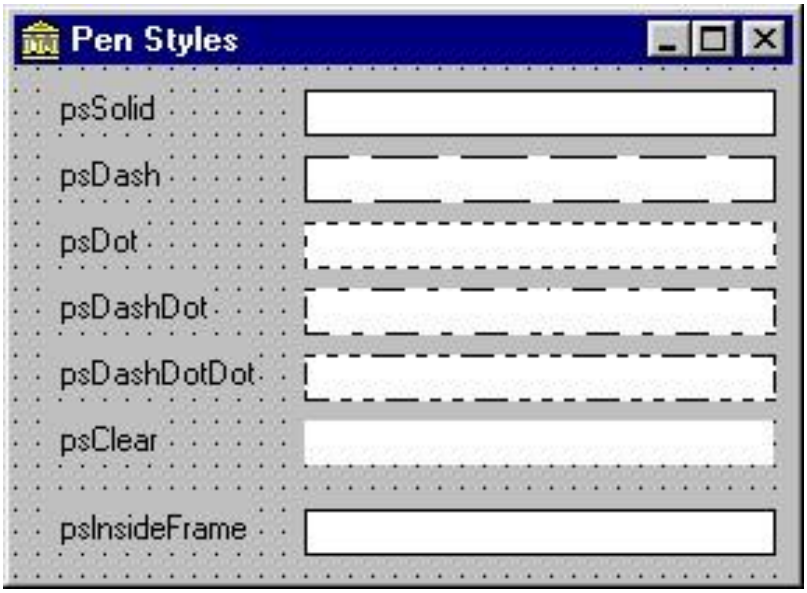
$$\sin\alpha = \frac{n_2}{n_1} ,$$

n_2=1 (повітря) => $sina = \frac{1}{n}$.

Явище повного внутрішнього відбивання має місце, коли світло взагалі не виходить у інше середовище.

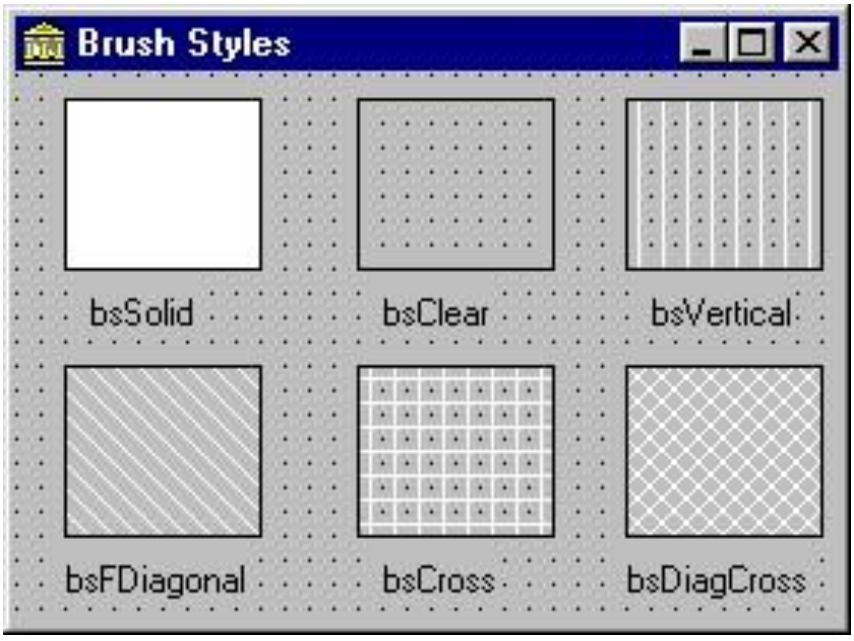
Графічні класи в середовищі Delphi

1) Клас *Pen* (перо)- призначений для створення ліній і границь геометричних фігур. Можна задавати такі властивості як колір, стиль, товщину. Приклад: Колір Pen.Color:=ClBlack; Тип: Pen.Style:=psSolid; Товщина: Pen.Width:=2.



2) Клас *Font*(шрифт) - призначений для виводу тестової інформації. Приклад: Колір: Font.Color:=ClRed; Розмір: Font.Size:=10; Ім'я шрифта: Font.Name:=«Arial»; Тип: Font.Style:=fsbold;

3) Клас *Brush* (пензлик)- призначений для зафарбовування і штрихування фігур. Приклад: Колір: Brush.Color:=ClGreen; Тип: Brush.Style:=bsSolid.



Клас *Canvas* (полотно) містить властивості класів Pen, Font, Brush і методи, які виконують побудову різного роду геометричних об'єктів (дуги, лінії, еліпси, прямокутники, кола і тп.)

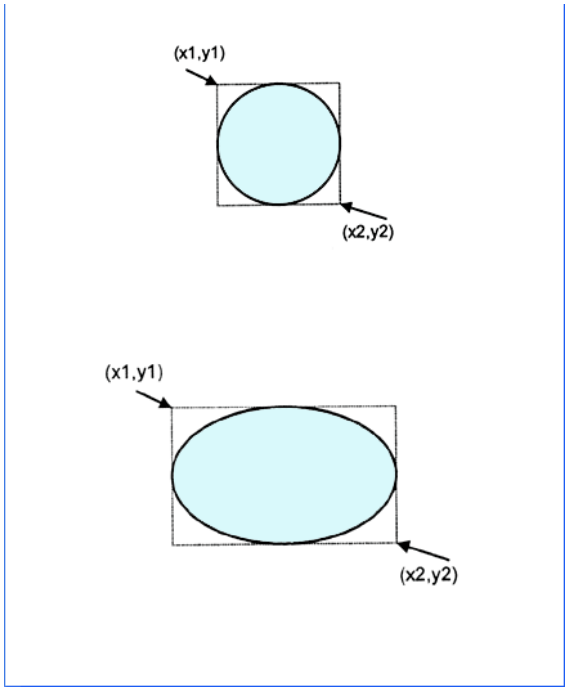
Викреслювання прямої лінії здійснює метод *LineTo*: *Компонент.Canvas.LineTo(x,y)*. Початкову точку лінії можна задати, перемістивши олівець в потрібну точку графічної поверхні. Зробити це можна за допомогою методу *MoveTo*, вказавши як параметри координати нового положення олівця. Вид лінії (колір, товщина і стиль) визначається значеннями властивостей об'єкту Pen графічної поверхні, на якій викреслюється лінія.

Ламана лінія.

Метод *Polyline* викреслює ламану лінію. Як параметр метод одержує масив типу TPoint. Кожний елемент масиву є записом, у полях котрого містяться координати точки перегину ламаної (Мар'ян, Юркович, 2007).

Коло і еліпс.

Метод Ellipse викреслює еліпс або коло, залежно від значень параметрів. Інструкція виклику методу в загальному вигляді виглядає таким чином: *Об'єкт.Canvas.Ellipse(x1,y1,x2,y2)*, де: об'єкт - ім'я об'єкту (компоненту) на поверхні якого виконується викреслювання; x1, y1, x2, y2 - координати прямокутника, усередині якого викреслюється еліпс або, якщо прямокутник є квадратом, коло.



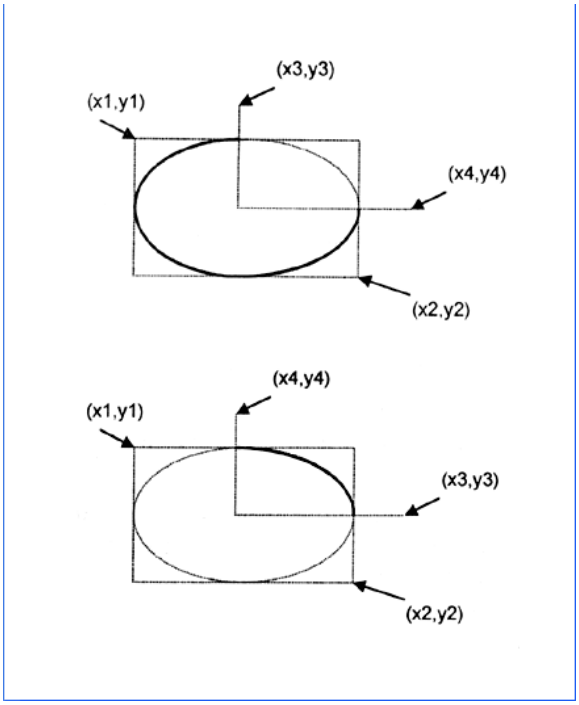
Значення параметрів методу Ellipse визначають вигляд геометричної фігури. Колір, товщина і стиль лінії еліпса визначаються значеннями властивості Pen, а колір і стиль заливки області усередині еліпса – значеннями властивості Brush поверхні (Canvas).

Дуга.

Викреслювання дуги виконує метод *Arc*, інструкція виклику якого в загальному вигляді виглядає таким чином (Мар'ян, Юркович, 2007) :

Об'єкт.Canvas.Arc(x1,y1,x2,y2,x3,y3,x4,y4), де: x1, y1, x2, y2 - параметри, що визначають еліпс (коло), частиною якого є викреслювана дуга; x3, y3 - параметри, що визначають початкову точку дуги; x4, y4 - параметри, що визначають кінцеву точку дуги.

Дуга викреслюється проти годинникової стрілки від початкової точки до кінцевої (див.нижче). Колір, товщина і стиль лінії, якій викреслюється дуга, визначаються значеннями властивості Pen поверхні (canvas), на яку виконується вивід.

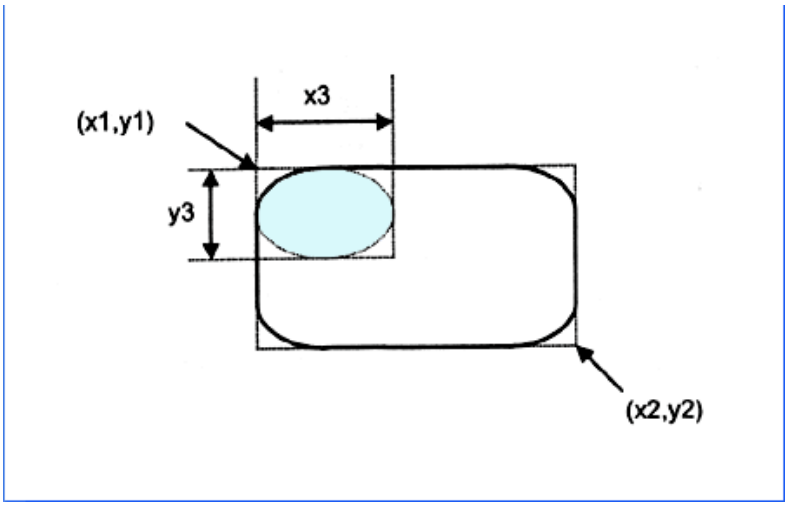


Значення параметрів методу Arc визначають дугу як частину еліпса (кола).

Прямокутник.

Прямокутник викреслюється методом *Rectangle*, інструкція виклику якого в загальному вигляді виглядає таким чином: *Об'єкт.Canvas.Rectangle(x1, y1,x2, y2)*, де: об'єкт - ім'я об'єкту (компоненту), на поверхні якого виконується викреслювання; x1, y1 і x2, y2 - координати лівого верхнього і правого нижнього кутів прямокутника.

Метод *RoundRec* теж викреслює прямокутник, але з округлими кутами. Інструкція виклику методу RoundRec виглядає так: *Об'єкт.Canvas.RoundRec(x1,y1,x2, y2, x3, y3)*, де: x1, y1, x2, y2 - параметри, що визначають положення кутів прямокутника, в який вписується прямокутник з округляючими кутами; x3 і y3 - розмір еліпса, одна четверть якого використовується для викреслювання округлого кута.



Метод RoundRec викреслює прямокутник з округлими кутами. Вид лінії контура (колір, ширина і стиль) визначається значеннями властивості Pen, а колір і стиль заливки області усередині прямокутника - значеннями властивості Brush поверхні (canvas), на якій прямокутник викреслюється.

Є ще два методи, які викреслюють прямокутник, використовуючи як інструмент тільки кисть (Brush).

- Метод *FillRect* викреслює зафарбований прямокутник;

-Метод *FrameRect* - тільки контур.

У кожного з цих методів лише один параметр - структура типу TRect. Поля структури TRect містять координати прямокутної області, вони можуть бути заповнені за допомогою функції Rect.

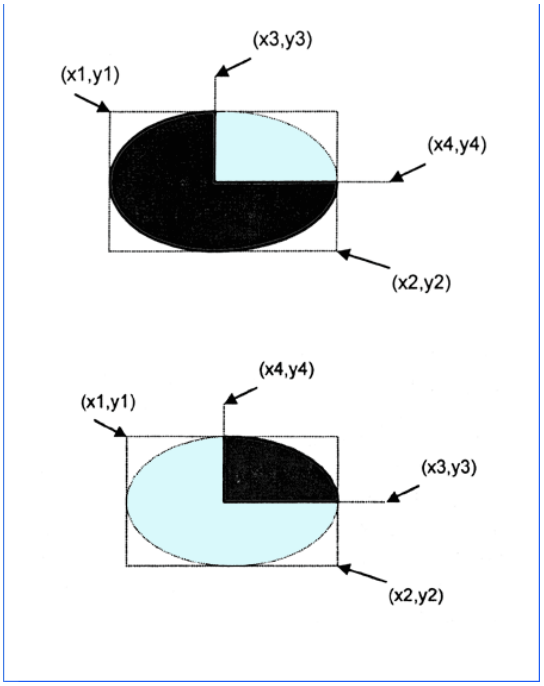
Багатокутник.

Метод *Polygon* викреслює багатокутник. Як параметр метод одержує масив типу TPoint. Кожний елемент масиву є записом, поля (x,y) якого містять координати однієї вершини багатокутника. Метод Polygon викреслює багатокутник, послідовно сполучаючи прямими лініями точки, координати яких знаходяться в масиві: першу з другою, другу з третьою і т.д. Потім з'єднуються остання і перша точки. Остання в масиві повинна бути такою ж як і перша, щоб багатокутник з'єднався.

Сектор.

Метод *Pie* викреслює сектор еліпса або круга. Інструкція виклику методу в загальному вигляді виглядає таким чином (Юркович, 2010):

Об'єкт. Canvas.Pie(x1,y1,x2,y2,x3,y3,x4,y4), де: x1, y1, x2, y2 - параметри, що визначають еліпс (коло), частиною якого є сектор; x3, y3, x4, y4 - параметри, визначальні координати кінцевих точок прямих, є межами сектора. Початкові точки прямих співпадають з центром еліпса (кола). Сектор вирізується проти годинникової стрілки від прямої, заданою точкою з координатами (x3, y3), до прямої, заданою точкою з координатами (x4, y4). Значення параметрів методу Pie визначають сектор як частину еліпса (кола).



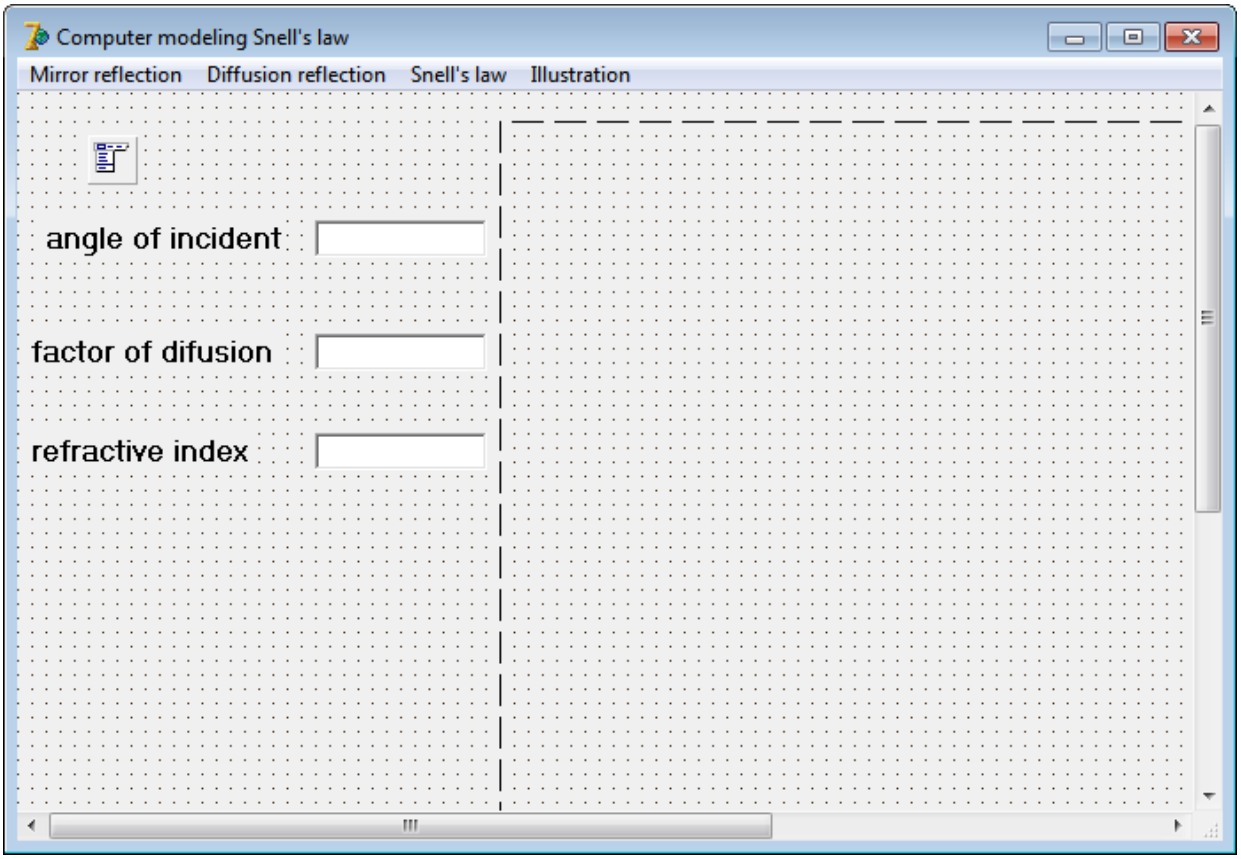
Точка.

Поверхні, на яку програма може здійснювати виведення графіки, відповідає об'єкт Canvas. Властивість *Pixels*, що є двовимірним масивом типу TColor, містить інформацію про колір кожної точки графічної поверхні. Використовуючи властивість Pixels, можна задати потрібний колір для будь-якої точки графічної поверхні, тобто «намалювати» точку. Наприклад, інструкція Form1.Canvas.Pixels[10,10]:=clRed забарвлює точку поверхні форми в червоний колір (Мар'ян, Юркович, 2008) .

Виконання роботи

Завдання 1. Користуючись вказівками скласти програму для моделювання закону Снеліуса.

1. Виконати візуалізацію форми для моделі закону Снеліуса: нанести на форму компоненти MainMenu, PaintBox, Edit, Label за допомогою Object Inspector задати потрібні зміни (форма, назва і т.д.).



MainMenu – компонента для створення головного меню програми. Основна подія – OnClick виникає при клацанні на неї. Саме в обробці цієї події записуються оператори, які повинні виконуватися при клацанні користувача на компоненту. Розміщена вона у вкладці Standart.

Edit - однорядкове текстове поле, яке слугує для введення/виведення тексту користувачем. Основною властивістю компонента Edit, яка передає введену інформацію є властивість Edit1.Text типу String. Розміщена дана компонента у вкладці Standart.

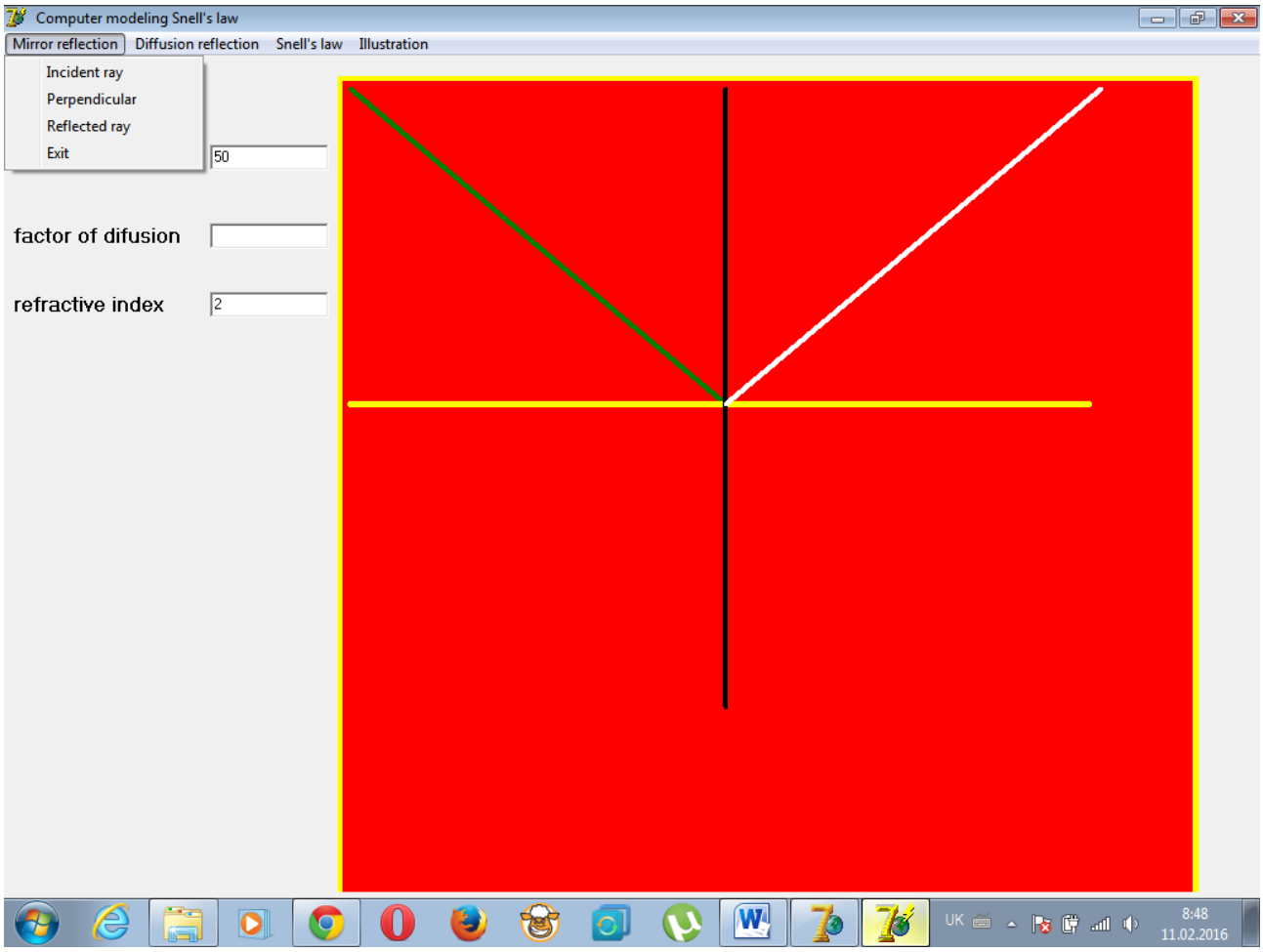
Label – мітка, спеціально призначена для відображення тексту на формі. Розміщена вона у вкладці Standart.

PaintBox (вкладка System) класу TPaintBox використовується в тих випадках, коли необхідно мати прямокутну область для виконання графічних операцій, використовуючи властивість Canvas. При цьому можна використовувати всі можливості Canvas - перо, пензлик, шрифт і методи побудови геометричних

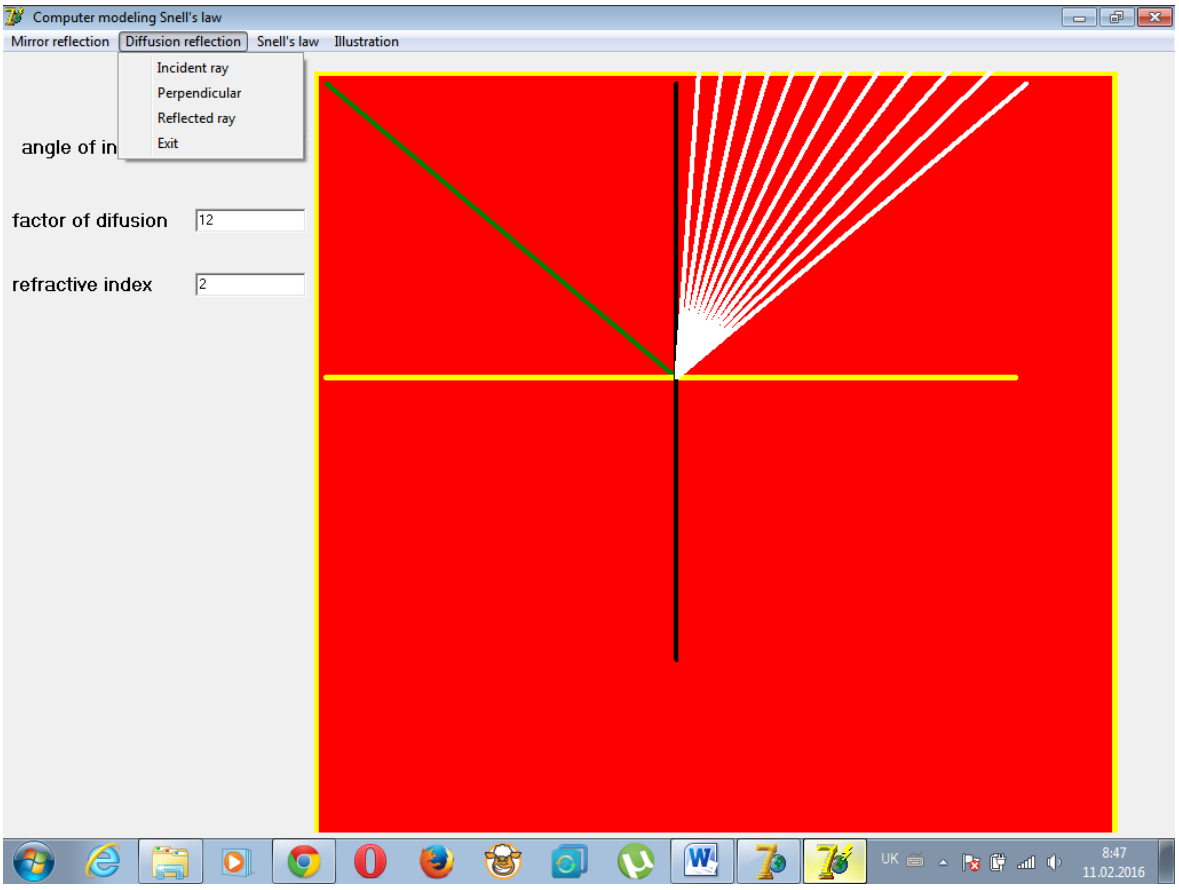
фігур. Але TPaintBox не дозволяє завантажувати готові зображення (Фаронов, 2012).

2. Клікнути по компоненті MainMenu**,** ввести код програми (див.Лістинг2).

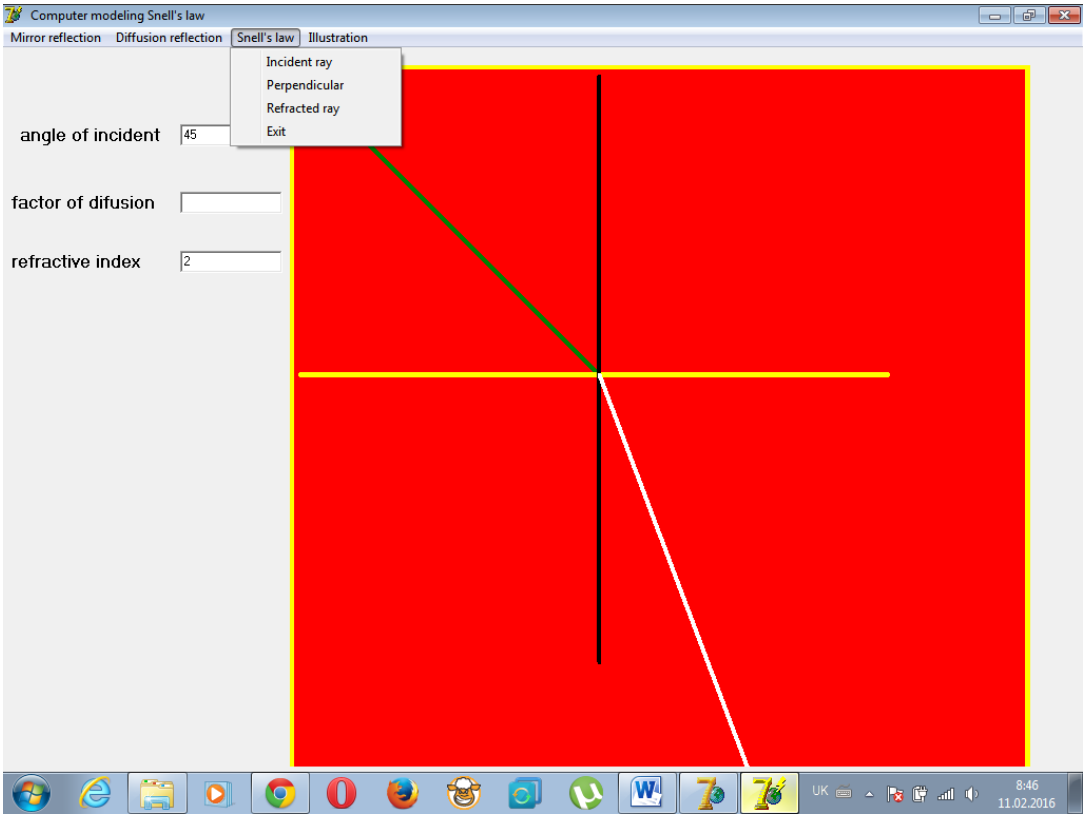
Візуальний інтерфейс комп'ютерного моделювання дзеркального відбивання



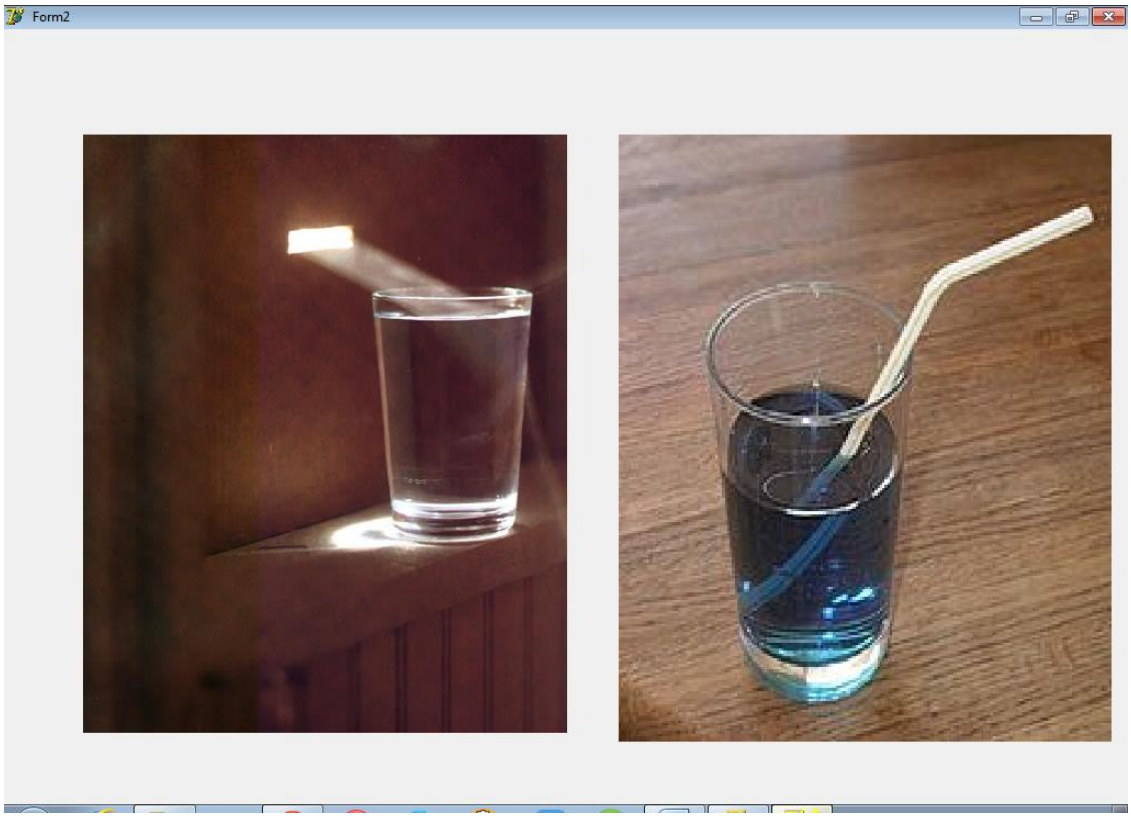
Візуальний інтерфейс комп'ютерного моделювання дифузного відбивання



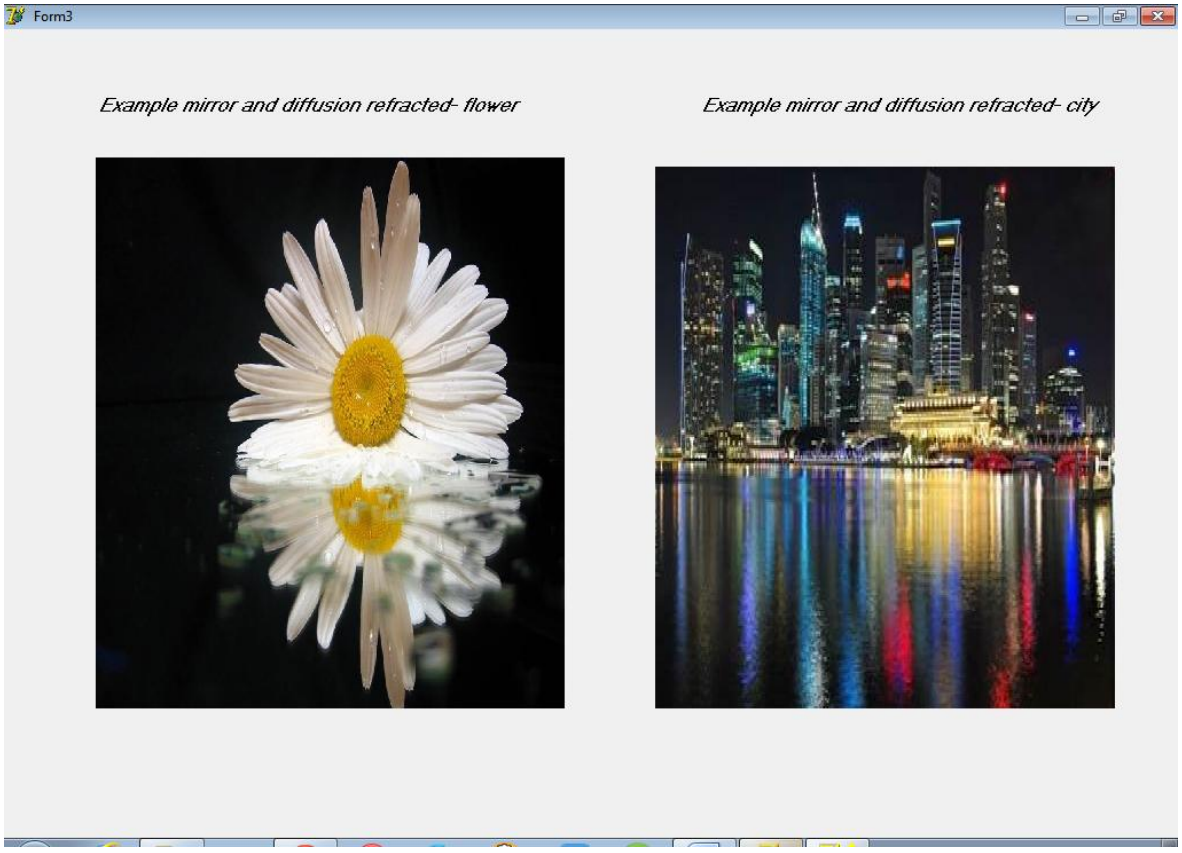
Візуальний інтерфейс комп'ютерного моделювання закону Снеліуса



Ілюстрації закону Снеліуса



Ілюстрації дифузного відбивання



Завдання 2.Створити ілюстрації для дзеркального відбивання із підключенням нової форми.

Закріплення.

1. Графічно зобразити хід променів у разі заломлення світла на межі поділу двох прозорих середовищ.
2. Яка формула виражає зв'язок відносного показника заломлення двох середовищ, що межують, з їх абсолютними показниками заломлення.

2.3. Візуальне програмування повного внутрішнього відбивання з використанням мови програмування Object Pascal.

Мета: Змоделювати закон повного внутрішнього відбивання.

Теоретичні відомості

Якщо збільшувати кут падіння α, то при граничному значенні кута α_0 (кут повного внутрішнього відбивання), кут $\beta = 90°$. При такому куті падіння і більших кутах заломлений промінь вже не може проникнути в інше середовище, а відбивається - відбувається явище повного внутрішнього відбивання світла (рис.2.6).

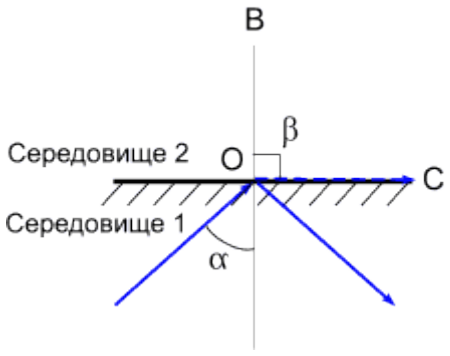


Рис.2.6. Явище повного внутрішнього відбивання світла.

Коли $\alpha = \alpha_0$, то $\beta = 90^o$ і $\frac{sin\alpha}{sin\beta} = \frac{1}{n}$, $\sin 90^o = 1$ то $\alpha_0 = arcsin\frac{1}{n}$ - граничний кут повного внутрішнього відбивання.

Деякі задачі геометричної оптики зручно розв'язувати з використанням принципа Ферма: світло поширюється за тим шляхом, на подолання якого йому потрібний мінімальний час.

Якщо світло поширюється в середовищі з показником заломлення n то час, за який світло долає деяку відстань S у середовищі з показником заломлення n, визначається співвідношенням $t = \frac{S}{V} = \frac{Sn}{c} = \frac{L}{c}$, де S – геометрична довжина шляху, $L = Sn$ - оптична довжина шляху.

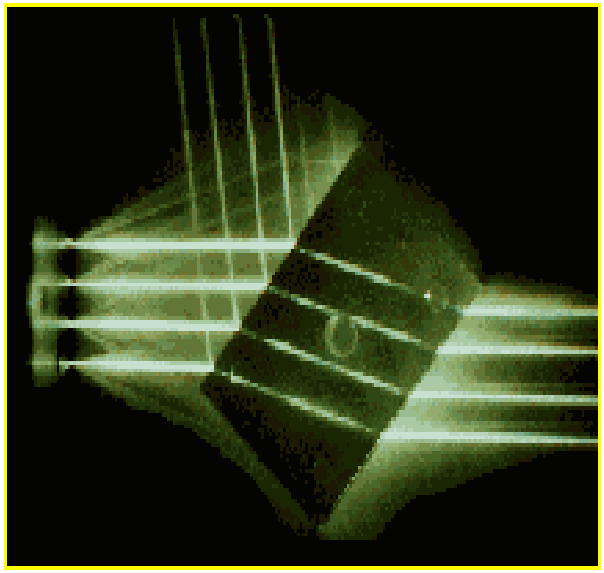


Рис.2.7. Хід світлових променів через прозорі середовища різної оптичної густини.

Явище повного внутрішнього відбивання застосовують у волоконній оптиці. Це явище зумовило революцію в передачі інформації на відстанях.

Оптоволокно

Оптичне волокно є технічним виробом, складається з оптичного світловоду, захисних покриттів та маркуючої кольорової оболонки. Оптичний світловід є фізичним середовищем транспортування оптичного сигналу, складається із серцевини та оболонки, що мають різні величини показників заломлення. Явище повного внутрішнього відбивання дає можливість транспортувати оптичні сигнали, які генеруються обладнанням, до якого підключене оптичне волокно.

Процес розповсюдження світла по оптичному волоконному світловоду (ВС) описує хвильова електромагнітна теорія. Згідно неї, у волоконному світловоді можуть розповсюджуватись лише хвилі, що формують резонансну хвилю у поперечному перерізі волоконного світловоду. Ці види хвиль утворюють моди хвилеводу. Одно - чи багатомодовий режим роботи волоконного світловоду визначається величиною нормованої частоти V (Таненбаум, Уэзеролл, 2012).

Розрізняють світловоди з градієнтним показником заломлення та сходинковим профілем показника заломлення. Оптоволокна використовуються в оптоволоконному зв'язку, з допомогою якого можна передавати цифрову інформацію на великі відстані і з високою швидкістю передачі даних. Крім того, їх використовують для створенні датчиків, сенсорів.

Вплив коефіцієнта заломлення

Коефіцієнт заломлення визначається співвідношенням швидкостей світла у вакуумі та матеріалі, до якого належить цей коефіцієнт. У залежності від властивостей матеріалу, промінь світла розповсюджується у вакуумі зі швидкістю 300 000 км/сек, а у діелектрику — повільніше. Отже, показник заломлення для матерілу оптоволокна більший одиниці (коефіцієнт заломлення для оболонки - 1.46, для серцевини - 1.48). Чим більший показник заломлення речовини, тим швидкість променю в ній нижча. Оптичний комунікаційний сигнал буде проходити приблизно 200 000 км/сек (10000 кілометрів сигнал пройде за 5 мсек).

Повне внутрішнє відбивання

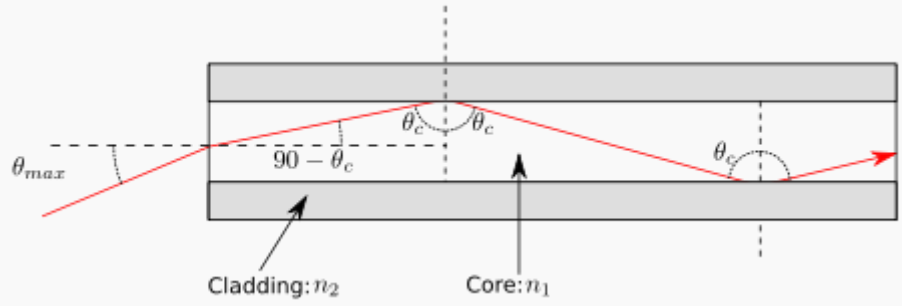


Рис.2.8. Траєкторія світлового променя в оптичному волокні.

Коли промінь, що проходить у оптично густому матеріалі, натикається на перешкоду під кутом падіння, більшим ніж критичний кут для даного матеріалу, світло повністю віддзеркалиться. Цим ефектом утримують світлове випромінювання у серцевині оптичного волокна.

Дифузне відбивання світла

Розсіювання світла залежить від довжини світлової хвилі. Причиною загасання є розсіювання світла, створеного внутрішніми поверхнями та границями розділу речовин. Нещодавно було продемонстровано, що коли розмір центру розсіювання менший величини довжини хвилі світла, що розсіюється, то показники величини дифузного відбивання більше не мають практично значення. Цей факт став початком виробництва прозорих керамічних матеріалів.

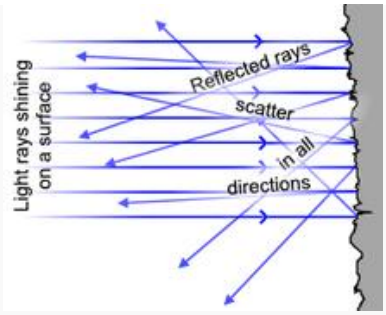
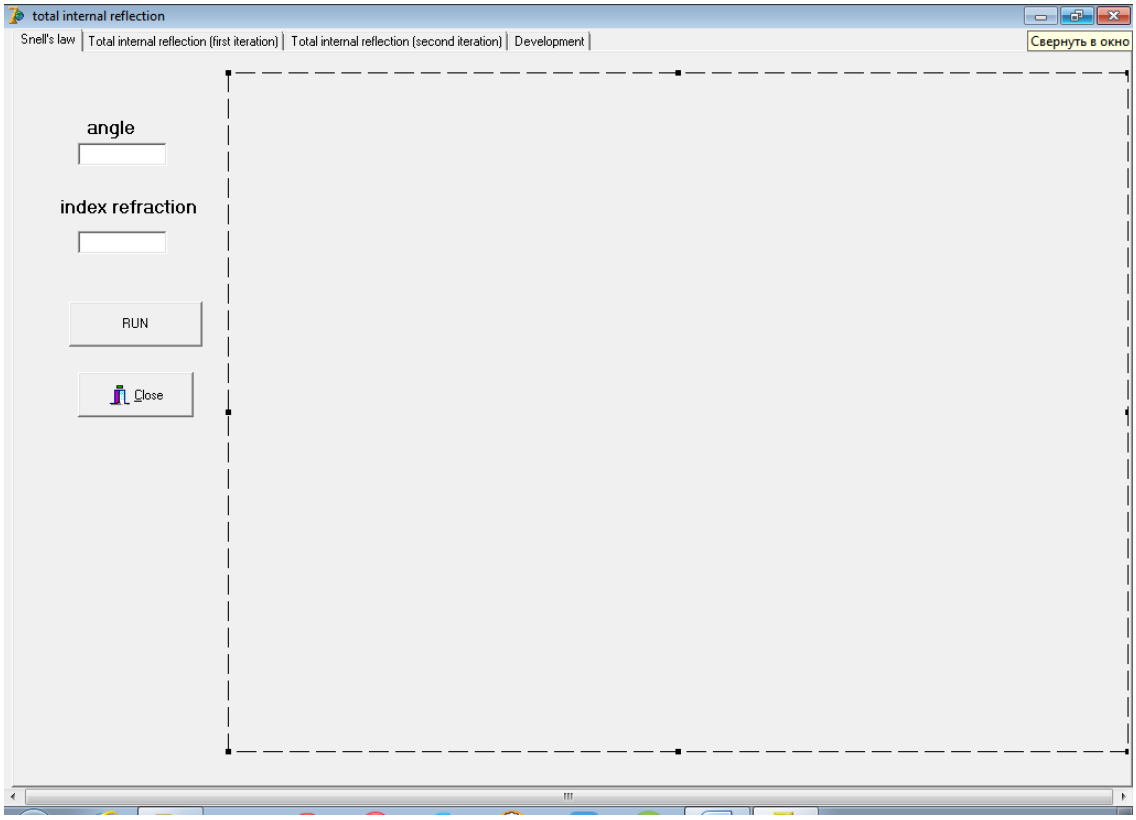


Рис.2.9. Дифузне відбивання світла.

Виконання роботи

Завдання 1. Користуючись вказівками скласти програму для моделювання повного внутрішнього відбивання з використанням мови програмування Object Pascal.

1. Виконати візуалізацію форми для інтерфейсу повного внутрішнього відбивання з допомогою компонент PageControl, PaintBox, Edit, Label, Button, BitBtn, Image та з допомогою Object Inspector внести потрібні зміни.



PageControl – компонента для створення сторінок, розміщена у вкладці Win32, властивість SheetTab, визначає нові сторінки та їх назви.

Button – компонента для виконання програми. Основна подія – OnClick виникає при клацанні на неї. Саме в обробці цієї події записуються оператори, які повинні виконуватися при клацанні користувача на компоненту. Розміщена вона у вкладці Standart.

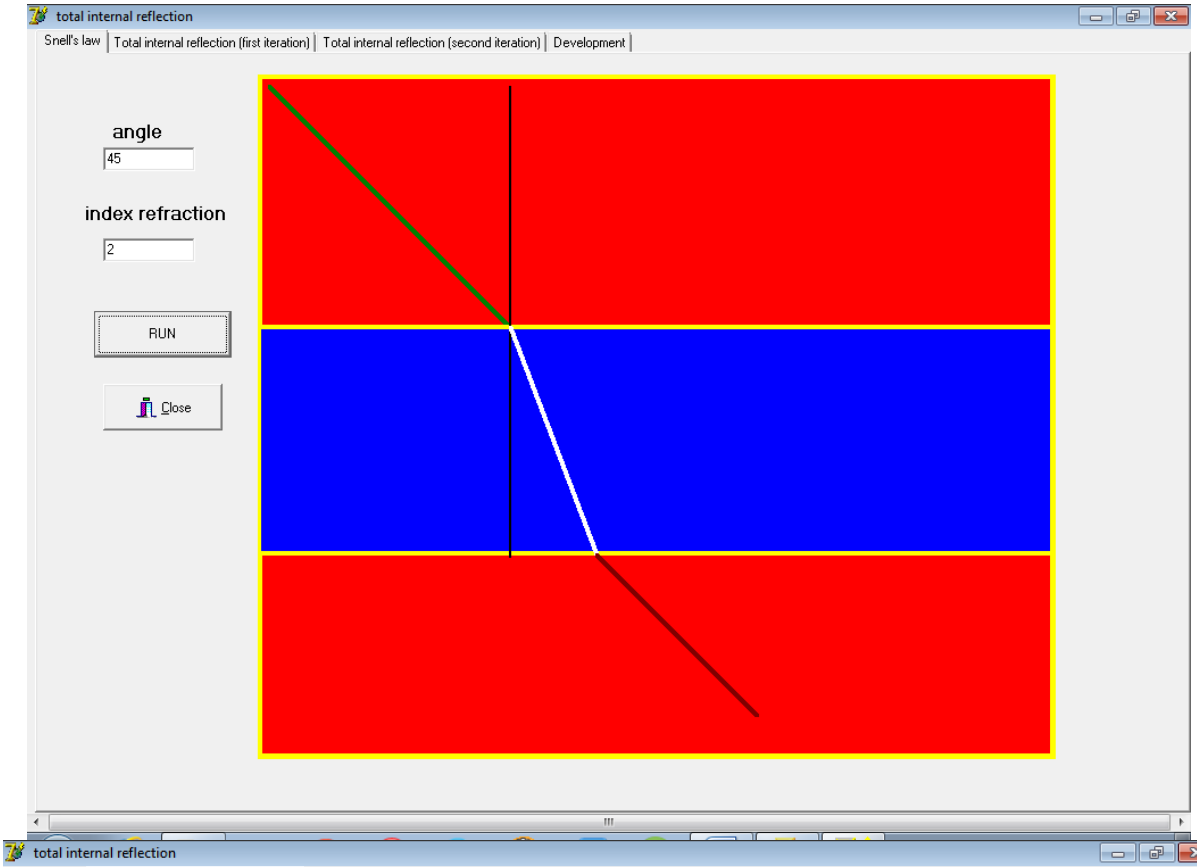
Edit - однорядкове текстове поле, яке слугує для введення/виведення тексту користувачем. Основною властивістю компоненти Edit, яка передає введену інформацію є властивість Edit1.Text типу String. Розміщена у вкладці Standart.

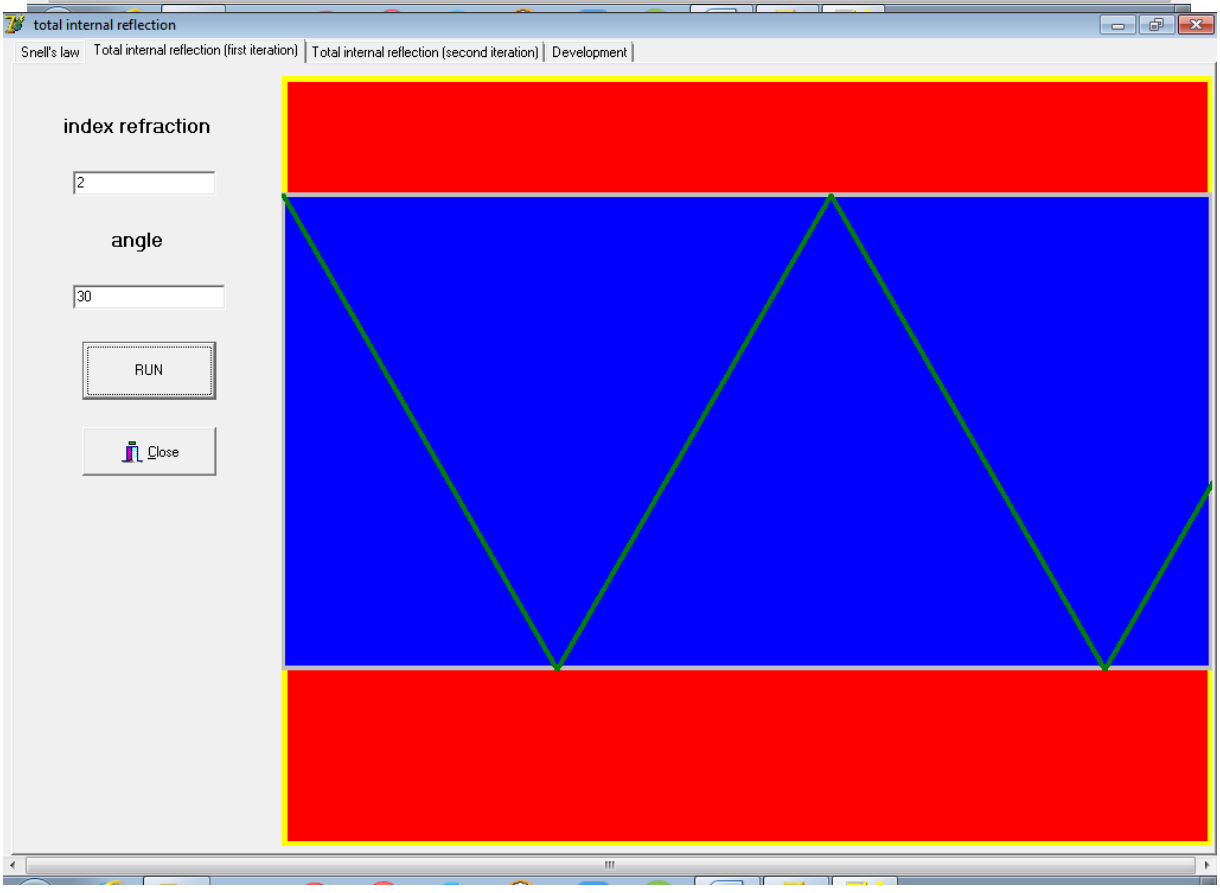
Label – мітка, спеціально призначена для відображення тексту на формі, розміщена вона у вкладці Standart.

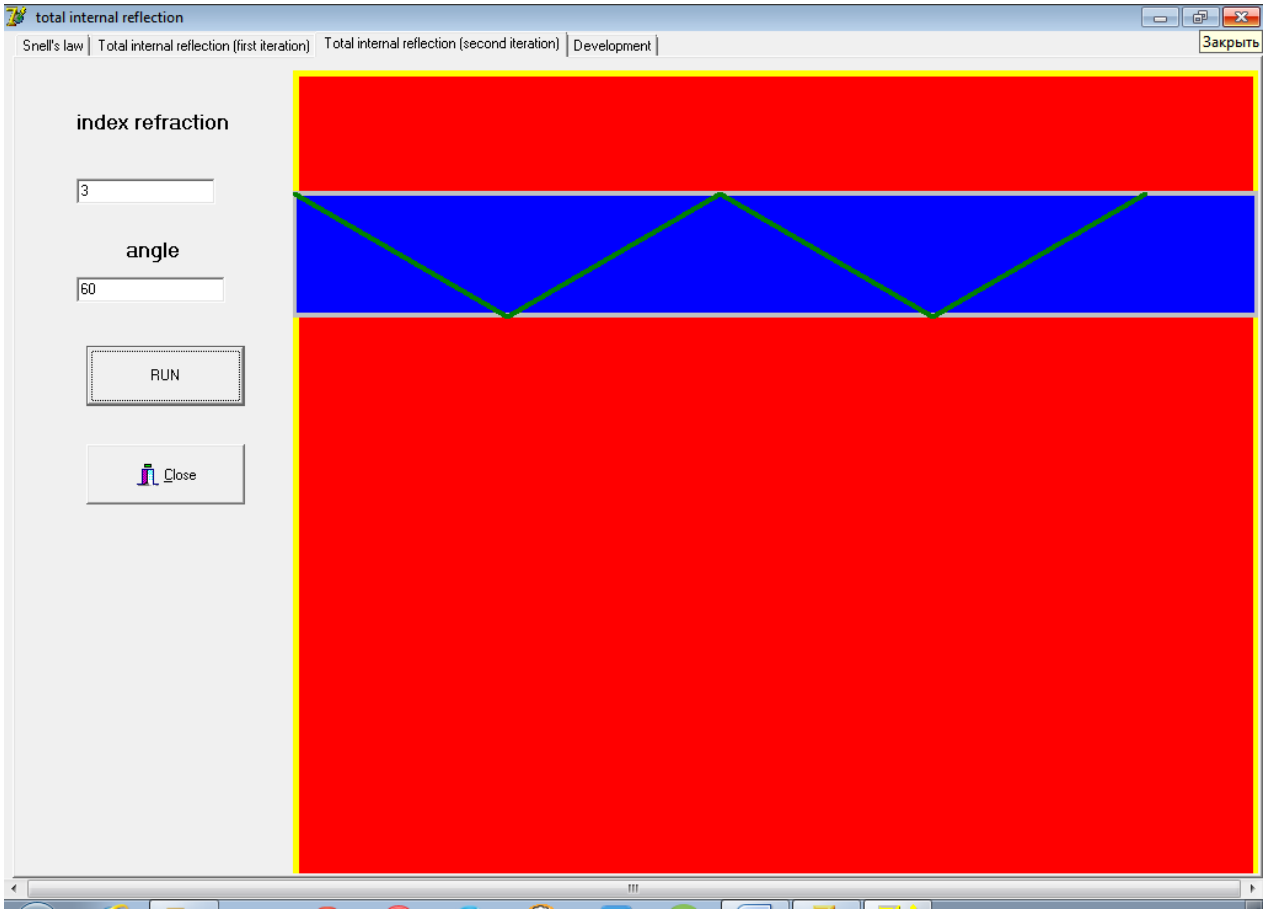
PaintBox (вкладка System) класу TPaintBox використовується в тих випадках (Фаронов, 2012), коли необхідно мати прямокутну область для виконання графічних операцій, використовуючи властивість Canvas. При цьому можна використовувати всі можливості Canvas - перо, пензлик, шрифт і методи побудови геометричних фігур. Але TPaintBox не дозволяє завантажувати готові зображення.

2. Клікнути на компоненті Button, ввести код програми (див.Лістинг3).

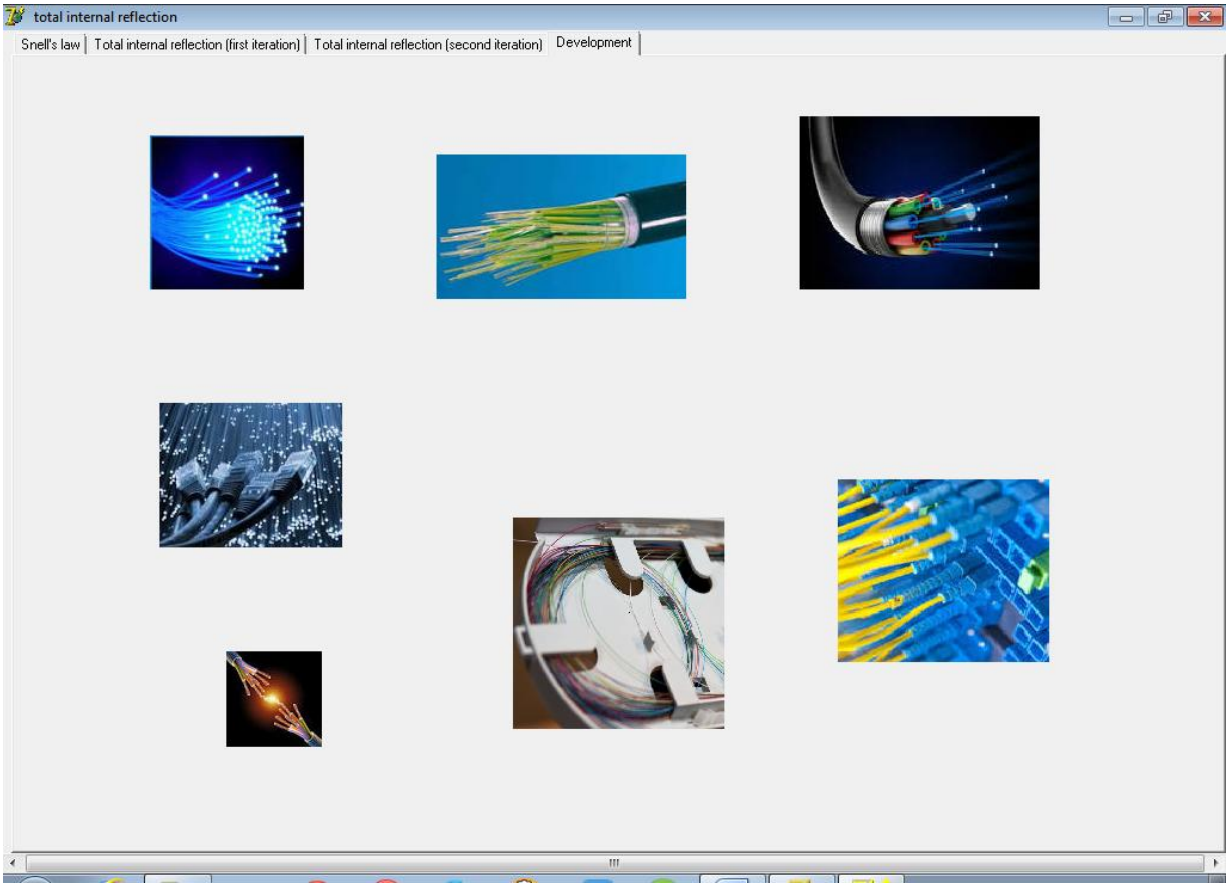
Візуальний інтерфейс комп'ютерного моделювання повного внутрішнього відбивання







Застосування явища повного внутрішнього відбивання



Завдання 2. Із використанням компоненти Image показати приклад застосування явища повного внутрішнього відбивання.

Закріплення.

1. Що таке граничний кут повного внутрішнього відбивання світла ?
2. Практичне застосування повного внутрішнього відбиття світла ?
3. Чому біле світло, проходячи крізь призму, розкладається в кольоровий спектр ?
4. Яке світло буде поширюватися в речовині призми (склі) з більшою швидкістю - червоне чи фіолетове ?
5. Що відбудеться під час з'єднання двох світлових променів спектру?

2.4. Визначення спектральної залежності пропускання з використанням методів апроксимації та їх моделювання в середовищі Delphi.

Мета: Поглибити уявлення про взаємодію світла з речовиною, ознайомитися з елементарними уявленнями і законами поглинання світла. Використовуючи методи апроксимації та їх моделювання в середовищі Delphi, дослідити спектральну залежність пропускання світла.

Теоретичні відомості

Поглинанням (абсорбцією) світла називається явище зменшення енергії світлової хвилі при її поширенні в речовині внаслідок перетворення енергії хвилі в інші види енергії. Поглинання світла в речовині описується законом Бугера:

$$I = I_0 e^{-\alpha x} \qquad (1)$$

де I_0 і I - інтенсивність плоскої монохроматичної світлової хвилі на вході і виході шару поглинаючої речовини товщиною x, α - коефіцієнт поглинання, залежить від довжини хвилі світла, хімічної природи і стану речовини. Коефіцієнт поглинання не залежить від інтенсивності падаючого світла (закон Ламберта). При товщині шару $x=1/\alpha$ інтенсивність світла I у порівнянні

з I_0 зменшується в $e=2,72$ разів. Розмірність коефіцієнта поглинання $см^{-1}$ (Гулд, Тобочник, 1990).

Коефіцієнт поглинання залежить від довжини хвилі світла і для різних речовин різний. Наприклад, одноатомні гази і пари металів володіють близьким до нуля коефіцієнтом поглинання і лише для дуже вузьких спектральних областей спостерігаються різкі максимуми поглинання (так званий лінійчатий спектр поглинання). Коефіцієнт поглинання для металів має великі значення (10^3 - 10^5 $см^{-1}$) і тому метали є непрозорими для світла. Коефіцієнт поглинання діелектриків зазвичай невеликий (10^{-3} - 10^{-5} $см^{-1}$). Скло, прозорі полімерні плівки, рідини та розчини мають селективне (виборче) поглинання світла в певних інтервалах довжин хвиль, коли α різко зростає, і спостерігаються порівняно широкі смуги поглинання (Сивухин, 1980).

Для характеристики поглинання зразка використовується або коефіцієнт пропускання T, який зазвичай вимірюється у відсотках або оптична щільність зразка D. Коефіцієнт пропускання (прозорості) показує яка частина світлового потоку, падає на досліджуваний об′єкт і проходить через нього не поглинаючись:

$$T = \frac{\Phi}{\Phi_0} = \frac{I - I_T}{I_0 - I_T} \cdot 100\%. \qquad (2)$$

Оптична густина D речовини характеризує ступінь поглинання нею монохроматичного випромінювання і описується співвідношенням:

$$D = \lg\frac{1}{T} = \lg\frac{I - I_T}{I_0 - I_T} \cdot 100\% \qquad (3)$$

Відповідно до закону Бугера коефіцієнт пропускання експоненціально зменшується залежно від товщини зразка (шару речовини):

$$T = \exp(-\alpha x), \qquad (4)$$

а оптична щільність залежить від товщини зразка лінійно:

$$D = \alpha \cdot x \cdot lge = 0.43 \cdot \alpha \cdot x \qquad (5),$$

тобто оптична щільність речовини прямо пропорційна товщині шару. Тому для оцінки поглинаючої здатності зразка застосування оптичної щільності більш зручно, ніж застосування коефіцієнта поглинання.

Використання методів апроксимації.

Для експериментально отриманої залежності пропускання від довжини хвилі у видимій області спектру проведено двопараметричне наближення з використанням методів апроксимації (Гулд, Тобочник, 1990).

Розглянуто двопараметричну залежність

$$y = ge^{mz} \ , \qquad (6)$$

де m, g - варіаційні параметри, які визначаються в процесі комп'ютерного моделювання. Прологарифмувавши, одержано:

$$\ln y = \ln g + mz \, . \qquad (7)$$

Звідси:

$$\ln y - \ln g - mz = 0 \, . \qquad (8)$$

Оскільки аналітична функція не дає точний опис експериментальної залежності, то права частина (8) не буде рівна нулю. Тоді, просумувавши квадрати величин, що входять в (8), одержано деяке середньоквадратичне відхилення

$$\delta_n^2 = \sum_{i=1}^{n} (\ln y_i - \ln g - mz_i)^2 \ , \qquad (9)$$

де n – число експериментальних значень.

Взявши похідні по варіаційних параметрах m, g :

$$\begin{aligned} \frac{\partial \delta_n^2}{\partial m} &= 2 \cdot \sum_{i=1}^{n} (\ln y_i - \ln g - mz_i) \cdot (-z_i). \\ \frac{\partial \delta_n^2}{\partial g} &= 2 \cdot \sum_{i=1}^{n} (\ln y_i - \ln g - mz_i) \cdot \frac{1}{g}. \end{aligned} \qquad (10)$$

Необхідною умовою мінімуму величини $\delta_n^2(m, g)$ по m, g є рівність нулю частинних похідних (10).Таким чином,

$$\begin{aligned} &\ln g\sum_{i=1}^{n} z_i + m\sum_{i=1}^{n}(z_i)^2 = \sum_{i=1}^{n} z_i \ln y_i. \\ &n\ln g + m\sum_{i=1}^{n} z_i = \sum_{i=1}^{n} \ln y_i. \end{aligned} \qquad (11)$$

З (11) знайдено:

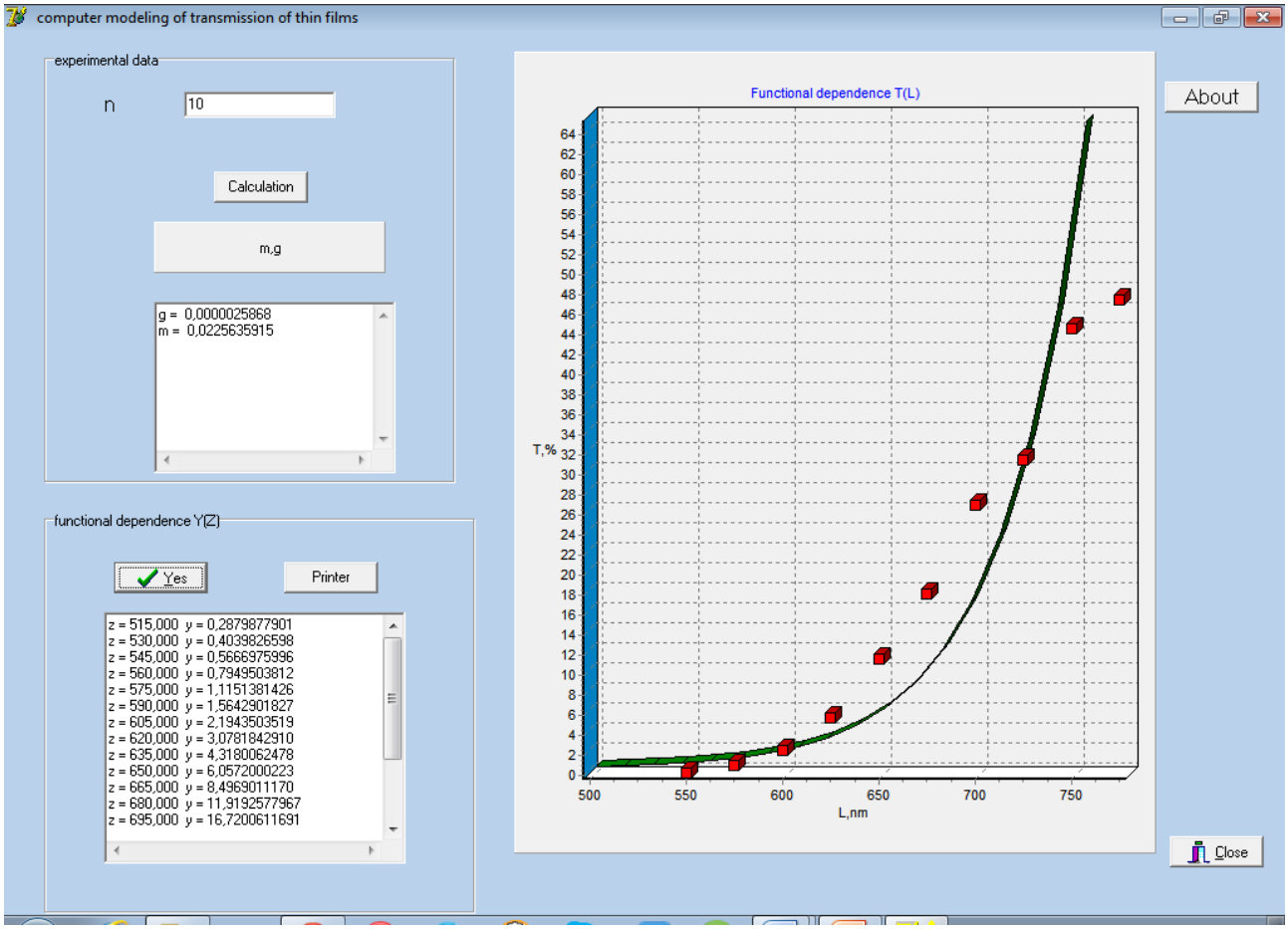
$$\begin{aligned} m &= \frac{n\sum_{i=1}^{n} z_i \cdot \ln y_i - \sum_{i=1}^{n} \ln y_i \cdot \sum_{i=1}^{n} z_i}{n\sum_{i=1}^{n} z_i^2 - \sum_{i=1}^{n} z_i \cdot \sum_{i=1}^{n} z_i}, \\ \ln g &= \frac{\sum_{i=1}^{n} z_i \ln y_i - m\sum_{i-1}^{n}(z_i)^2}{\sum_{i=1}^{n} z_i}. \end{aligned} \qquad (12)$$

Отже, з (12) можна розрахувати значення параметрів m, g , які відповідають експериментальним залежностям (див.інтерфейс програми)

Виконання роботи

Завдання 1. Користуючись вказівками визначити спектральну залежність пропускання світла з використанням методів апроксимації та їх моделювання в середовищі Delphi.

1. Виконати візуалізацію форми для визначення пропускання світла тонкої плівки з використанням компонент Button, GroupBox, Edit1, Label, Panel, Memo, Chart та з допомогою Object Inspector надати їм необхідних властивостей (в т.ч. типи серій). Для компоненти Button задати обробку події (програма розрахунку параметрів приведена в Лістинг4).



Завдання 2. Показати залежність зміни графічних залежностей від числа експериментальних точок.

Закріплення

1.Що таке поглинання?

2.Що ви знаєте про взаємодію світла з речовиною?

3. Які є методи апроксимації і на чому вони базуються?

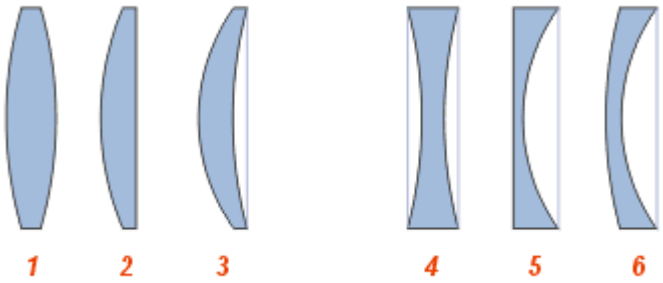
2.5. Компонента Image середовища програмування Delphi і побудова зображень за допомогою лінз.

Мета: Створити форму «Побудова зображень за допомогою лінз» з оптичними даними та різними графічними об'єктами, утвореними з допомогою лінз, використовуючи різні властивості компонент.

Теоретичні відомості

При виготовленні лінз для видимого діапазону світла використовують оптичне або органічне скло, в ультрафіолетовому діапазоні спектру кварц, в інфрачервоному діапазоні - скло, германій , кремній, сапфір тощо. Природною оптичною лінзою є наш кришталик ока. Лінза є тонкою, якщо її товщина мала у порівнянні з радіусами сферичних поверхонь, що її обмежують. Сферичні тонкі лінзи є опуклі і ввігнуті (Сивухин, 1980).

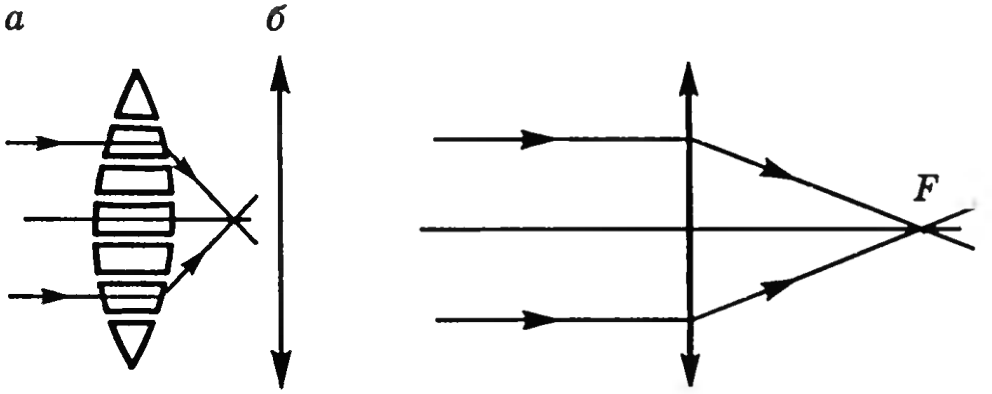
Опуклі лінзи збирають заломлене світло, у них середина товста, а краї тонші (кожну лінза умовно розділяється на три частини, де краї - призми, що заломлюють промені до основи, а середина - плоскопаралельна пластинка). Їх називають збиральними. Ввігнуті лінзи (середина тонка, а краї товстіші) розсіюють світло після заломлення, вони є розсіювальними. У залежності від розміщення центрів сферичних поверхонь та їхнього радіусу розрізняють такі типи лінз:



1. двоопукла;
2. плоско-опукла;
3. збиральний меніск;
4. двоввігнута;
5. плоско-ввігнута;
6. розсіювальний меніск.

Лінзи поділяються на збиральні й розсіювальні, у залежності від того, сходяться чи розходяться паралельні пучки променів після проходження лінзи.

Якщо лінзи перетворюють паралельний пучок світлових променів в пучок, що сходиться у фокусі F, вониє збиральними.



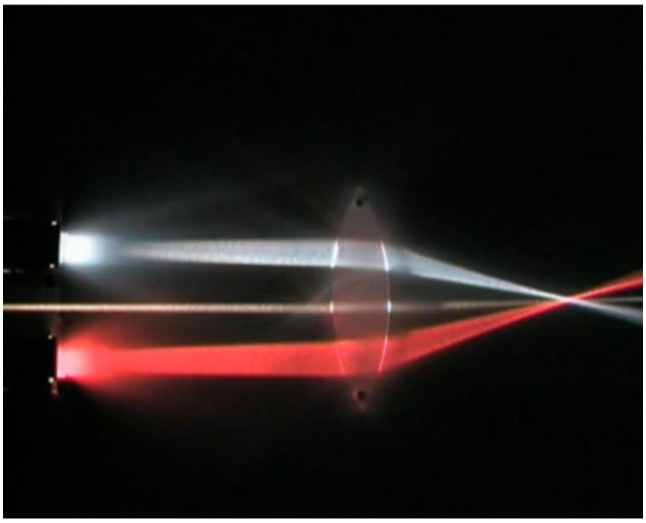
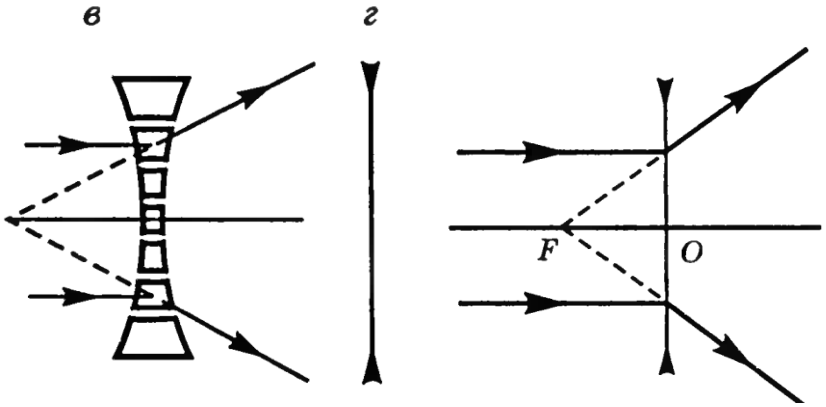


Рис.2.10. Збиральні лінзи.

Якщо лінзи перетворюють паралельний пучок світлових променів в пучок, що розходиться, вони є розсіювальними .



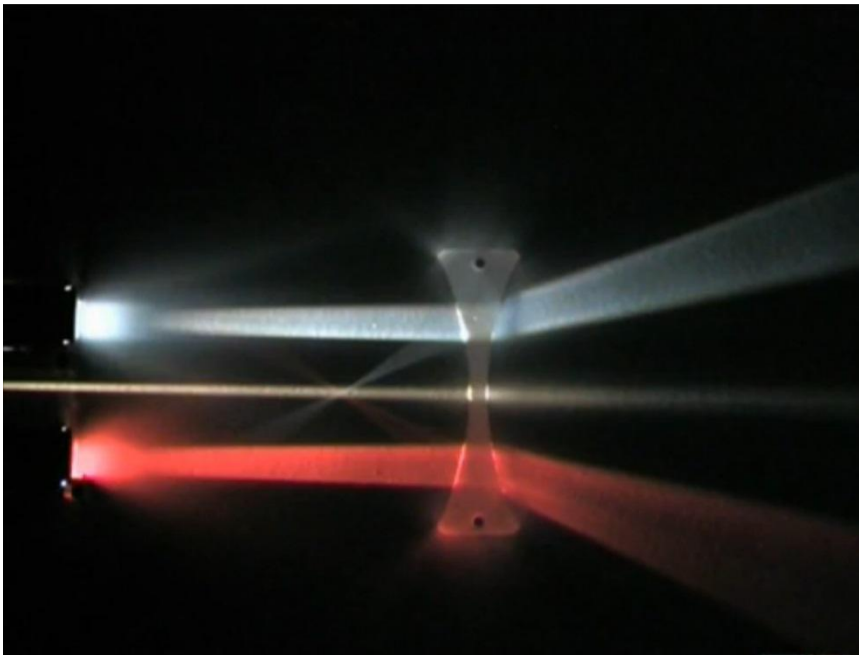


Рис.2.11. Розсіювальні лінзи.

Головна площина лінзи проходить через оптичний центр перпендикулярно головній оптичній осі. Оптичний центр лінзи є точкою, де перетинаються головна оптична вісь з площиною лінзи. Головний фокус збиральної лінзи є точкою на головній оптичній осі, в якій збираються промені, які падають паралельно головній оптичній осі після заломлення в лінзі. Оптична сила лінзи є фізичною величиною, що характеризує здатність лінзи заломлювати світло й дорівнює оберненій величині фокусної відстані.

Зображення в плоскому дзеркалі

Побудова зображення відрізка предмета АВ, що розміщений перед плоским дзеркалом. Припустимо, що предмет АВ паралельний дзеркалу. Проведемо із точки А предмета два променя: AM_1, що падає перпендикулярно до дзеркала, і АО, що утворить із нормалю NN кут ε. При цьому точку О виберемо так, щоб $M_1O = M_1M_2/2 = AB/2$. Після відбивання від дзеркала промінь AM_1 піде у зворотному напрямку M_1A, а промінь АО відіб'ється під кутом ε' і пройде через точку В. Продовження променів M_1A та ОВ перетнуться в точці А'. Аналогічно виходить зображення точки В.

Зображення предмета А є уявним, бо утворено продовженнями променів, що рівні предмету АВ.

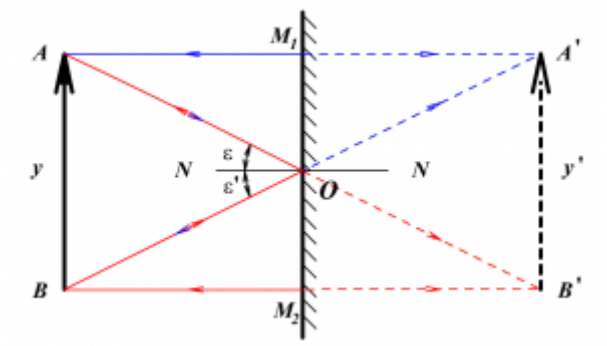


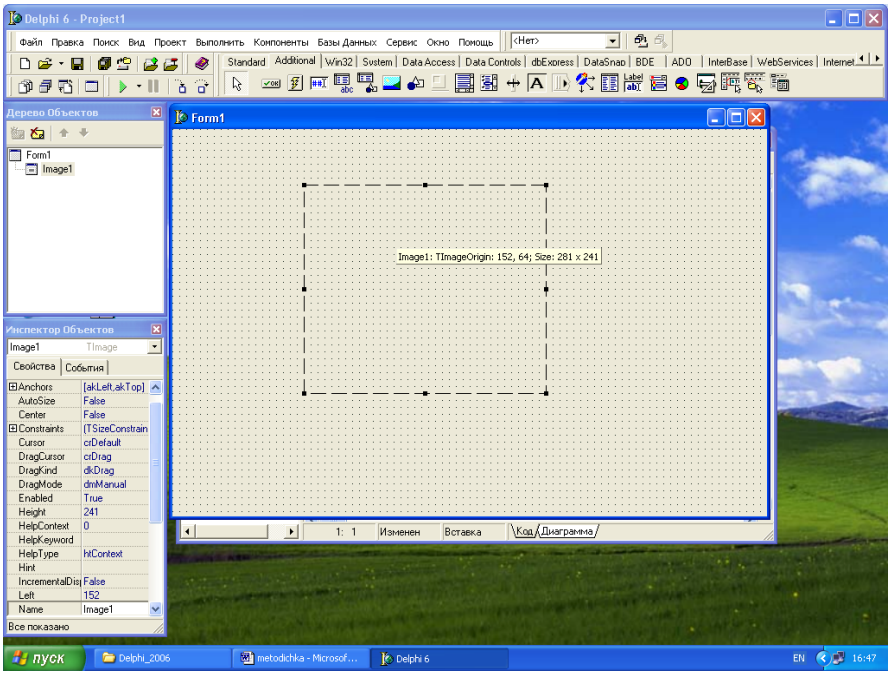
Рис.2.12. Зображення предмета, розташованого перед плоским дзеркалом.

Застосування компоненти Image.

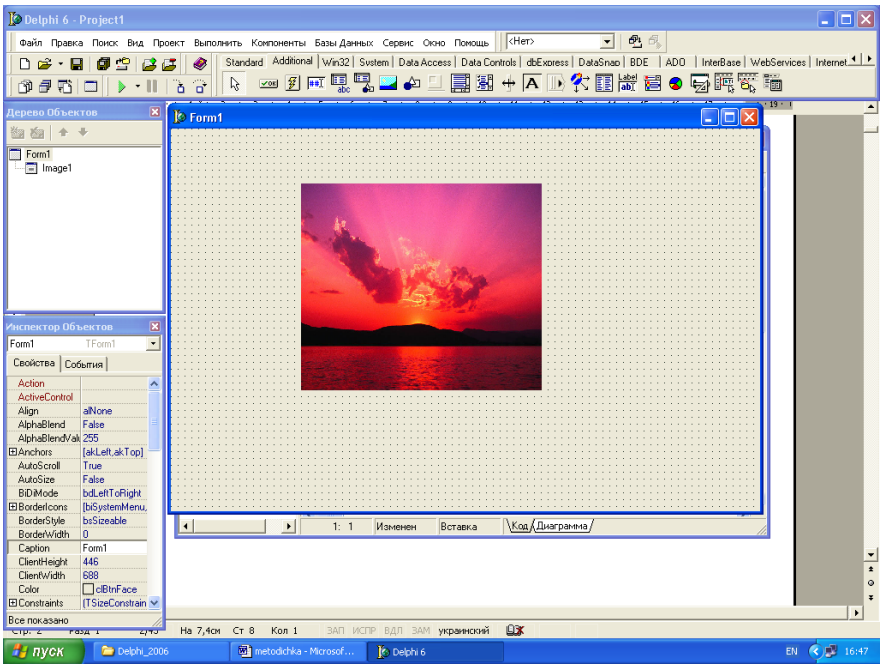
Об'єкт Image використовують для вставки графічних об'єктів з файлів типу *.bmp, *.emf, *.ico, *.wmf у форму. Використовують такі властивості даної компоненти (Мар'ян, Юркович, 2009):

Властивість	*Опис властивості*	*Приклади значень*
Align	Вирівнювання поля відносно об'єкта, що його містить (форми)	alBottom, alClient, alLeft, alNone, alTop
Width, Height	Ширина і висота вікна у пікселях	503, 224 (числове значення)
Name	Ім'я форми	Form1 (ідентифікатор)
Cursor	Вигляд вказівника миші на формі під час виконання програми	crDrag, crCross, crHelp, crArrow (перелічувальний тип)
Enabled	Доступність для дій об'єктів у формі під час	True, False

	виконання	
Left, Top	Координати лівого верхнього кутка вікна у пікселях	200, 108 (числове значення)
Visible	Видимість об'єкта	True, False
Center	Вирівнювання рисунка до центру відносно поля, що його містить	True, False
Picture	Ім'я графічного файлу	Задається у діалоговому вікні
Stretch	Приведення розміру зображення об'єкта до заданих розмірів об'єкта	True, False
Autosize	Приведення розміру об'єкта до реальних розмірів зображення	True, False



а)



б)

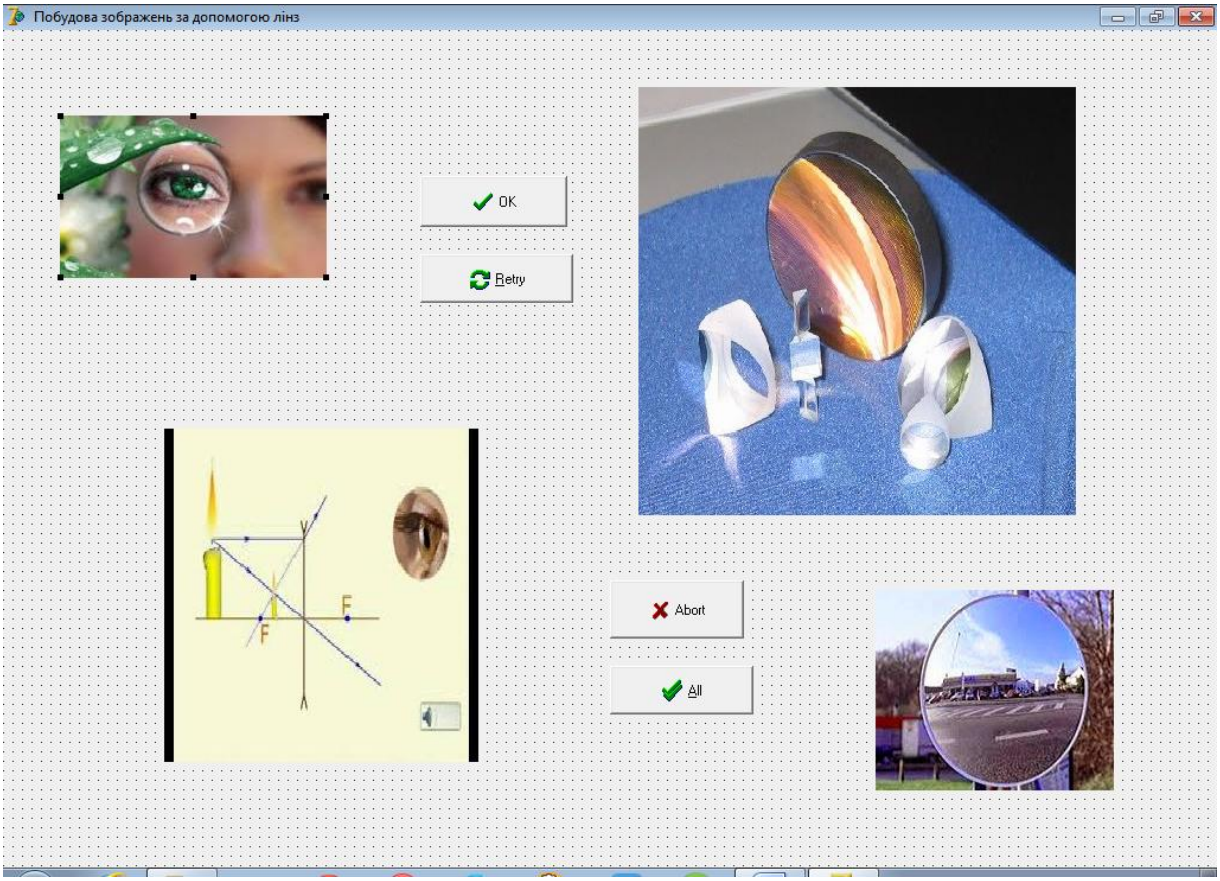
Рис.2.13. Приклад застосування компоненти Image: а) розташування компоненти на формі; б) використання властивостей Picture і Stretch.

Виконання роботи

1. Задати назву форми «Побудова зображень за допомогою лінз».

2. Створити візуальний інтерфейс форми з використанням об'єкта Image вкладки Additional (додаткові) і задати контур для майбутнього об'єкта.

3.Вставити рисунки за допомогою властивості Picture об'єкта Image1. Використати властивість Stretch (*Зауваження.* Під час накладання об'єктів може виникнути потреба використати команду Send To Back (переслати назад) чи Bring To Front (перенести наперед), які є в їх контекстових меню).



4.Поекспериментувати з властивостями Visible (видимість) зображень, Width зміна розмірів зображень, Left – розміщення зображень, Enabled – властивість доступності, виконуючи програму (див.Лістинг5).

5. Зберегти програму «Побудова зображень з допомогою лінз» і створити файл *.exe .

6. Перепрограмувати кнопки до нового призначення, використовуючи підказку: у тексті процедур, що описують роботу кнопок, можна скористатися командами вигляду:

If Image1. Visible = True **then… {**Якщо видимість = True**}**

2.6. Визначення оптичних характеристик збірної і розсіюючої лінз та побудова графіків з використанням компоненти Chart середовища програмування Delphi.

Мета роботи: ознайомитись з деякими методами визначення фокусних віддалей лінз, визначити оптичну силу вибраної лінзи.

Теоретичні відомості

Лінзу можна розглянути як систему двох заломлюючих поверхонь. Зображення предметів за допомогою ідеальної оптичної системи може бути побудоване без докладного дослідження ходу променів всередині систем. Для цього потрібно лише знати фокусну відстань і положення головних площин (Гулд, Тобочник, 1990).

Головними площинами ідеальної оптичної системи є спряжені площини, лінійне збільшення для яких дорівнює $\beta=+1$. У тонкій лінзі головні площини співпадають, і їх перетин з оптичною віссю дає оптичний центр. У залежності від форми лінзи, головні площини можуть знаходитись як всередині лінзи, так і зовні. Головним фокусом лінзи є промені, паралельні головній оптичній осі, які, заломлюючись у збірній лінзі, перетинаються в точці, що лежить на оптичній осі. Розрізняють передній головний фокус F і задній головний фокус F'. Відстані від головних площин до головних фокусів є фокусними відстанями. Згідно з правилом знаків, для збірної лінзи $f<0$, $f'>0$, для розсіюючої – $f>0$, $f'<0$.

Оптична сила товстої лінзи може бути розрахована за формулою (1):

$$\Phi = \frac{1}{f'} = (n-1)\left(\frac{1}{R_1} - \frac{1}{R_2}\right) + \frac{d(n-1)^2}{n \cdot R_1 R_2} \qquad (1)$$

де f' –задня фокусна відстань лінзи, R_1 і R_2 – радіуси кривизни заломлюючих поверхонь, n - показник заломлення матеріалу лінзи, d -товщина лінзи.

Відстані від головних площин до спряжених точок S та S' зв'язані формулою Гауса (2):

$$\frac{1}{f'} = \frac{1}{S'} - \frac{1}{S} \qquad (2)$$

При користуванні формулами (1) та (2) слід враховувати в кожній конкретній задачі знаки величин f, f ', R_1, R_2, S', S, згідно з діючим у геометричній оптиці правилом знаків. Формула (2) виконується як для товстої, так і для тонкої лінз.

Інший спосіб визначення фокусної відстані тонкої лінзи (3):

$$f' = \frac{SS'}{S - S'} = \frac{L^2 - l^2}{4L} \qquad (3)$$

Цей спосіб зручний тим, що експериментально вимірюються лише переміщення тонкої лінзи l та відстань L між двома спряженими площинами, в яких розташовані предмет і екран.

Для товстої лінзи формула (3) має вигляд:

$$f' = \frac{(L - \delta)^2 - l^2}{4(L - \delta)} \qquad (4)$$

де δ -відстань між головними площинами лінзи. Якщо для товстої лінзи відстань між головними площинами δ невідома, то цей спосіб непридатний для визначення f ' товстих лінз.

Способи виводу графічної інформації:

- Побудова графіків і діаграм (компонент Chart і ін.);
- Вивід наявних зображень (компоненти *Image*, *Shape*);
- Формування зображень програмним способом (об'єкт Canvas).

Компонент середовища Delphi Chart як засіб відображення даних

Компонент Delphi Chart розміщений на сторінці TeeChart Lite палітри компонент. Це багатий можливостями, дуже потужний компонент, що дозволяє будувати дво- і тривимірні діаграми на основі різних даних. Він має велику кількість різноманітних властивостей. Частина з них, у свою чергу, є об'єктами і володіють власними властивостями. Є контейнером об'єктів Series типу TChartSeries - серій даних, що характеризуються різними стилями

відображення. Кожна серія буде відповідати одній кривій або поверхні на графіку.

Значок на палітрі компонент має вид, зображений на рис. 2.14.

Рис. 2.14. Вигляд значка об'єкту TChart на сторінці TeeChart Lite у палітрі компонент.

Розташований компонент на формі буде мати вигляд, як на рис. 2.15.

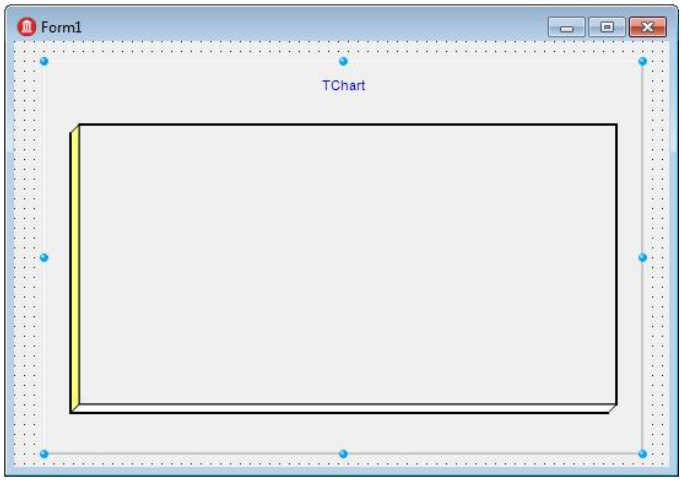


Рис. 2.15. Вигляд розташованого компоненту Chart на формі.

Створити діаграму можна також за допомогою майстра. Для запуску майстра з меню запускаємо File -> New -> Other і потрапляємо у вікно New Items, на вкладці Business вибираємо TeeChartWisard. Майстер потребує уточнення, на основі бази даних чи ні будуватиметься графік. Нехай він генерується програмно (перемикач Non Database Chart - не на основі бази даних). Наступний крок - вибір виду діаграми, вона може бути дво- або тримірна (вибір перемикача 2D або 3D) (Фаронов, 2012). Натиснувши Next вказуємо, чи потрібна нам легенда (опція Show Legend); жовті підказки, поруч з діаграмою, включає прапорець Show Marks. Натиснувши на кнопку Finish, ми отримаємо діаграму, заповнену випадково згенерованими числами (Мар'ян, Юркович, 2014) .

Налаштування властивостей компонента TChart відбувається в редакторі Editing Chart (рис. 2.16).

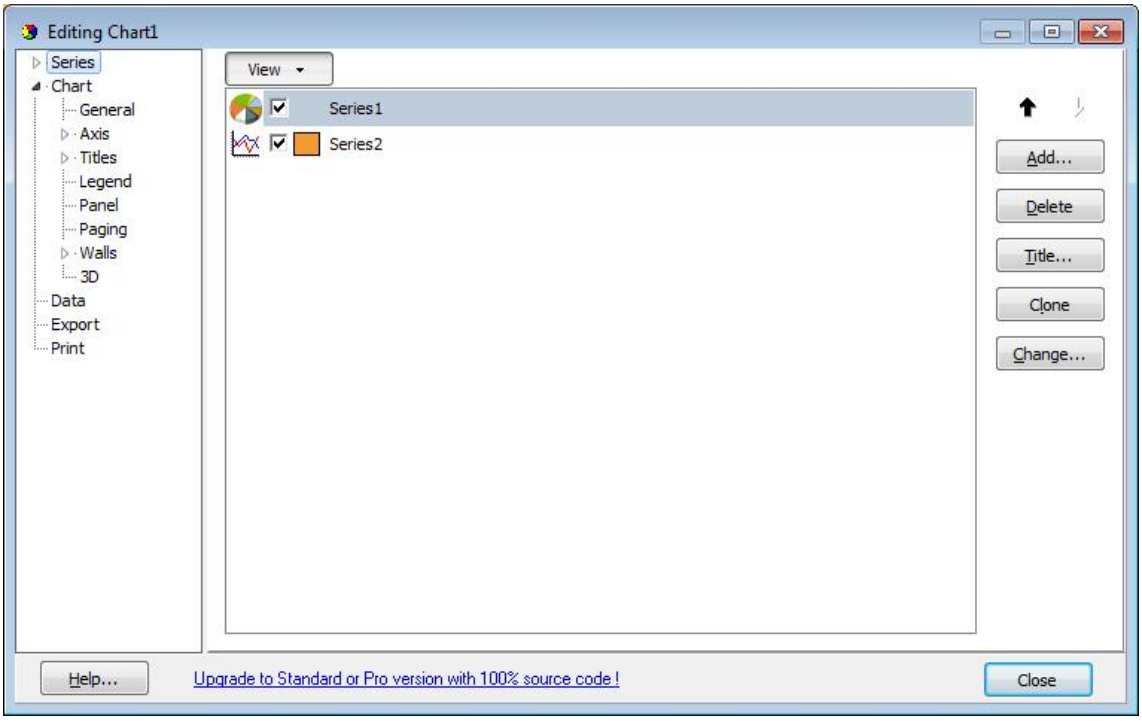


Рис. 2.16. Вигляд властивостей Editing Chart.

Викликати його можна подвійним клацанням по компоненту або використовуючи властивість SeriesList інспектора об'єктів.

Основні параметри діаграми визначаються на вкладці Chart (діаграма), вона, у свою чергу, складається з набору додаткових панелей.

Панель Ряд даних (Series) важлива, в неї можна додати на один графік кілька діаграм за допомогою кнопки Add. При цьому над значеннями даних можна виконувати операції, які задаються у вкладці Series -> Data Source вибравши function зі списку function.

Панель Загальні (General) містить такі елементи управління:

- кнопка Export - експорт зображення у файл;
- кнопка Print Preview - попередній перегляд і друк діаграми;
- панель Zoom – масштабування;
- панель AllowScroll - відповідає за прокрутку зображення.

Засоби панелі Осі (Axis) регулюють настройку координатних осей, заголовків, їх масштаб, крок пунктирної сітки і багато іншого (Мар'ян, Юркович, 2007).

Панель Заголовків (Titles) допомагає оформити заголовки.

Панель Легенда (Legend) відповідає за зовнішній вигляд і зміст легенди.

Панель Панель (Panel) задає оформлення панелі основи: колір і форму границі панелі (можна зробити так, що діаграма буде розташована на панелі, яка має градієнтну заливку).

Панель Сторінки (Paging), дозволяє розділити діаграми на сторінки. Для цього необхідно в полі (точки на сторінці) Points per Page підібрати відповідне значення.

Панель Границі (Walls) відповідає за колір і границі діаграми.

Панель 3D дає можливість налаштування 3D ефектів (зміна масштабу, положення в просторі), простим переміщенням повзунків управління.

Вкладка Ряди даних (Series) відповідає за оформлення кожного ряду даних (графіків доданих за допомогою вкладки Chart). Поточний ряд даних задається за допомогою списку. Найбільш важлива панель Джерело даних даних (Data Source). У ній можна вибрати відмову від генерації значень (No Data), створити випадкові значення (Random Values) або сформувати значення, як результат застосування функції (список Function) до значень рядів.

Основні властивості об'єкта Chart (Фаронов, 2012):

AllowPanning - визначає можливість користувача прокручувати спостережувану частину графіка під час роботи, натискаючи праву кнопку миші. Можливі значення: pmNone - прокрутка заборонена, pmHorizontal, pmVertical або pmBoth - дозволена відповідно прокрутка тільки в горизонтальному напрямку, тільки у вертикальному або в обох напрямках.

- AllowZoom - дозволяє користувачеві змінювати під час роботи масштаб зображення, вирізаючи фрагменти діаграми або графіка курсором миші.
- Title - визначає заголовок діаграми.
- Frame - визначає рамку навколо діаграми.

- Legend - легенда діаграми - список позначень.
- MarginLeft, MarginRight, MarginTop, MarginBottom - значення лівого, правого, верхнього і нижнього полів.
- BottomAxis, LeftAxis, RightAxis - ці властивості визначають характеристики відповідно нижньої, лівої і правої осей. Завдання цих властивостей має сенс для графіків і деяких типів діаграм.
- LeftWall, BottomWall, BackWall - ці властивості визначають характеристики відповідно лівої, нижньої і задньої граней області тривимірного відображення графіка.
- SeriesList - список серій даних, що відображаються у компоненті.
- View3d - дозволяє або забороняє тривимірне відображення діаграми.
- View3DOptions - характеристики тривимірного відображення.
- Chart3DPercent - масштаб тривимірності (товщина діаграми і ширина стрічок графіка).

Приклад діаграми, побудованої за допомогою об'єкта Chart, показано на рис. 2.17.

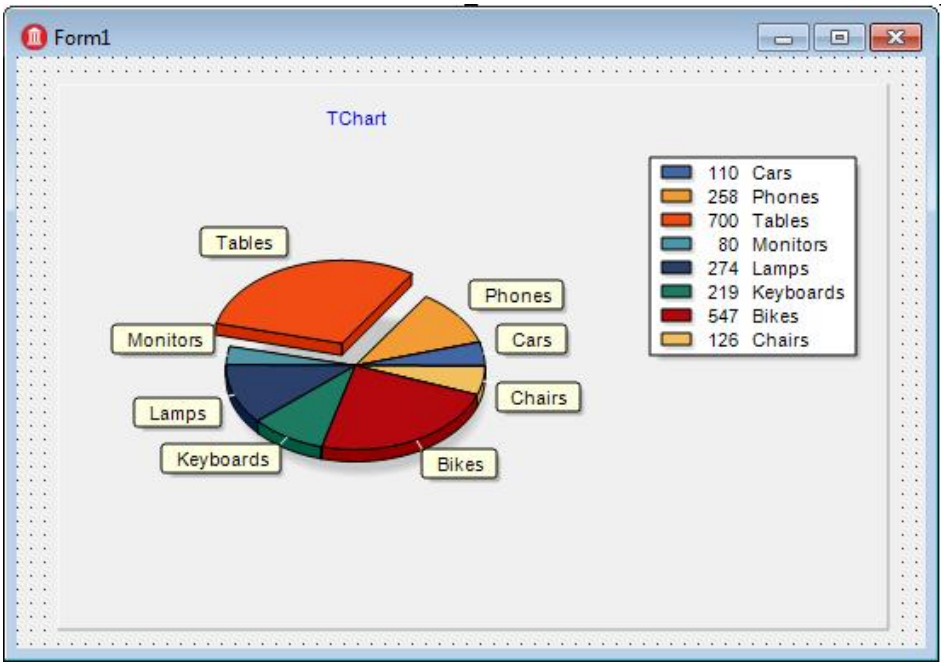


Рис. 2.17. Приклад діаграми, побудованої за допомогою об'єкта TChart.

Приклад коду створення двовимірної серії та додавання її до графіка:

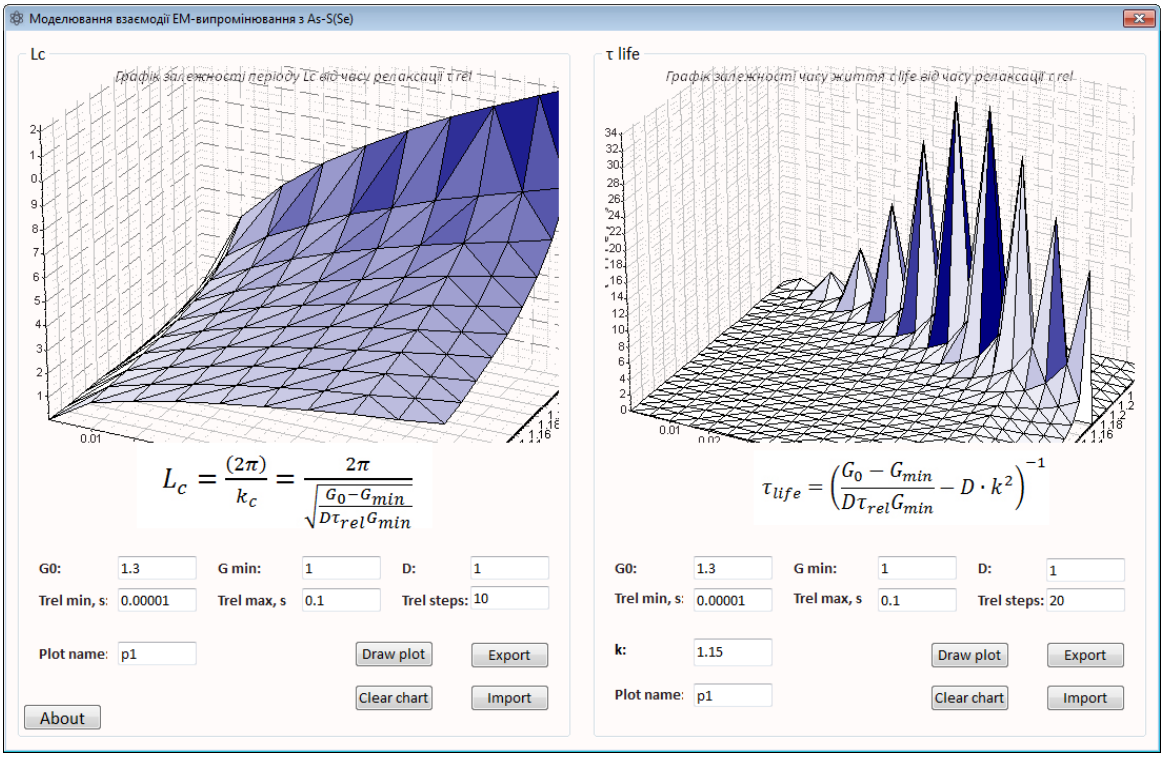
```
procedure Series(const _Chart: TChart);
var
  LineSeries: TLineSeries;
  x: double;
  y: double;
  i: integer;   //counter
begin
  LineSeries:= TLineSeries.Create(_Chart);  //create series

  for i := 0 to 10 do begin
        x := i;
        y := i * i;
        LineSeries.AddXY(x, y);
  end;
  _Chart.AddSeries(LineSeries);
end;
```

У даній програмі побудовано поверхню значень при зміні параметрів, яка дозволяє наочно побачити зміну функції при всіх можливих значеннях із заданого діапазону (Юркович, 2010). Для цієї поверхні використано тип серії TTriSurfaceSeries, приклад коду створення та додавання до графіка якої показаний нижче.

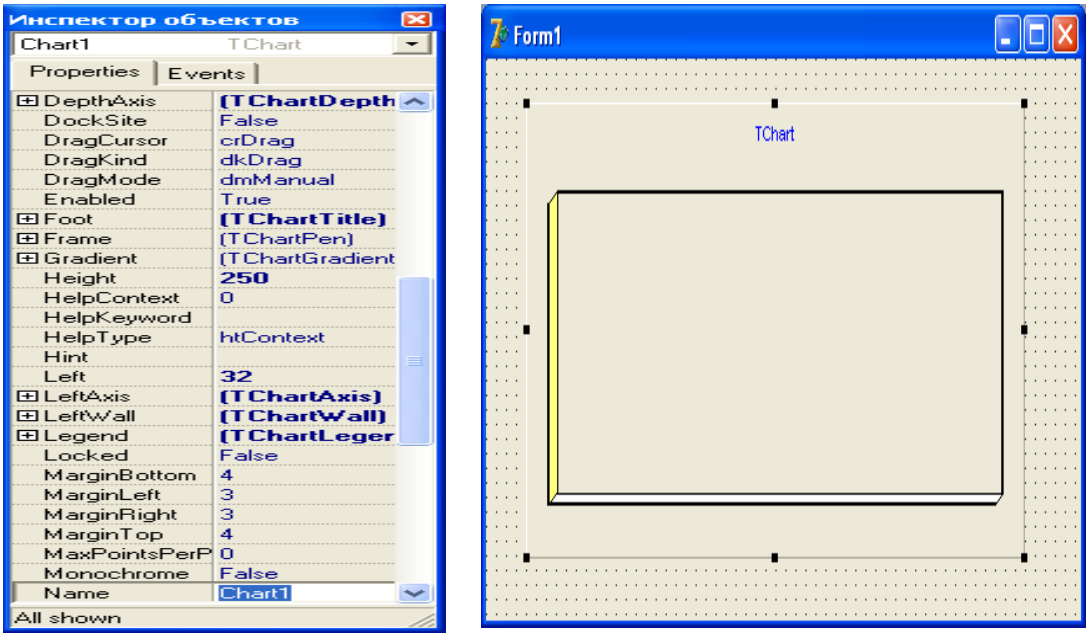
```
function PrepareSeries(const _Chart: TChart): PTTriSurfaceSeries;
var
        //pointer to TTriSurfaceSeries
        PTriSurfaceSeries: PTTriSurfaceSeries;
        x, y, z: double;
        //counters
        i, j: integer;
```

```
begin
  New(PTriSurfaceSeries);
  PTriSurfaceSeries^ := TTriSurfaceSeries.Create(_ChartLC);

        for i := 0 to 10 do
        begin
                x := x + i;
                for j := 10 do
                begin
                        y := x * x;
                        z := y * x;
                        PTriSurfaceSeries^.AddXYZ(x, y, z);
                end;
        end;

  PTriSurfaceSeries^.Title := 'PlotName';
  PTriSurfaceSeries^.TimesZOrder := 20;

  Result := PTriSurfaceSeries;
end;
```

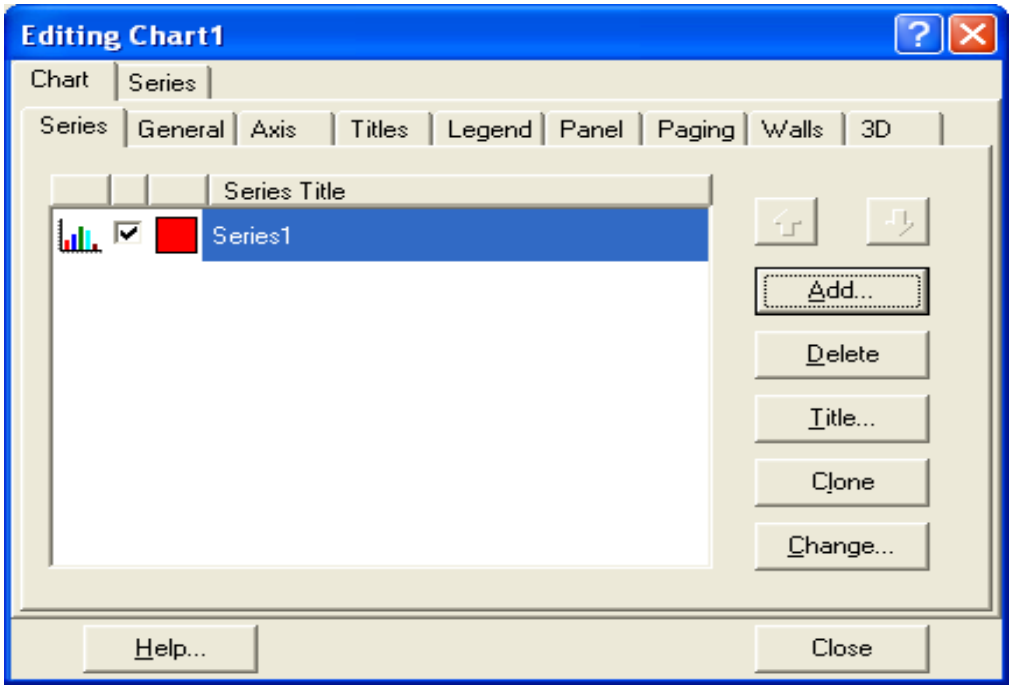
У розробленій програмі компонент Chart використано для побудови поверхні функції і порівняння отриманих даних при різних вхідних параметрах. Задані такі параметри: увімкнені тільки ліва, нижня і вісь глибини; задана сітка пунктиром і крапками; вибраний білий колір фону (Мар'ян, Юркович, 2015).



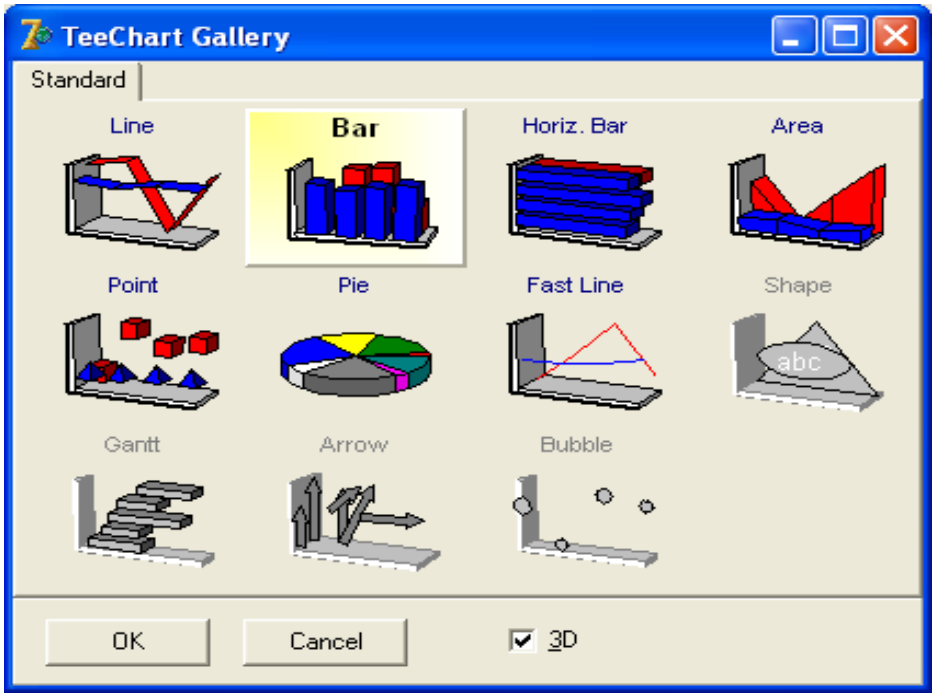
Вид компоненту Chart



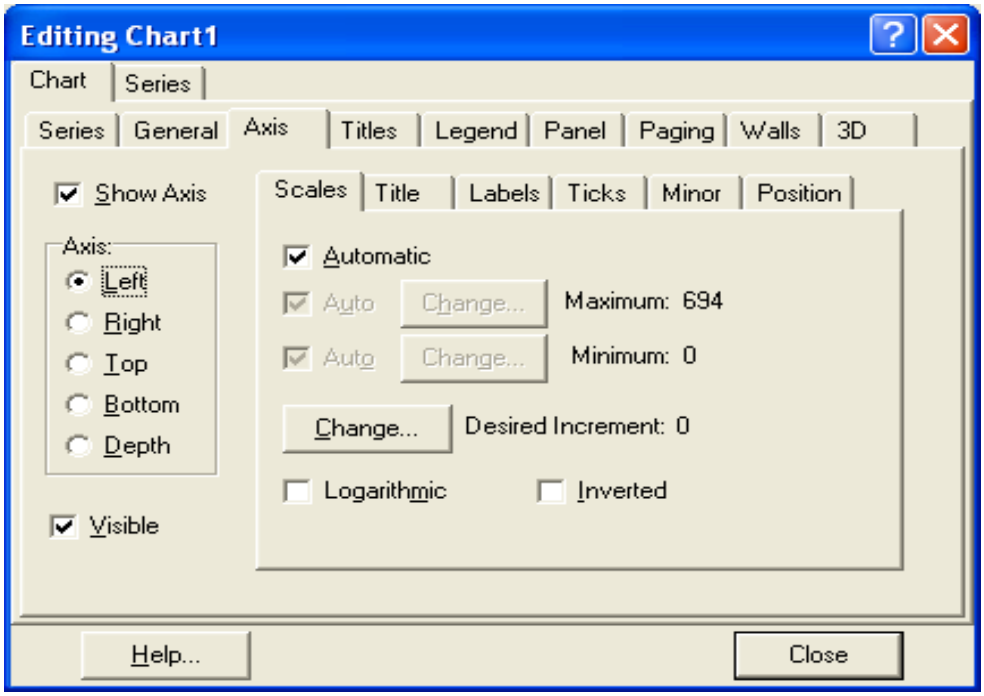
Побудова графіків і діаграм із можливістю зміни типу діаграми Change та додаванням серії даних Add:



Вибір типу діаграми:



Установка властивостей для осей координат (Axis):



Методи серій **Series**:

Ø**Clear** – очищає серію від занесених раніше даних;

Ø**Add** – дозволяє додати в діаграму нову точку:

Add(Const **AValue**:Double; Const **ALabel**:String; **AColor**:TColor);

Параметр **AValue** відповідає значенню, параметр**ALabel** – назва, буде відображатися на діаграме і в легенді, параметр **AColor –** колір діаграми.

Параметр**ALabel** необов'язковий, його можна задавати порожнім: ' ';

Ø**AddXY** – дозволяє додати нову точку в графік функції.

Дані для відображення передаються в Chart програмно, наприклад:

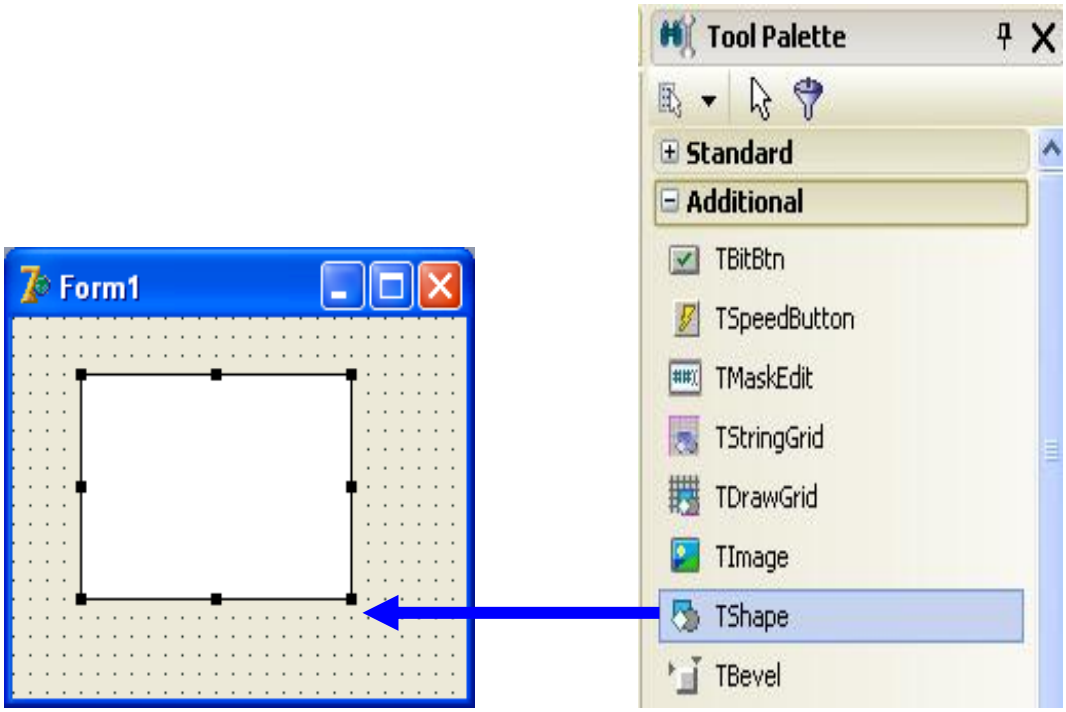
```
Series1.Clear; {очистити серію}
for i:=1 to N do
    Series1.addxy(i, A[i], '', clGreen).
```

Приклад лістингу побудови діаграми

```
procedure TForm1.FormCreate(Sender: TObject);
 var
 i: integer;
 begin
 Series1.Clear;
 for i:=(-30) to 30 do
```

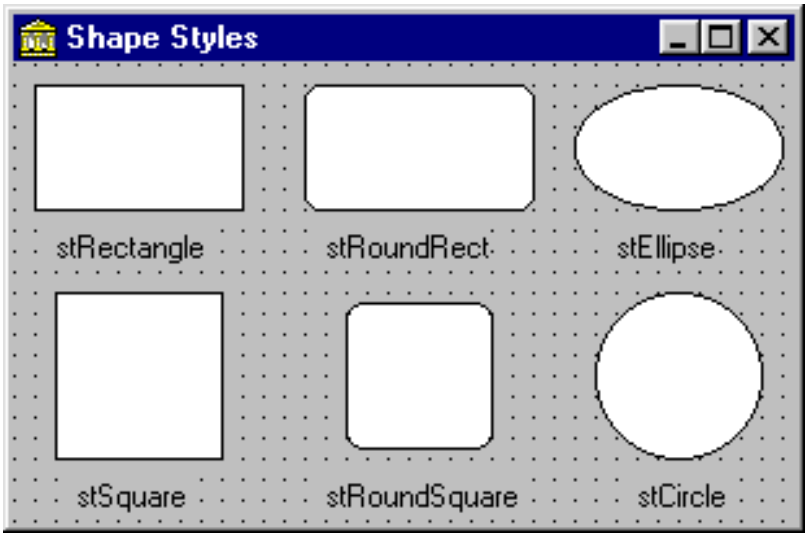
```
  Series1.AddXY(i, i*i, ' ', clBlue);
  end;
end.
```

Відображення геометричних фігур - компонента Shape.



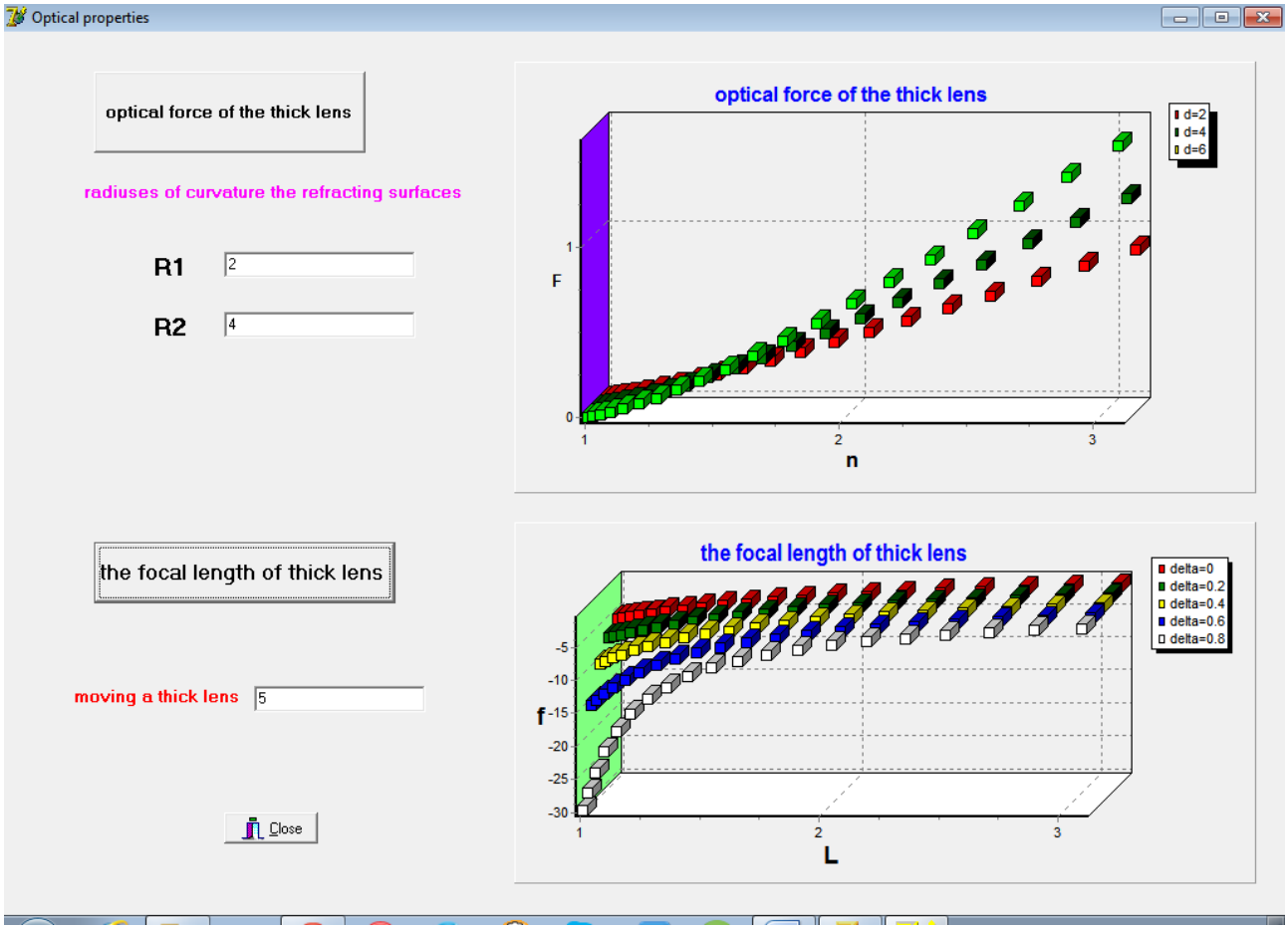
Основні властивості компонента Shape (Фаронов, 2012)*:*

Brush	Колір (.Color) і стиль (.Style) для заповнення фігури.
Pen	Колір (.Color), стиль (.Style), ширина (.Width) і спосіб виводу (.Mode) ліній фігури.
Shape	Вид геометричної фігури.



Виконання роботи.

Завдання1. Скласти програму в середовищі програмування Delphi моделювання оптичних характеристик збірної і розсіюючої лінз, побудувати графіки з використанням компонент Chart,Edit, Label, Button, BitBtn, Image та Object Inspector (див.Лістинг6).



Завдання2. Із використанням заданих формул показати інші графічні залежності оптичних характеристик збірної і розсіюючої лінз.

Закріплення.

1. Спряжені, кардинальні і особливі точки оптичної системи.
2. Інваріант Аббе. Формули Ньютона, Гаусса.
3. Збільшення оптичних систем (лінійне, поздовжне, кутове).
4. Залежність оптичної сили від параметрів лінзи.
5. Методи вимірювання фокусних відстаней лінзи.
6. Додавання двох оптичних систем.

2.7. Візуальне моделювання явища інтерференції в середовищі Delphi (C++).

Мета: Дослідити явище інтерференції світла та кільця Ньютона з використанням середовища програмування Delphi (C++).

Теоретичні відомості

Процес поширення електромагнітних коливань у просторі є електромагнітною хвилею (Сивухин, 1980). Існування електромагнітних хвиль є експериментально доведеним фактом та прямим наслідком спеціальної теорії відносності. У найпростішому випадку процес поширення коливань описується рівнянням плоскої монохроматичної хвилі

$$\xi(x,t) = \xi_0 \cos(\omega t - kx + \alpha), \qquad (1)$$

де ξ(x, t) – відхилення величини, що зазнає коливань, від положення рівноваги в момент часу t у точці з координатою x; ξ_0– максимальне відхилення; ω– циклічна частота; k– хвильове число (k= ω/υ= 2π/λ; υ– швидкість поширення хвилі; λ– довжина хвилі); α– початкова фаза. Для електромагнітної хвилі в ролі параметра ξ виступають напруженості електричного та магнітного полів.

Вираз(1) є розв'язком так званого хвильового рівняння, яке випливає з системи рівнянь Максвела в диференціальній формі:

$$\mathrm{rot}\vec{H} = \frac{\partial \vec{D}}{\partial t} + \vec{j}, \tag{2}$$

$$\mathrm{div}\vec{D} = \rho, \tag{3}$$

$$\mathrm{rot}\vec{E} = -\frac{\partial \vec{B}}{\partial t}, \tag{4}$$

$$\mathrm{div}\vec{B} = 0. \tag{5}$$

Вираз (2) фізично означає, що вихрове магнітне поле $\vec{H}$ створюється змінним у часі електричним полем з індукцією $\vec{D}$, а також сторонніми струмами з поверхневою густиною $\vec{j}$. Фізичний зміст рівняння (3) такий: електричне поле з індукцією $\vec{D}$ створюється сторонніми електричними зарядами, густина яких ρ. Рівняння (4) є законом електромагнітної індукції в диференціальній формі запису і свідчить про те, що вихрове електричне поле $\vec{E}$ породжується змінним у часі магнітним полем $\vec{B}$. І, нарешті, співвідношення (5) означає, що в природі не існує магнітних зарядів, на яких би починалися та закінчувалися силові лінії магнітного поля.

Сукупність виразів (2)-(5) - вісім скалярних рівнянь відносно шістнадцяти невідомих. Тому для розрахунку полів їх доповнюють так званими матеріальними співвідношеннями, які для ізотропних та однорідних середовищ мають такий вигляд:

$$\vec{B} = \mu\mu_0\vec{H}, \tag{6}$$

$$\vec{D} = \varepsilon\varepsilon_0\vec{E}, \tag{7}$$

$$\vec{j} = \sigma \vec{E}\,, \tag{8}$$

де μ– відносна магнітна проникність речовини; μ_0 – магнітна стала; ε– відносна діелектрична проникність речовини; ε_0– електрична стала; σ– питома провідність.

Нехай величини $\vec{E}, \vec{D}, \vec{B}, \vec{H}$ (тобто компоненти цих векторів) залежать лише від координати x та часу t. Нехай також у просторі, де поширюються коливання, немає сторонніх струмів та зарядів $(\vec{j} = 0, \rho = 0)$. Тоді з рівнянь(2) та(3) з урахуванням матеріальних співвідношень (6) і (7) випливає система:

$$\begin{cases} \dfrac{\partial H_z}{\partial x} = -\dfrac{\partial D_y}{\partial t} = -\varepsilon\varepsilon_0 \dfrac{\partial E_y}{\partial t}, \\ \dfrac{\partial E_y}{\partial x} = -\dfrac{\partial B_z}{\partial t} = -\mu\mu_0 \dfrac{\partial H_z}{\partial t}. \end{cases} \tag{9}$$

Тут враховано, що відмінними від 0 є лише похідні по x. Продиференціювавши перше рівняння в системі(9) за координатою x, а друге – за часом t

$$\begin{cases} \dfrac{\partial^2 H_z}{\partial x^2} = -\varepsilon\varepsilon_0 \dfrac{\partial^2 E_y}{\partial t \partial x}, \\ \dfrac{\partial^2 E_y}{\partial x \partial t} = -\mu\mu_0 \dfrac{\partial^2 H_z}{\partial t^2} \end{cases}$$

та одержано хвильове рівняння для H_z

$$\frac{\partial^2 H_z}{\partial x^2} = \varepsilon\varepsilon_0\mu\mu_0 \frac{\partial^2 H_z}{\partial t^2}\,. \tag{10}$$

Змінивши послідовність диференціювання аналогічно одержано:

$$\frac{\partial^2 E_y}{\partial x^2} = \varepsilon\varepsilon_0\mu\mu_0 \frac{\partial^2 E_y}{\partial t^2}\,. \tag{11}$$

Безпосередньою підстановкою можна переконатися, що вираз типу (1) є розв'язком диференціального рівняння(10) або (11). Така підстановка також показує, що швидкість поширення хвилі υ визначається таким чином:

$$\upsilon = \sqrt{\frac{1}{\varepsilon\varepsilon_0\mu\mu_0}} = \frac{c}{n}, \tag{12}$$

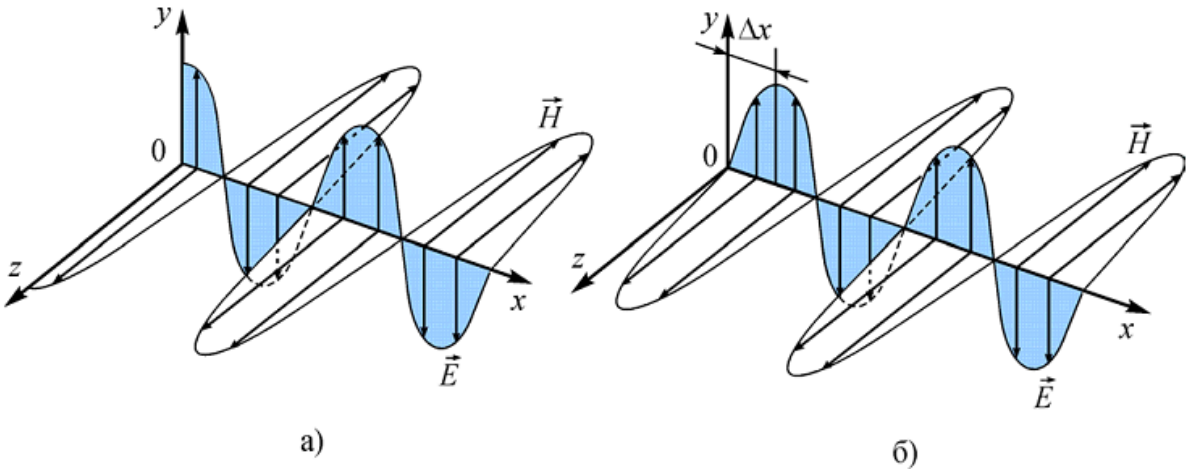
де:

$$c = \sqrt{\frac{1}{\varepsilon_0\mu_0}}, \tag{13}$$

$$n = \sqrt{\varepsilon\mu}, \tag{14}$$

де c – швидкість поширення електромагнітних хвиль у вакуумі; n– так званий показник заломлення середовища, в якому поширюється хвиля. Він показує, у скільки разів швидкість поширення хвиль у вакуумі більша за швидкість у середовищі.

Характер поширення електромагнітної хвилі ілюструє рис. 2.18а. Як видноз рисунка, електромагнітна хвиля є поперечною, оскільки напрямки коливань векторів $\vec{H}$ та $\vec{E}$ перпендикулярні до напрямку поширення хвиль. Картина поля в момент часу $t_1 = t + \Delta t$ (див. рис.2.18б) буде зміщеною вздовж осі ox на відстань $\Delta x = c \cdot \Delta t$ щодо зображення на рис.2.18а.



а) б)

Рис.2.18. Електромагнітна хвиля у момент часу t (а); у момент часу $t_1 = t + \Delta t$ (б).

Проявом електромагнітної природи світла є явище інтерференції. Інтерференцією світла є ефект перерозподілу енергії світлових хвиль у просторі внаслідок їх накладання. Досліджувати такий перерозподіл можна шляхом спостереження за освітленістю в тій чи іншій точці простору. Необхідною умовою існування інтерференції є когерентність хвиль, що накладаються. Когерентністю у широкому значенні називають узгоджене проходження хвильових або коливальних процесів. Отже, когерентними вважаються ті хвилі, які мають однакові частоти та незмінну різницю фаз $\delta\varphi$.

Унаслідок особливостей процесу випромінювання світлових хвиль речовиною світло від переважної більшості звичних нам джерел не є когерентним. Проте одну й ту саму хвилю неважко поділити на дві складові, примусити пройти різні відстані, а потім знову звести в одну точку (див. рис.2.19а). За таких умов різниця фаз буде визначатися різницею ($\tau_2 - \tau_1$) часів долання кожною з хвиль свого шляху. З рис.2.19а видно, що означені часові проміжки визначатимуться не лише шляхами s_1 та s_2, а й швидкостями $\upsilon_1 = c/n_1$, та $\upsilon_2 = c/n_2$. Тут c– швидкість світла у вакуумі, а n_1 та n_2– показники заломлення середовищ, через які прямуватимуть відповідно перша й друга частини розділеної хвилі. Тоді різниця фаз двох хвиль знаходиться як:

$$\delta\varphi = \omega(\tau_2 - \tau_1) = \omega\left(\frac{s_2}{\upsilon_2} - \frac{s_1}{\upsilon_1}\right) = \frac{\omega}{c}(s_2 n_2 - s_1 n_1) = \frac{\omega}{c}\Delta,$$

де $\Delta = s_2 n_2 - s_1 n_1$ називають оптичною різницею ходу, ω–циклічна частота хвилі.

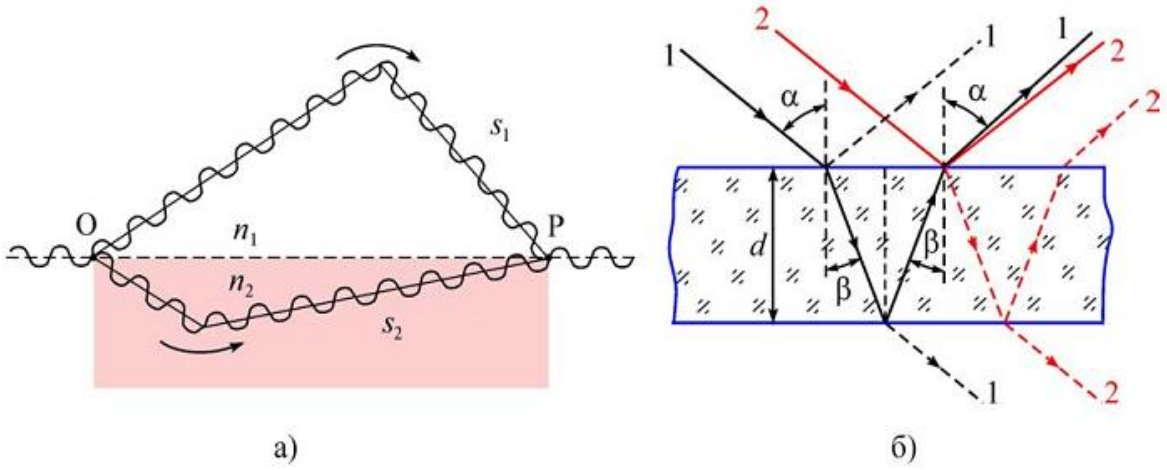


Рис.2.19. Схематичне зображення розділення світлової хвилі на дві частини з подальшим їх зведенням (а); проходження світлового пучка через тонку плівку(б).

Максимального значення енергія світлової хвилі досягатиме тоді, коли різниця фаз δ буде визначатись як ±2πm, m=(0,1,2,...). У цьому випадку обидві хвилі йтимуть в точку P в одній фазі.

$$\Delta_{\max} = \pm m\frac{c}{\nu} = \pm m\lambda_0,$$

де ν–частота хвилі; λ_0–довжина хвилі у вакуумі. Умова мінімуму інтерференції аналогічним чином запишеться як:

$$\Delta_{\min} = \pm\left(m + \tfrac{1}{2}\right)\lambda_0.$$

Ситуація з розділенням однієї хвилі на дві частини з подальшим їх зведенням у межах одного напрямку виникає під час проходження світлового пучка через тонку плівку (див. рис.2.19б). Під час проходження хвилі через межу розділу двох середовищ виникає як відбита, так і заломлена хвиля, енергія яких в сумі дорівнює енергії початкової хвилі. Частина хвильового фронту1, відбита від верхньої поверхні плівки, та частина хвильового фронту 2, що відбивається від нижньої поверхні, а потім заломлюється під час проходження через верхню, будуть збігатися за напрямком (на рис.2.19б 1 і 2

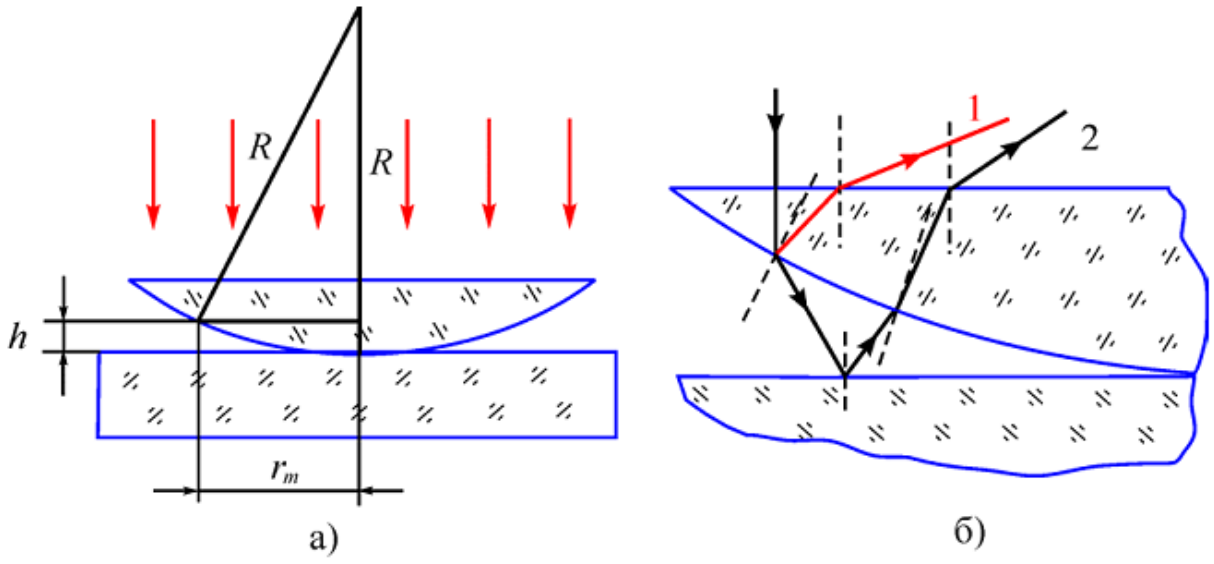
для наочності розділені). Неважко показати, що оптична різниця ходу буде дорівнювати:

$$\Delta = 2d\sqrt{n^2 - \sin^2\alpha} + \lambda_0/2\,,$$

де d–товщина плівки; α–кут падіння; n–показник заломлення плівки, а доданок $\lambda_0/2$ виникає за рахунок того, що під час відбивання від середовища з більшим показником заломлення фаза хвилі змінюється на π. Таким чином, залежно від кута падіння α або товщини плівки d для хвилі довжиною λ_0 буде виконуватись умова максимуму або мінімуму інтерференції. Саме цим пояснюється різнокольорове забарвлення плям нафтопродуктів на воді або яскраві кольори на мильній бульбашці.

Важливою умовою спостереження інтерференції є мала товщина плівки. Цього вимагає особливість природного світла, що є набором коротких хвильових імпульсів (цугів), у межах довжини яких зберігаються монохроматичність та сталість різниці фаз. Якщо оптична різниця ходу Δ перевищить довжину хвильових імпульсів, то порушиться умова когерентності, й інтерференція спостерігатися не буде.

Одним із прикладів системи заломлюючих поверхонь, в якій може спостерігатись інтерференція, є плоскоопукла лінза з великим радіусом кривини, покладена на плоскопаралельну пластину(див. рис.2.20).



а) б)

Рис.2.20. Оптична система для спостереження кілець Ньютона (а); зображення ходу світлових хвиль у такій системі (б).

Тут роль тонкої плівки виконує повітряний прошарок між лінзою та пластиною. Під час освітлення такої системи пучком світла, паралельним до її осі симетрії, виникає інтерференційна картина у вигляді концентричних кілець, яка отримала назву кілець Ньютона (Сивухин, 1980). Якщо світло монохроматичне, то матимуть місце лише кільця двох видів – світлі та темні. Освітлення білим світлом зумовить різнобарвну інтерференційну картину. Хід променів у такій системі зображений на рис.2.20б. Проте за умови, що радіус кривини лінзи великий, можна вважати, що всі зображені промені поширюються майже паралельно до осі симетрії. Оптична різниця ходу буде дорівнювати сумі подвоєної величині зазору між лінзою та пластинкою і половини довжини хвилі за рахунок відбиття від поверхні пластинки:

$$\Delta = 2h + \lambda_0/2 .$$

Знаходження радіусів кілець Ньютона. З рис.2.21а видно, що

$$R^2 = r_m^2 + \left(R - d\right)^2 ,$$
$$r_m^2 \approx 2hR ,$$

оскільки $h^2 << 2hR$. Тут індекс m вказує на порядок максимуму або мінімуму інтерференційної картини. Скориставшись цим виразом, для двох різних темних кілець (мінімумів інтерференції) одержано вираз для радіуса кривизни лінзи:

$$R = \frac{r_m^2 - r_k^2}{\left(m-k\right)\lambda_0} = \frac{d_m^2 - d_k^2}{4\left(m-k\right)\lambda_0} ,$$

де d_m, d_k– діаметри m-го та k-го темних кілець. Цей вираз дозволяє знаходити величину R за результатами двох вимірювань.

Іншу методику визначення величини R можна здійснити з урахуванням такої обставини: за рахунок сили тяжіння та механічних напружень у фіксуючій

обоймі лінзи в області контакту відбувається деформація лінзи та плоскопаралельної пластини. Про це свідчить залежність розмірів інтерференційної картини від ступеня фіксації системи спеціальними гвинтами.

Згідно з рис.2.21а радіус m-го темного кільця визначається співвідношенням:

$$r_m^2 = R^2 - (R - h - \delta)^2,$$

а радіус темної плями– співвідношенням

$$r_0^2 = R^2 - (R - \delta)^2,$$

де δ–величина деформації лінзи та плоскопаралельної пластини у місці контакту; r_0– радіус темного кола в центрі інтерференційної картини.

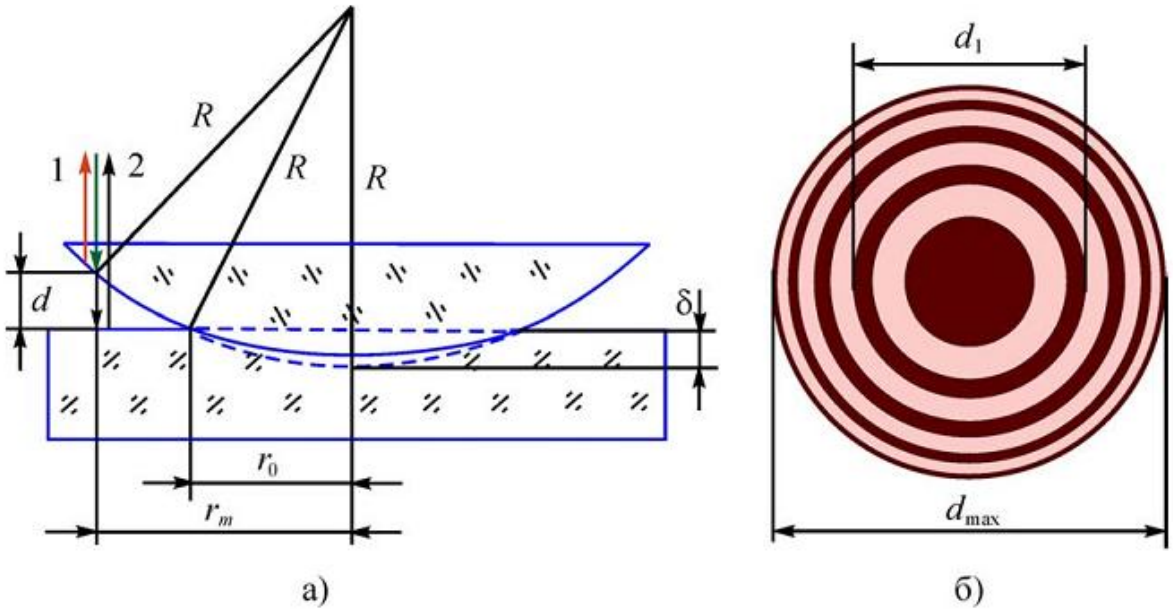


Рис.2.21. Обчислення радіуса кривизни лінзи з урахуванням деформації (а); інтерференційна картина «кілець Ньютона» (б).

З урахуванням умов $r_m >> h$; $r_m >> \delta$; $r_0 >> \delta$ знайдено:

$$2Rh = r_m^2 - r_0^2 .$$

Умовою мінімуму інтерференційної картини буде $2h = m\lambda_0$. Тому формула для розрахунку радіуса лінзи має такий вигляд:

$$R = \frac{r_m^2 - r_0^2}{m\lambda_0} = \frac{d_m^2 - d_0^2}{4m\lambda_0},$$

де d_0– діаметр темного кола в центрі інтерференційної картини.

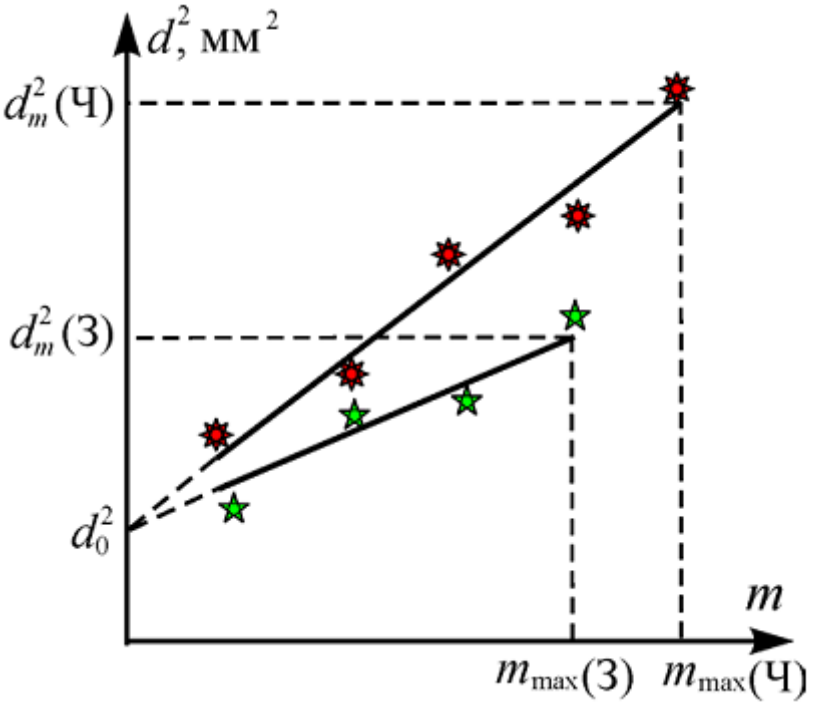


Рис.2.22. Визначення радіуса кривизни лінзи за експериментальними даними.

Оцінити похибку визначення радіуса кривизни лінзи R можна у такий спосіб:

$$\varepsilon = \frac{\Delta R}{R} = \frac{2d_m\Delta d_m + 2d_0\Delta d_0}{d_m^2 - d_0^2} + \frac{\Delta\lambda_0}{\lambda_0}.$$

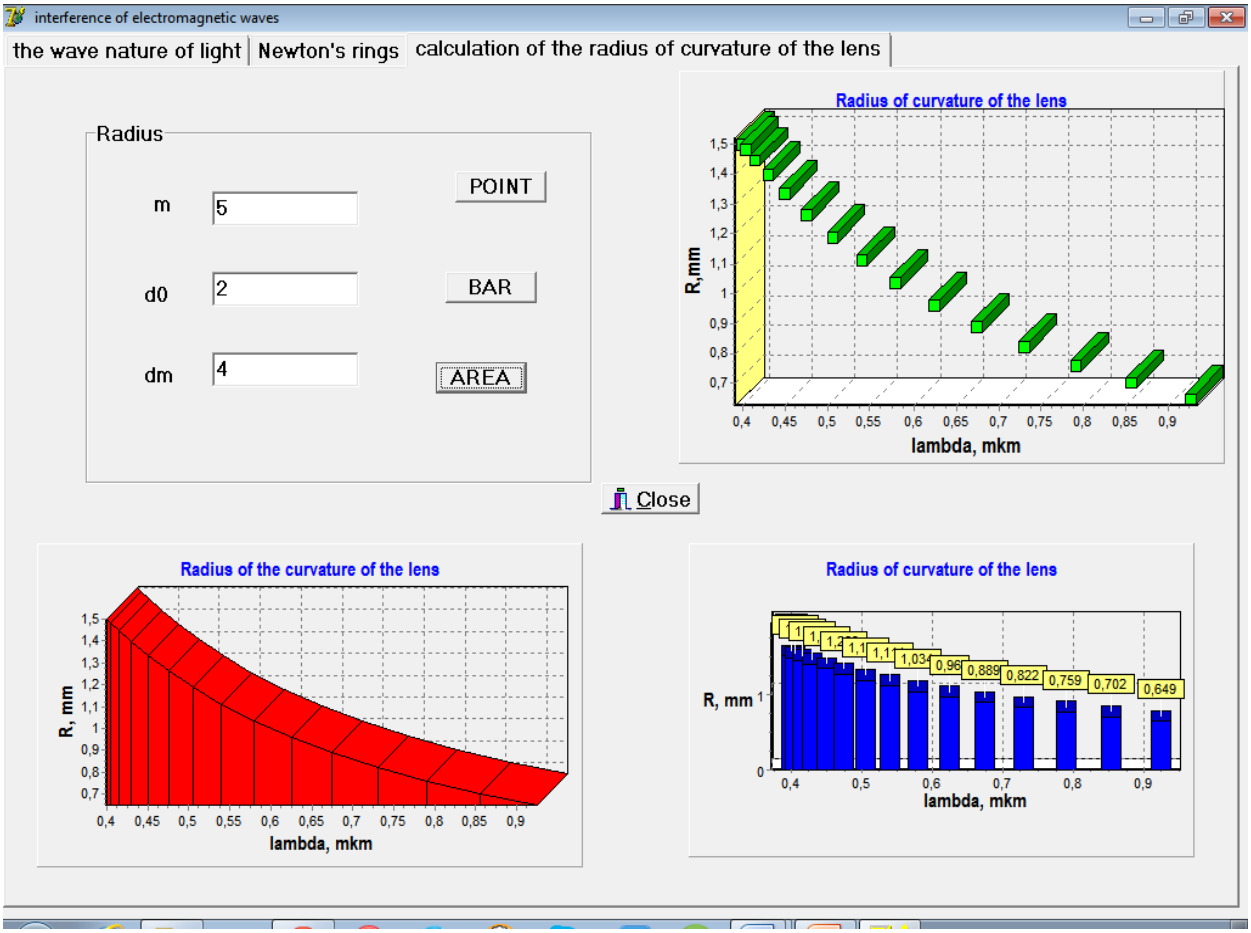
Тут ураховано, що похибка вимірювання констант дорівнює нулю і всі доданки у чисельнику мають бути більше нуля. Точне значення $d_0\Delta d_0$ невідоме, але матиме той самий порядок, що й $d_m\Delta d_m$. Для оцінки ε можна записати, що $d_0\Delta d_0 = d_m\Delta d_m$. Таким чином, абсолютна похибка вимірювання радіуса кривизни лінзи запишеться як:

$$\Delta R = R\left[\frac{4d_m\Delta d_m}{d_m^2 - d_0^2} + \frac{\Delta\lambda_0}{\lambda_0}\right].$$

Виконання роботи

1.Скласти програму для візуального моделювання явища інтерференції світла та кілець Ньютона з використанням середовища програмування Delphi за таким алгоритмом:

- помістити на форму компоненти PageControl, Label, Edit, GroupBox, Chart, Button, BitBtn;
- виконати редагування компоненти Chart;
- введення вихідних значень виконати в Edit;
- задати обробку подій для кнопок Button, Bitbtn (див.Лістинг7);



2.Побудувати графічні залежності радіуса кривизни лінзи від довжини хвилі падаючого світла з використанням різних типів серій: точки, лінії, сектори, бари, площі.

РОЗДІЛ ІІІ. КРИТЕРІЇ ТЕХНОЛОГІЧНОСТІ: КОНЦЕПТУАЛЬНІСТЬ, СИСТЕМНІСТЬ, ВІЗУАЛІЗАЦІЯ

3.1. Розробка тестових завдань з оптики (контрольно - оцінювальний етап) в середовищі Delphi.

Комп'ютерне тестування:"Оптика. Світлові хвилі".

1. Яка пара явищ найбільш яскраво виявляє квантові властивості світла.

а) фотосинтез і поляризація;

б) фотоефект і тиск світла;

в) дифракція та інтерференція.

2. Які з наведених нижче виразів відображають закон заломлення світла ?

а) $\frac{\cos\alpha}{\cos\gamma} = \frac{n_2}{n_1}$;

б) $\frac{\angle\alpha}{\angle\gamma} = \frac{n_2}{n_1}$;

в) $\frac{\sin\alpha}{\sin\gamma} = \frac{n_2}{n_1}$;

3. Який характер світлових хвиль ?

а) поперечні;

б) повздовжні;

в) можуть бути як повздовжніми, так і поперечними.

4. Які явища пояснюються інтерференцією світла ?

а) райдужне забарвлення тонких мильних, масляних плівок;

б) відхилення світлових променів в область геометричної тіні;

в) затемнення Місяця.

5. Яке зображення дає плоске дзеркало?

а) обернене, дійсне;

б) пряме, уявне;

в) обернене, збільшене.

6. Дифракцією світла називають…

а) взаємне посилення або послаблення двох когерентних світлових хвиль;

б) розкладання світла в спектр при проходженні через трикутну призму;

в) огинання світлом перешкод.

7. Як зміниться частота червоного випромінювання при переході світла з повітря у воду?

а) збільшується;

б) зменшується;

в) не змінюється.

8. Оптичну силу лінзи вимірюють у …

а) люменах ;

б) діоптріях;

в) ньютонах.

9. Якого кольору побачив би спостерігач небо, потрапивши на Місяць ?

а) блакитне;

б) чорне;

в) червоне.

10.Для яких променів – червоних чи фіолетових – буде більша головна фокусна відстань збиральної лінзи ?

а) червоних;

б) фіолетових;

в) однакова.

11. Зазначте ту пару явищ, у якої найбільш яскраво виявляються хвильові властивості світла.

а) дисперсія і відбивання;

б) поляризація і заломлення;

в) дифракція та інтерференція.

12. Які вирази відображають закон відбивання світла ?

а) $\angle a_{\text{падіння}} = \angle \beta_{\text{відбивання}}$;
б) $\angle a_{\text{падіння}} > \angle \beta_{\text{відбивання}}$;
в) $\angle a_{\text{падіння}} < \angle \beta_{\text{відбивання}}$.

13. Яку з формул називають формулою тонкою лінзи ?

а) $\frac{1}{D}=\frac{1}{d}+\frac{1}{f}$;

б) $\frac{1}{F}=\frac{1}{d}+\frac{1}{f}$;

в) $\frac{1}{D}=\frac{1}{d}-\frac{1}{f}$.

14. Яке явище доводить поперечність світлових хвиль ?

а) дисперсія;

б) відбивання;

в) поляризація.

15. При переході світлового променя з менш оптично густого середовища в більш густе, кут заломлення –

а) менший кута падіння;

б) більший кута падіння;

в) рівний куту падіння.

16. Як зміниться довжина жовтого випромінювання при переході світла з повітря у воду?

а) збільшується;

б) зменшується;

в) не змінюється.

17. При збільшенні розмірів джерела світла, що освітлює предмет, розміри його тіні …

а) зменшуються, а напівтіні збільшуються;

б) збільшуються, а напівтіні зменшуються;

в) збільшуються, а напівтіні не змінюються .

18. Деякі предмети мають білий колір, тому що…

а) відбивають всі кольори білого світла;

б) поглинають усі кольори білого світла;

в) розсіюють біле світло.

19.При накладанні двох світлових хвиль з однаковою частотою та постійною різницею фаз спостерігається …

а) відбивання світла;

б) дифракція світла;

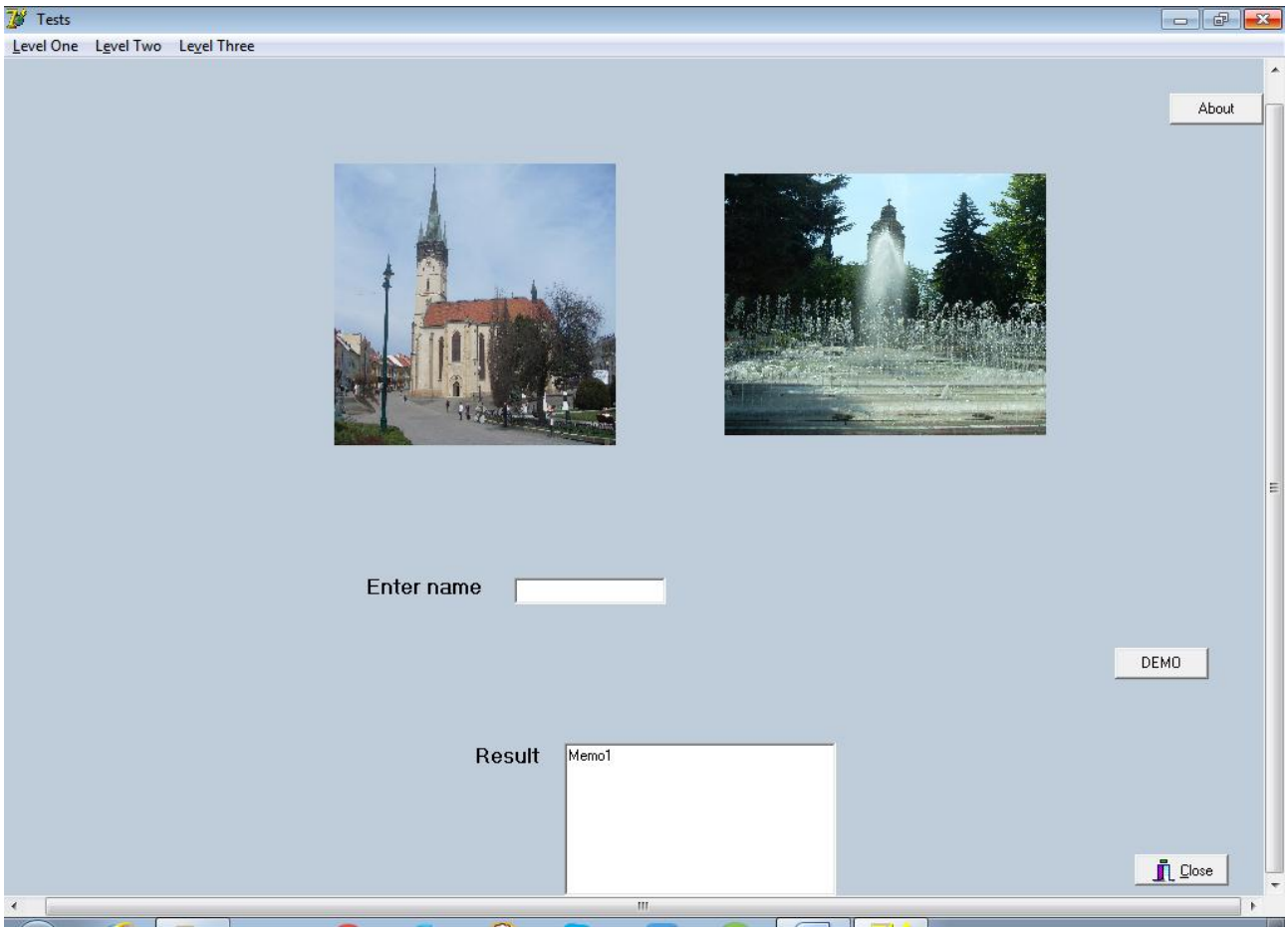
в) інтерференція світла.

20. Оптична сила лінз в окулярах 2 дптр. Яка фокусна відстань лінз? Для короткозорого чи далекозорого ока будуть ці окуляри?

а) 0,5м, короткозорого;

б) 2м, далекозорого;

в) 0,5м, далекозорого.



Інтерфейс програми комп'ютерного тестування розділу фізики "Оптика. Світлові хвилі" (Лістинг8).

3.2. Аналіз відповідності розробленого навчального матеріалу критеріям технологічності (концептуальність, системність, керованість, ефективність, відтворюваність, візуалізація).

Концептуальність розробленого інноваційного підходу викладання фізики випливає з теоретико-методологічних досліджень, спрямованих на досягнення позитивного результату формування професійно мобільного педагога – викладача фізики. *Системність* приведеного підходу пов'язана з логікою освітнього процесу, взаємозв'язком всіх його частин, професійно орієнтованим змістом навчального матеріалу, цілісністю, на основі яких і твориться процес формування професійної мобільності майбутніх педагогів у фізиці. *Керованість* (регулювання) приведеного підходу формування професійно мобільного викладача фізики припускає можливість цілевизначення, планування, організації навчально-пізнавальної діяльності, оцінки цієї діяльності, прогнозування процесу формування професійної мобільності майбутніх педагогів професійного навчання, поетапної діагностики, варіювання засобами й методами з метою корекції результатів (Мар'ян, Шебень, Юркович, 2016). *Ефективність* визначається здатністю майбутніх педагогів професійного навчання бути конкурентоздатними на ринку праці, виявом мотиваційної готовності до професійної мобільності та сформованістю ключових компетенцій. *Відтворюваність* пропонованої нами педагогічної технології формування майбутніх професійно мобільних педагогів професійного навчання пов'язана з можливістю масового застосування її у вищих навчальних закладах за допомогою включення в структуру професійної підготовки майбутнього педагога професійного навчання в умовах сучасного освітнього процесу й можливості її реалізації на основі одержаних і перевірених дослідно-експериментальним шляхом результатів.

Список використаних джерел

1. Гулд Х., Тобочник Я. Компьютерное моделлирование в физике. М., Мир. 1990. – 400 с.
2. Мар'ян М.І., Юркович Н.В. Програмування в середовищі DELPHI: компоненти image, paintbox, chart, dbchart та їх застосування. Ужгород, вид-во «Говерла» УжНУ, 2007.- 77с.
3. Мар'ян М.І., Юркович Н.В. Графічні редактори і графічний WEB-дизайн. Ужгород, вид-во «Говерла» УжНУ, 2008.-126с.
4. Мар'ян М.І., Юркович Н.В. Основи програмування у візуальному середовищі DELPHI. Частина 2. Ужгород, вид-во «Говерла» УжНУ, 2009.-64с.
5. Мар'ян М.І., Юркович Н.В. Об'єктно-орієнтоване візуальне програмування в середовищах Delphi, Java, C++. Ужгород, вид-во «Гражда», 2014. -65с.
6. Мар'ян М.І., Юркович Н.В. Комп'ютерне моделювання і програмування в середовищах Delphi, Java, C++, Ruby. Ужгород, вид-во «Гражда», 2015. -56с.
7. Мар'ян М., Шебень В., Юркович Н. Інноваційні підходи до викладання фізики з використанням комп'ютерного моделювання. Педагогічні інновації у фаховій освіті. Випуск 1(7). Ужгород, 2016, с.136-141.
8. Сивухин Д.В. Общий курс физики.Т.IV Оптика. – Москва, Наука. 1980.-752с.
9. Таненбаум Э., Уэзеролл Д. Компьютерные сети. 5-е изд.СПб.: Питер, 2012.-960с.
10. Фаронов В. Delphi. Учебный курс.- Санкт-Петурбург, Питер. 2012. – 507 с.
11. Юркович Н.В. Лекції з курсу „Обчислювальна техніка і програмування". Частина І: Програмування в середовищі Delphi. Ужгород, вид-во «Гражда», 2010. -86 с.

12. Bucknall, J. The Tomes of Delphi: Algorithms and Data Structures. Wordware Publishing, Inc. 2001.-P.545.
13. Cantu, M. Delphi 2009 Handbook. Wintech Italia Srl. 2008.- 400P.
14. Cavinato M., Giliberti M., Perotti L. Cross section for (nearly) everyone. Canadian Journal of Physics.- 2015.- 93(12),-p.1555-1560.
15. Huffman D. Effect of explicit problem solving instruction on high school students' Problem-solving performance and conceptual understanding of physics. Journal of Research in Science Teaching.- 1997.- 34(6), p.551 – 570.
16. Kuo E., Hull M., Gupta A., Elby A. How Students Blend Conceptual and Formal Mathematical Reasoning in Solving Physics Problems. Science Education.- 2013.-97(1),-p. 32–57.
17. Leung A., Terrana A., Jerzak S. Students' opinions on the educational value of physics laboratories: a cross-sectional survey. Canadian Journal of Physics.- 2016.- 94(9),-p. 913-919.
18. Nicolis, G., & Prigogin, I. Exploring Complexity. An introduction. New York: Freeman. 1989. – P. 344.

ДОДАТОК. Лістинги програм.

Лістинг1. Комп'ютерне моделювання оптичних параметрів законів відбивання та заломлення світла.

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, ExtCtrls;
type
  TForm1 = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Memo1: TMemo;
    Button1: TButton;
    Memo2: TMemo;
    Edit2: TEdit;
    Edit3: TEdit;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    BitBtn1: TBitBtn;
    Button2: TButton;
    Panel1: TPanel;
    Panel2: TPanel;
    Panel3: TPanel;
    Panel4: TPanel;
    Panel5: TPanel;
    Panel6: TPanel;
    Panel7: TPanel;
    procedure Button1Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
implementation
{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
 Const c=3E+8;
```

```
 Var v,n:real;
 begin
   n:=strtofloat(edit1.text);
   v:=c/n;
   memo1.Lines.Clear;
   memo1.Lines.Add('V='+ floattostr(v)+' m/s');
end;

procedure TForm1.BitBtn1Click(Sender: TObject);
begin
close;
end;

procedure TForm1.Button2Click(Sender: TObject);
Const c=3E+8;
Var f,n,lambda:real;
begin
  n:=strtofloat(edit2.text);
  lambda:=strtofloat(edit3.text);
  f:=c/(n*lambda);
  memo2.Lines.Clear;
  memo2.Lines.Add('f='+floattostr(f)+' 1/s')
end;
end.
```

Лістинг2. Комп'ютерне моделювання закону Снеліуса в середовищі Delphi з використанням компоненти PaintBox.

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Menus, ExtCtrls;
type
  TForm1 = class(TForm)
    PaintBox1: TPaintBox;
    MainMenu1: TMainMenu;
    Mirrorreflection1: TMenuItem;
    Diffusionreflection1: TMenuItem;
    Snellslaw1: TMenuItem;
    Illustration1: TMenuItem;
    Incidentray1: TMenuItem;
    Perpendicular1: TMenuItem;
    Reflectedray1: TMenuItem;
    Exit1: TMenuItem;
    Incidentray2: TMenuItem;
    Perpendicular2: TMenuItem;
    Reflectedray2: TMenuItem;
    Incidentray3: TMenuItem;
    Perpendicular3: TMenuItem;
    Refractedray1: TMenuItem;
```

```
    Exit2: TMenuItem;
    Mirrorreflection2: TMenuItem;
    Diffusionreflection2: TMenuItem;
    Snellslaw2: TMenuItem;
    Exit3: TMenuItem;
    Label1: TLabel;
    Label2: TLabel;
    Edit1: TEdit;
    Edit2: TEdit;
    E1: TMenuItem;
    Label3: TLabel;
    Edit3: TEdit;
    procedure Incidentray1Click(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure Perpendicular1Click(Sender: TObject);
    procedure Reflectedray1Click(Sender: TObject);
    procedure Perpendicular2Click(Sender: TObject);
    procedure Incidentray2Click(Sender: TObject);
    procedure Reflectedray2Click(Sender: TObject);
    procedure E1Click(Sender: TObject);
    procedure Incidentray3Click(Sender: TObject);
    procedure Perpendicular3Click(Sender: TObject);
    procedure Refractedray1Click(Sender: TObject);
    procedure Snellslaw2Click(Sender: TObject);
    procedure Diffusionreflection2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
uses Unit2, Unit3;
{$R *.dfm}

procedure TForm1.Incidentray1Click(Sender: TObject);
const pi=3.14;      //  description constants
var angle:integer;  // description of variables
angle1:real;
factor:integer;
x,y:integer;
x0,y0:integer;
begin
    angle:=StrToInt(Edit1.Text); // input data
    angle1:=pi*angle/180;
    with form1.PaintBox1.Canvas do   // Graphic Connection class
    begin
    Brush.Color:=clred;
    Pen.Color:=clyellow;
    pen.Width:=5;
    rectangle (1,1,700,700);
```

```
   x0:=10;
   y0:=10;
   moveto(x0,y0);   //  output beams
   x:=round(400*sin(angle1));
   y:=round(400*cos(angle1));
   Pen.Color:=clgreen;
   Pen.Width:=4;
   Lineto(x0+x,y0+y);
   Pen.Color:=Clyellow;
   Pen.Width:=5;
   Moveto(x0,y0+y);
   Lineto(2*x,y0+y);
      end ;    end;

procedure TForm1.Exit1Click(Sender: TObject);
begin
close;
end;

procedure TForm1.Perpendicular1Click(Sender: TObject);
const pi=3.14;
var angle:integer;
angle1:real;
factor:integer;
x,y:integer;
x0,y0:integer;
begin
   angle:=StrToInt(Edit1.Text);
   angle1:=pi*angle/180;
   with form1.PaintBox1.Canvas do
   begin
   Brush.Color:=clred;
   Pen.Color:=clyellow;
   pen.Width:=5;
   rectangle (1,1,700,700);
   x0:=10;
   y0:=10;
   moveto(x0,y0);
   x:=round(400*sin(angle1));
   y:=round(400*cos(angle1));
   Pen.Color:=clgreen;
   Pen.Width:=4;
   Lineto(x0+x,y0+y);
   Pen.Color:=Clyellow;
   Pen.Width:=5;
   Moveto(x0,y0+y);
   Lineto(2*x,y0+y);
   Pen.Color:=clblack;
   Pen.Style:=psdashdot;
   Pen.Width:=4;
   Moveto(x0+x,y0);
```

```
    Lineto(x0+x,2*y);
     end ;
      end;

procedure TForm1.Reflectedray1Click(Sender: TObject);
const pi=3.14;
var angle:integer;
angle1:real;
factor:integer;
x,y:integer;
x0,y0:integer;
x1,y1:integer;
begin
    angle:=StrToInt(Edit1.Text);
    angle1:=pi*angle/180;
    with form1.PaintBox1.Canvas do
    begin
    Brush.Color:=clred;
    Pen.Color:=clyellow;
    pen.Width:=5;
    rectangle (1,1,700,700);
    x0:=10;
    y0:=10;
    moveto(x0,y0);
    x:=round(400*sin(angle1));
    y:=round(400*cos(angle1));
    Pen.Color:=clgreen;
    Pen.Width:=4;
    Lineto(x0+x,y0+y);
    Pen.Color:=Clyellow;
    Pen.Width:=5;
    Moveto(x0,y0+y);
    Lineto(2*x,y0+y);
    Pen.Color:=clblack;
    Pen.Style:=psdashdot;
    Pen.Width:=4;
    Moveto(x0+x,y0);
    Lineto(x0+x,2*y);
    Pen.Color:=clwhite;
    Pen.Width:=4;
    x1:=x0+x;
    y1:=y0+y;
    Moveto(x1,y1);
    x:=round(400*sin(angle1));
    y:=round(400*cos(angle1));
    Lineto(x1+x,y1-y);
     end ;
      end;

procedure TForm1.Perpendicular2Click(Sender: TObject);
const pi=3.14;
var angle:integer;
```

```
angle1:real;
factor:integer;
x,y:integer;
x0,y0:integer;
begin
   angle:=StrToInt(Edit1.Text);
   angle1:=pi*angle/180;
   with form1.PaintBox1.Canvas do
   begin
   Brush.Color:=clred;
   Pen.Color:=clyellow;
   pen.Width:=5;
   rectangle (1,1,700,700);

   x0:=10;
   y0:=10;
   moveto(x0,y0);
   x:=round(400*sin(angle1));
   y:=round(400*cos(angle1));
   Pen.Color:=clgreen;
   Pen.Width:=4;
   Lineto(x0+x,y0+y);
   Pen.Color:=Clyellow;
   Pen.Width:=5;
   Moveto(x0,y0+y);
   Lineto(2*x,y0+y);
   Pen.Color:=clblack;
   Pen.Style:=psdashdot;
   Pen.Width:=4;
   Moveto(x0+x,y0);
   Lineto(x0+x,2*y);
    end ;
     end;

procedure TForm1.Incidentray2Click(Sender: TObject);
 const pi=3.14;
var angle:integer;
angle1:real;
factor:integer;
x,y:integer;
x0,y0:integer;
begin
   angle:=StrToInt(Edit1.Text);
   angle1:=pi*angle/180;
   with form1.PaintBox1.Canvas do
   begin
   Brush.Color:=clred;
   Pen.Color:=clyellow;
   pen.Width:=5;
   rectangle (1,1,700,700);
   x0:=10;
   y0:=10;
```

```
    moveto(x0,y0);
    x:=round(400*sin(angle1));
    y:=round(400*cos(angle1));
    Pen.Color:=clgreen;
    Pen.Width:=4;
    Lineto(x0+x,y0+y);
    Pen.Color:=Clyellow;
    Pen.Width:=5;
    Moveto(x0,y0+y);
    Lineto(2*x,y0+y);
     end ;
     end;

procedure TForm1.Reflectedray2Click(Sender: TObject);
const pi=3.14;
var angle:integer;
angle1:real;
factor:integer;
x,y:integer;
x0,y0:integer;
x1,y1:integer;
i:integer;
begin
    angle:=StrToInt(Edit1.Text);
    Factor:=StrToInt(Edit2.Text);
    angle1:=pi*angle/180;
    with form1.PaintBox1.Canvas do
    begin
    Brush.Color:=clred;
    Pen.Color:=clyellow;
    pen.Width:=5;
    rectangle (1,1,700,700);
    x0:=10;
    y0:=10;
    moveto(x0,y0);
    x:=round(400*sin(angle1));
    y:=round(400*cos(angle1));
    Pen.Color:=clgreen;
    Pen.Width:=4;
    Lineto(x0+x,y0+y);
    Pen.Color:=Clyellow;
    Pen.Width:=5;
    Moveto(x0,y0+y);
    Lineto(2*x,y0+y);
    Pen.Color:=clblack;
    Pen.Style:=psdashdot;
    Pen.Width:=4;
    Moveto(x0+x,y0);
    Lineto(x0+x,2*y);
    Pen.Color:=clwhite;
    Pen.Width:=4;
    x1:=x0+x;
```

```
    y1:=y0+y;
    for i:=1 to factor do
    begin
     Moveto(x1,y1);
    x:=round(400*sin(angle1*i/factor));
    y:=round(400*cos(angle1*i/factor));
    Lineto(x1+x,y1-y);
      end;
     end ;
        end;

procedure TForm1.E1Click(Sender: TObject);
begin
close;
end;

procedure TForm1.Incidentray3Click(Sender: TObject);
const pi=3.14;
var angle:integer;
angle1:real;
factor:integer;
x,y:integer;
x0,y0:integer;
begin
    angle:=StrToInt(Edit1.Text);
    angle1:=pi*angle/180;
    with form1.PaintBox1.Canvas do
    begin
    Brush.Color:=clred;
    Pen.Color:=clyellow;
    pen.Width:=5;
    rectangle (1,1,700,700);
    x0:=10;
    y0:=10;
    moveto(x0,y0);
    x:=round(400*sin(angle1));
    y:=round(400*cos(angle1));
    Pen.Color:=clgreen;
    Pen.Width:=4;
    Lineto(x0+x,y0+y);
    Pen.Color:=Clyellow;
    Pen.Width:=5;
    Moveto(x0,y0+y);
    Lineto(2*x,y0+y);
     end ;
   end;

procedure TForm1.Perpendicular3Click(Sender: TObject);
const pi=3.14;
var angle:integer;
angle1:real;
factor:integer;
```

```
x,y:integer;
x0,y0:integer;
begin
   angle:=StrToInt(Edit1.Text);
   angle1:=pi*angle/180;
   with form1.PaintBox1.Canvas do
   begin
   Brush.Color:=clred;
   Pen.Color:=clyellow;
   pen.Width:=5;
   rectangle (1,1,700,700);
   x0:=10;
   y0:=10;
   moveto(x0,y0);
   x:=round(400*sin(angle1));
   y:=round(400*cos(angle1));
   Pen.Color:=clgreen;
   Pen.Width:=4;
   Lineto(x0+x,y0+y);
   Pen.Color:=Clyellow;
   Pen.Width:=5;
   Moveto(x0,y0+y);
   Lineto(2*x,y0+y);
   Pen.Color:=clblack;
   Pen.Style:=psdashdot;
   Pen.Width:=4;
   Moveto(x0+x,y0);
   Lineto(x0+x,2*y);
    end ;
     end;

procedure TForm1.Refractedray1Click(Sender: TObject);
const pi=3.14;
var angle:integer;
angle1:real;
factor:integer;
x,y:integer;
x0,y0:integer;
x1,y1:integer;
n:real;
beta:real;
begin
   angle:=StrToInt(Edit1.Text);
   angle1:=pi*angle/180;
   n:=StrToFloat(Edit3.Text);
      beta:=sin(angle1)/n;
      beta:=arctan(beta/sqrt(1-sqr(beta)));
   with form1.PaintBox1.Canvas do
   begin
   Brush.Color:=clred;
   Pen.Color:=clyellow;
   pen.Width:=5;
```

```
   rectangle (1,1,700,700);
   x0:=10;
   y0:=10;
   moveto(x0,y0);
   x:=round(400*sin(angle1));
   y:=round(400*cos(angle1));
   Pen.Color:=clgreen;
   Pen.Width:=4;
   Lineto(x0+x,y0+y);
   Pen.Color:=Clyellow;
   Pen.Width:=5;
   Moveto(x0,y0+y);
   Lineto(2*x,y0+y);
   Pen.Color:=clblack;
   Pen.Style:=psdashdot;
   Pen.Width:=4;
   Moveto(x0+x,y0);
   Lineto(x0+x,2*y);
   Pen.Color:=clwhite;
   Pen.Width:=4;
   x1:=x0+x;
   y1:=y0+y;
   Moveto(x1,y1);
   x:=round(400*sin(beta));
   y:=round(400*cos(beta));
   Lineto(x1+x,y1+y);
    end ;
     end;

procedure TForm1.Snellslaw2Click(Sender: TObject);
begin
  Form2.ShowModal;
end;

procedure TForm1.Diffusionreflection2Click(Sender: TObject);
begin
Form3.ShowModal;
end;
end.
```

Лістинг3. Візуальне програмування повного внутрішнього відбивання з використанням мови програмування Object Pascal.

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ComCtrls, Buttons, StdCtrls, ExtCtrls, jpeg;
type
  TForm1 = class(TForm)
```

```
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    TabSheet3: TTabSheet;
    TabSheet4: TTabSheet;
    PaintBox1: TPaintBox;
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Edit2: TEdit;
    Button1: TButton;
    BitBtn1: TBitBtn;
    PaintBox2: TPaintBox;
    Label3: TLabel;
    Edit3: TEdit;
    Label4: TLabel;
    Edit4: TEdit;
    Button2: TButton;
    BitBtn2: TBitBtn;
    PaintBox3: TPaintBox;
    Label5: TLabel;
    Edit5: TEdit;
    Label6: TLabel;
    Edit6: TEdit;
    Button3: TButton;
    BitBtn3: TBitBtn;
    Image1: TImage;
    Image2: TImage;
    Image3: TImage;
    Image6: TImage;
    Image7: TImage;
    Image8: TImage;
    Image4: TImage;
    Image5: TImage;
    procedure BitBtn1Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}

procedure TForm1.BitBtn1Click(Sender: TObject);
begin
```

```
close;
end;

procedure TForm1.Button1Click(Sender: TObject);
const pi=3.14;  //discription constants
var angle:integer;//discription variables
angle1:real;
x,y:integer;
x0,y0:integer;
x1,y1:integer;
x2,y2:integer;
n:real;
beta:real;
begin
   angle:=StrToInt(Edit1.Text);  //input data
   angle1:=pi*angle/180;
   n:=StrToFloat(Edit2.Text);
     beta:=sin(angle1)/n;
     beta:=arctan(beta/sqrt(1-sqr(beta)));
   with form1.PaintBox1.Canvas do//graphic connection class
   begin
   Brush.Color:=clred;
   Pen.Color:=clyellow;
   pen.Width:=5;
   rectangle (1,1,700,600);
   x0:=10;
   y0:=10;
   moveto(x0,y0);//output beams
   x:=round(300*sin(angle1));
   y:=round(300*cos(angle1));
   Pen.Color:=clgreen;
   Pen.Width:=4;
   Lineto(x0+x,y0+y);
    brush.Color:=clblue;
   pen.Color:=clyellow;
   pen.Width:=4;
   rectangle(1,y0+y, 700,y0+y+200);
   Pen.Color:=Clyellow;
   Pen.Width:=4;
   Moveto(x0,y0+y);
   Lineto(700,y0+y);
   Pen.Color:=clblack;
   Pen.Style:=psdashdot;
   Pen.Width:=2;
   Moveto(x0+x,y0);
   Lineto(x0+x,2*y);
   Pen.Color:=clwhite;
   Pen.Width:=4;
   x1:=x0+x;
   y1:=y0+y;
   Moveto(x1,y1);
   x:=round(200*sin(beta)/cos(beta));
```

```
    y:=200;
    Lineto(x1+x,y1+y);
    x2:=x1+x;
    y2:=y1+y;
    Pen.Color:=clmaroon;
    Pen.Width:=4;
    Pen.Style:=psdash;
    Moveto(x2,y2);
     x:=round(200*sin(angle1));
    y:=round(200*cos(angle1));
    Lineto(x2+x,y2+y);
     end ;
     end;
procedure TForm1.BitBtn2Click(Sender: TObject);
begin
close;
end;

procedure TForm1.Button2Click(Sender: TObject);
const pi=3.14;
var angle:integer;
angle1:real;
x,y:integer;
x0,y0:integer;
x1,y1:integer;
x2,y2:integer;
x3,y3:integer;
x4,y4:integer;
n:real;
begin
     n:=StrToFloat(Edit3.Text);
    angle:=StrToInt(Edit4.Text);
    angle1:=pi*angle/180;
    with form1.PaintBox2.Canvas do
    begin
    Brush.Color:=clred;
    Pen.Color:=clyellow;
    pen.Width:=7;
    pen.Style:=psdashdot;
    rectangle (1,1,785,649);
    brush.Color:=clblue;
    pen.Color:=clsilver;
    pen.Width:=4;
    rectangle(1,100, 785,500);
    x0:=1;
    y0:=100;
    moveto(x0,y0);
    x:=round(400*sin(angle1)/cos(angle1));
    y:=400;
      Pen.Color:=clgreen;
    Pen.Width:=4;
    Lineto(x0+x,y0+y);
```

```
    x1:=x0+x;
    y1:=y0+y;
    moveto(x1,y1);
    x2:=x1+x;
    y2:=y1-400;
    Lineto(x2,y2);
    moveto(x2,y2);
    x3:=x2+x;
    y3:=y2+y;
    Lineto(x3,y3);
    moveto(x3,y3);
    x4:=x3+x;
    y4:=y3-y;
    Lineto(x4,y4);
     end ;
      end;

procedure TForm1.BitBtn3Click(Sender: TObject);
begin
close;
end;
procedure TForm1.Button3Click(Sender: TObject);
const pi=3.14;
var angle:integer;
angle1:real;
x,y:integer;
x0,y0:integer;
x1,y1:integer;
x2,y2:integer;
x3,y3:integer;
x4,y4:integer;
x5,y5, x6,y6,x7,y7,x8,y8:integer;
n:real;
begin
     n:=StrToFloat(Edit5.Text);
    angle:=StrToInt(Edit6.Text);
    angle1:=pi*angle/180;
    with form1.PaintBox3.Canvas do
    begin
    Brush.Color:=clred;
    Pen.Color:=clyellow;
    pen.Width:=7;
    pen.Style:=psdashdot;
    rectangle (1,1,785,657);
    brush.Color:=clblue;
    pen.Color:=clsilver;
    pen.Width:=4;
    rectangle(1,100, 785,200);
    x0:=1;
    y0:=100;
    moveto(x0,y0);
    x:=round(100*sin(angle1)/cos(angle1));
```

```
   y:=100;
     Pen.Color:=clgreen;
   Pen.Width:=4;
   Lineto(x0+x,y0+y);
   x1:=x0+x;
   y1:=y0+y;
   moveto(x1,y1);
   x2:=x1+x;
   y2:=y1-100;
   Lineto(x2,y2);
   moveto(x2,y2);
   x3:=x2+x;
   y3:=y2+y;
   Lineto(x3,y3);
   moveto(x3,y3);
   x4:=x3+x;
   y4:=y3-y;
   Lineto(x4,y4);
    end ;
      end;
     end.
```

Лістинг4. Визначення спектральної залежності пропускання з використанням методів апроксимації та їх моделювання.

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, Buttons, Series, TeEngine, TeeProcs, Chart;
type
  TForm1 = class(TForm)
    Button1: TButton;
    GroupBox1: TGroupBox;
    Edit1: TEdit;
    Label1: TLabel;
    Panel1: TPanel;
    Button2: TButton;
    Memo1: TMemo;
    GroupBox2: TGroupBox;
    Memo2: TMemo;
    BitBtn1: TBitBtn;
    Button3: TButton;
    BitBtn2: TBitBtn;
    Chart1: TChart;
    Series1: TLineSeries;
    Series2: TPointSeries;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
```

```
    procedure BitBtn2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
  Const n=10;
 z: array [1..n] of real=(550,575,600,625,650,675,700,725,750,775); //experimental data
 y: array [1..n] of real=(0.23,0.9,2.5,5.7,11.6,18.1,27,31.5,44.6,47.5);
var
  Form1: TForm1;
  g,m:real;
implementation
uses Unit2;
{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
begin
form2.showmodal;
end;

procedure TForm1.Button2Click(Sender: TObject);
var i:integer;
    f1,f2,f3,f4,lng:real;
begin
f1:=0;
    f2:=0;
    f3:=0;
    f4:=0 ;
    Memo1.Lines.clear;
    for i:=1 to n do
          begin
          f1:=f1 + z[i];
           f2:=f2+ z[i]*z[i];
           f3:=f3+ ln(y[i]);
           f4:=f4+ z[i]*ln(y[i]);
           end;
           m:= (n*f4-f3*f1)/(n*f2-f1*f1);
           lng:= (f4-m*f2)/f1;
          g:=exp(lng);     // calculation m,g
Memo1.Lines.Add('g =  ' + FloatToStrF(g,fffixed,15,10));
Memo1.Lines.Add('m =  '+FloatToStrF(m,fffixed,15,10));//output m,g
    end;

procedure TForm1.BitBtn1Click(Sender: TObject);
Var i:integer;
x,yy:real;
begin
Memo2.Lines.clear;
x:=500;
For i:=1 to 18 do
begin
```

```
x:=x+15;
Memo2.Lines.Add('z = '+FloatToStrF(x,fffixed,5,3)+' y = '+
FloatToStrF(g*exp(x*m),fffixed,15,10)); // calculation y(z)
end;
 Chart1.SeriesList[1].Clear;
 for i:=1 to 10 do
  Chart1.SeriesList[1].AddXY(z[i],y[i],",clred);
 Chart1.SeriesList[0].Clear;
 x:=500;
For i:=1 to 18 do
begin
 yy:=g*exp(m*x);
 Chart1.SeriesList[0].AddXY(x,yy,",clgreen);
 x:=x+15;
  end;
 end;
procedure TForm1.BitBtn2Click(Sender: TObject);
begin
close;
end;
end.
```

Лістинг5. Компонента Image середовища програмування Delphi і побудова зображень за допомогою лінз.

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, jpeg, ExtCtrls, StdCtrls, Buttons;
type
  TForm1 = class(TForm)
    Image1: TImage;
    Image3: TImage;
    Image4: TImage;
    Image5: TImage;
    BitBtn1: TBitBtn;
    BitBtn2: TBitBtn;
    BitBtn3: TBitBtn;
    BitBtn4: TBitBtn;
    procedure BitBtn1Click(Sender: TObject);
    procedure BitBtn2Click(Sender: TObject);
    procedure BitBtn3Click(Sender: TObject);
    procedure BitBtn4Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
```

```
  Form1: TForm1;
implementation
{$R *.dfm}

procedure TForm1.BitBtn1Click(Sender: TObject);
begin
Image1.Visible:=False;
end;

procedure TForm1.BitBtn2Click(Sender: TObject);
begin
Image1.Visible:=True;
end;

procedure TForm1.BitBtn3Click(Sender: TObject);
begin
Image3.Width:=300;
Image3.Enabled:=False;
end;

procedure TForm1.BitBtn4Click(Sender: TObject);
begin
Image3.Left:=650;
end;
end.
```

Лістинг6. Визначення оптичних характеристик збірної і розсіюючої лінз та побудова графіків з використанням компоненти Chart середовища програмування Delphi.

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, TeEngine, Series, StdCtrls, ExtCtrls, TeeProcs, Chart, Buttons;
type
  TForm1 = class(TForm)
    Chart1: TChart;
    Chart2: TChart;
    Button1: TButton;
    Edit1: TEdit;
    Edit2: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Series1: TPointSeries;
    Series2: TPointSeries;
    Series3: TPointSeries;
    Button2: TButton;
    Edit3: TEdit;
```

```
    Label4: TLabel;
    Series4: TPointSeries;
    Series5: TPointSeries;
    Series6: TPointSeries;
    Series7: TPointSeries;
    Series8: TPointSeries;
    BitBtn1: TBitBtn;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
var n,R1,R2,d:real;  i:integer;
function force(n:real):real;
begin
Result:=(n-1)*(1/R1-1/R2)+d*sqr(n-1)/(n*R1*R2);
end;
begin
R1:=StrToFloat(Edit1.Text);
R2:=StrToFloat(Edit2.Text);
 Chart1.Serieslist[0].Clear;
  d:=2; n:=1;
  for  i:=1 to 20 do
  begin
  n:=n+i*0.01;
     Chart1.Serieslist[0].AddXY(n,force(n),'',clred);
  end;
  Chart1.Serieslist[1].Clear;
  d:=4; n:=1;
  for  i:=1 to 20 do
  begin
  n:=n+i*0.01;
     Chart1.Serieslist[1].AddXY(n,force(n),'',clgreen);
  end;
  Chart1.Serieslist[2].Clear;
  d:=6; n:=1;
  for  i:=1 to 20 do
  begin
  n:=n+i*0.01;
     Chart1.Serieslist[2].AddXY(n,force(n),'',cllime);
  end;
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
var delta,L,ml:real;  i:integer;
function length(L:real):real;
begin
Result:=(sqr(L-delta)-sqr(ml))/(4*(L-delta));
end;
begin
ml:=StrToFloat(Edit3.Text);
 Chart2.Serieslist[0].Clear;
  delta:=0; L:=1;
  for  i:=1 to 20 do
  begin
  L:=L+i*0.01;
    Chart2.Serieslist[0].AddXY(L,length(L),'',clred);
  end;

  Chart2.Serieslist[1].Clear;
  delta:=0.2; L:=1;
  for  i:=1 to 20 do
  begin
  L:=L+i*0.01;
    Chart2.Serieslist[1].AddXY(L,length(L),'',clgreen);
  end;
Chart2.Serieslist[2].Clear;
  delta:=0.4; L:=1;
  for  i:=1 to 20 do
  begin
  L:=L+i*0.01;
    Chart2.Serieslist[2].AddXY(L,length(L),'',clyellow);
  end;
 Chart2.Serieslist[3].Clear;
  delta:=0.6; L:=1;
  for  i:=1 to 20 do
  begin
  L:=L+i*0.01;
    Chart2.Serieslist[3].AddXY(L,length(L),'',clblue);
  end;
  Chart2.Serieslist[4].Clear;
  delta:=0.8; L:=1;
  for  i:=1 to 20 do
  begin
  L:=L+i*0.01;
    Chart2.Serieslist[4].AddXY(L,length(L),'',clwhite);
  end;
  end;
procedure TForm1.BitBtn1Click(Sender: TObject);
begin
close;
end;
end.
```

Лістинг7. Візуальне моделювання явища інтерференції в середовищі Delphi (C++)».

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Series, TeEngine, BubbleCh, ExtCtrls, TeeProcs, Chart,
  jpeg, ComCtrls, Buttons, ArrowCha, TeeShape;
type
  TForm1 = class(TForm)
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    TabSheet3: TTabSheet;
    Image1: TImage;
    Label1: TLabel;
    Image2: TImage;
    Label2: TLabel;
    Chart1: TChart;
    Image3: TImage;
    Image4: TImage;
    Label3: TLabel;
    Chart2: TChart;
    Chart3: TChart;
    GroupBox1: TGroupBox;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Series1: TPointSeries;
    BitBtn1: TBitBtn;
    Series3: TBarSeries;
    Series2: TAreaSeries;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
```

```
implementation
{$R *.dfm}

procedure TForm1.Button1Click(Sender: TObject);
var m,i:integer;
r,d0,dm,lambda:real;
begin
m:=StrToInt(Edit1.Text);
d0:=StrToFloat(Edit2.Text);
dm:=StrToFloat(Edit3.Text);
Chart1.SeriesList[0].Clear;
 lambda:=0.4;
 for i:=1 to 15 do
 begin
 r:=(sqr(dm)-sqr(d0))/(4*m*lambda) ;
 Chart1.SeriesList[0].AddXY(lambda, r,",cllime);
 lambda:=lambda+i*0.005;
 end;
end;

procedure TForm1.Button2Click(Sender: TObject);
var m,i:integer;
r,d0,dm,lambda:real;
begin
m:=StrToInt(Edit1.Text);
d0:=StrToFloat(Edit2.Text);
dm:=StrToFloat(Edit3.Text);
Chart2.SeriesList[0].Clear;
 lambda:=0.4;
 for i:=1 to 15 do
 begin
 r:=(sqr(dm)-sqr(d0))/(4*m*lambda) ;
 Chart2.SeriesList[0].AddXY(lambda, r,",clred);
 lambda:=lambda+i*0.005;
 end;
end;

procedure TForm1.Button3Click(Sender: TObject);
var m,i:integer;
r,d0,dm,lambda:real;
begin
m:=StrToInt(Edit1.Text);
d0:=StrToFloat(Edit2.Text);
dm:=StrToFloat(Edit3.Text);
Chart3.SeriesList[0].Clear;
 lambda:=0.4;
 for i:=1 to 15 do
 begin
 r:=(sqr(dm)-sqr(d0))/(4*m*lambda) ;
 Chart3.SeriesList[0].AddXY(lambda, r,",clblue);
 lambda:=lambda+i*0.005;
   end;
```

```
   end;
procedure TForm1.BitBtn1Click(Sender: TObject);
begin
close;
end;
end.
```

Лістинг8. Розробка тестових завдань з оптики в середовищі Delphi.

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, StdCtrls, Buttons, jpeg, ExtCtrls;

type
  TForm1 = class(TForm)
    Image1: TImage;
    Image2: TImage;
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Memo1: TMemo;
    BitBtn1: TBitBtn;
    Button1: TButton;
    MainMenu1: TMainMenu;
    LevelOne1: TMenuItem;
    LevelTwo1: TMenuItem;
    LevelThree1: TMenuItem;
    Case11: TMenuItem;
    Case21: TMenuItem;
    Case31: TMenuItem;
    Case41: TMenuItem;
    Case51: TMenuItem;
    Case61: TMenuItem;
    Exit1: TMenuItem;
    Case12: TMenuItem;
    Ca1: TMenuItem;
    Case32: TMenuItem;
    Case42: TMenuItem;
    Case52: TMenuItem;
    Case62: TMenuItem;
    Exit2: TMenuItem;
    jgjfg1: TMenuItem;
    Case22: TMenuItem;
    Case33: TMenuItem;
    Case43: TMenuItem;
    Case53: TMenuItem;
    Case63: TMenuItem;
    Exit3: TMenuItem;
```

```
    Button2: TButton;
    procedure Button2Click(Sender: TObject);
    procedure Case11Click(Sender: TObject);
    procedure Exit1Click(Sender: TObject);
    procedure Case21Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;
  type fam=record
  level1, level2, level3:integer
  end;
var
L:string[25];
fm:fam;
  Form1: TForm1;
implementation
{$R *.dfm}


procedure TForm1.Button2Click(Sender: TObject);
Var S1,S2,S3,S4,S5:string;
begin
Memo1.Clear;
S1:=InputBox('What is the nature of light waves?','a)transverse b)longitudinal','');
If (S1='a') or (S1='A') then  Memo1.text:='true' else   Memo1.text:='false';
  S2:=inputBox('Optical lens power measured in...','a)lumens b)diopter','');
   If (S2='b') or (S2='B')  then
   Memo1.text:='true' else   Memo1.text:='false';
   S3:=InputBox('What phenomenon proves transverse light waves?','a)reflection b)polarization','');
If (S3='a') or (S3='A') then   Memo1.text:='false' else  Memo1.text:='true';
   S4:=InputBox('Some subject have white color because?','a)reflect all colors of white light;
b)scatter white light','');
If (S4='a') or (S4='A') then  Memo1.text:='true' else  Memo1.text:='false';
  S5:=InputBox('How will the length of the yellow light emission when passing from air into
water?','a)increases;b)decreases','');
If (S4='a') or (S4='A') then Memo1.text:='false' else  Memo1.text:='true';
     end;

procedure TForm1.Case11Click(Sender: TObject);
      Var S:string;
begin
Memo1.Clear;
S:=InputBox('What is the nature of light waves?','a)transverse b)longitudinal','');
If (S='a') or (S='A') then   fm.level1:=1 else  fm.level1:=0
end;

procedure TForm1.Exit1Click(Sender: TObject);
begin
```

```
   L:=edit1.Text;
   Memo1.Lines.Add(L+' =  ' +inttostr(fm.level1))
end;

procedure TForm1.Case21Click(Sender: TObject);
Var S : string;
begin
Memo1.Clear;
 S:=inputBox('Optical lens power measured in...','a)lumens b)diopter','');
   If (S='b') or (S='B')  then    fm.level1:= fm.level1+1 else  fm.level1:= fm.level1+0
end;
procedure TForm1.BitBtn1Click(Sender: TObject);
begin
close;
end;
end.
```