**МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ
МІЖНАРОДНИЙ ЕКОНОМІКО-ГУМАНІТАРНИЙ
УНІВЕРСИТЕТ ІМЕНІ АКАДЕМІКА
СТЕПАНА ДЕМ'ЯНЧУКА**

О.М. ЧЕРКУН

Розробка і дослідження графічного середовища LOTOS для створення динамічних геометричних побудов

**Науковий керівник: Ю. Г. Лотюк,
кандидат педагогічних наук, доцент
Науковий консультант: Р.М.Літнарович,
кандидат технічних наук, доцент**

Рівне, 2012

УДК 004.92

ББК В6

Черкун О.М. Розробка і дослідження графічного середовища LOTOS для створення динамічних геометричних побудов. Монографія. Науковий керівник Ю.Г.Лотюк. Науковий консультант Р.М.Літнарович. МЕГУ, Рівне, 2012. – 103с.
Cherkun O.M.. Development and research of graphic environment of LOTOS is for creation of dynamic geometrical constructions. Monograph. Scientific leader Yu.G.Lotyuk. Scientific consultant R.M.Litnarovich . IEGU, Rivne, 2012. – 103p.

Рецензенти: В.Г.Бурачек, доктор технічних наук, професор
Є.С. Парняков, доктор технічних наук, професор
В.О. Боровий, доктор технічних наук, професор

Під редакцією Р.М.Літнаровича, кандидата технічних наук, доцента

В монографії розроблений, вдосконалений і досліджений графічний редактор LOTOS для школярів, геометрів і конструкторів, який дає змогу легко створювати прості геометричні фігури

Ключові слова: графічне середовище, створення, графічні побудови, геометричні побудови.

В монографии разработанный, усовершенствованный и исследован графический редактор LOTOS для школьников, геометров и конструкторов, который дает возможность легко создавать простые геометрические фигуры

Ключевые слова: графическая среда, создание, графические построения, геометрические построения.

In a monograph developed, and improved graphics editor LOTOS is investigational for schoolboys, geometers and designers, which enables easily to create simple geometrical figures

Keywords: graphic environment, creation, graphic constructions, geometrical constructions.

Черкун Оксана Мирославівна,
магістрант інформаційних технологій

ЗМІСТ

ВСТУП

Комп'ютерна графіка з'явилась достатньо давно вже у 1960-х роках існували повноцінні програми роботи з графікою. Поняття «комп'ютерна графіка» об'єднує всі види робот зі статичними зображеннями. У теперішній час, завдяки грандіозному розвитку комп'ютерної техніки, деякі сторони нашого життя неможливо уявити собі без застосування комп'ютерних технологій, у тому числі без комп'ютерної графіки. Це, насамперед: усі види поліграфічних процесів; майже вся рекламна індустрія; телебачення; моделювання нових видів одягу; проектно-конструкторські розробки і т. і.

По своїй структурі зображення можуть бути растровими та векторними. Наприклад, сканер під час сканування розбиває зображення на безліч дрібних елементів (пікселів) і формує з них растрову картинку. Колір кожного пікселя записується у пам'ять комп'ютера за допомогою певної кількості бітів. Біт - мінімальна одиниця пам'яті комп'ютера, яка може зберігати значення або 0, або 1. Піксель це найменший елемент, растрового зображення. Чим більша кількість пікселів у зображенні, тим краще його вигляд на екрані і при друці. Але цей засіб подання зображення не підходить для тих випадків, коли виникає необхідність у масштабуванні зображення у великих межах. Цього браку позбавлені векторні зображення, у яких розмір будь-якого елемента може бути змінений аж «до нескінченності Векторна графіка (також геометричне моделювання або об'єктно-орієнтована графіка) створення зображення з сукупності геометричних примітивів (точок, ліній, кривих, полігонів), тобто об'єктів які можна описати математичним рівнянням. Векторна графіка створюється за допомогою спеціальних програмних засобів типу CorelDRAW, Adobe Illustrator. Також такий формат зображення використовується в усіх програмах САПР (Системи Автоматичного Проектування) (P-CAD, Auto-CAD і тому подібне). Векторна графіка не залежить від продуктивності апаратних засобів, яка дозволяє легко змінювати розміри статичних зображень

без втрати загальної кількості елементів зображення, ясності і чіткості їхніх меж при виведенні на екран монітору або при друці.

Графічний редактор — прикладна програма (або пакет програм), що дозволяє створювати і редагувати зображення за допомогою комп'ютера. Відповідно до типів зображення графічні редактори бувають векторними та растровими, а також існують гібридні та тривимірні графічні редактори.

Векторні графічні редактори дозволяють користувачеві створювати і редагувати векторні зображення безпосередньо на екрані комп'ютера, а також зберігати ці зображення у різних векторних форматах.

Растровий графічний редактор — спеціалізована програма для створення і обробки растрових зображень. Растрові графічні редактори дозволяють користувачеві створювати і редагувати зображення на екрані комп'ютера (серед звичних інструментів — декілька типів ліній, стирання, копіювання об'єктів, додавання тексту, заповнення кольору фону…), а також зберігати їх в різних растрових форматах. На противагу векторним редакторам, растрові використовують для утворення зображень матрицю крапок (bitmap). При цьому, більшість сучасних растрових редакторів містять векторні інструменти редагування як допоміжні.

В монографії розроблений, вдосконалений і досліджений графічний редактор LOTOS для школярів, геометрів і конструкторів, який дає змогу легко створювати прості геометричні фігури.

1. ПРАКТИЧНА РЕАЛІЗАЦІЯ ПРОЕКТУ

1.1. Класи. Ієрархія класів застосованих у програмі

Об’єкт – це ключ до розуміння об'єктно́ - орієнтованих технологій. Клас – це і є те саме креслення, із якого створюються певні об’єкти. Кожен клас може мати довільну кількість нащадків, що дає змогу створювати і ієрархічні дерева успадкування. Нащадок може перекривати деякі методи предка, і тоді метод з одним іменем для різних класів виконуватиметься по різному.

Конструктор – метод класу, який має таке ж ім’я як клас, створює екземпляр класу і заповнює його поля конкретними значеннями. [9, с.112]

Механізм успадкування для класів програми LOTOS проілюстровано на схемі у Додатку А.

Клас Pset (Точка) вибрано за батьківський оскільки він містить основні властивості графічних класів: координати точки, назву точки, шрифт колір лінії та штриховки точки, вказівник на компонент на якому буде будуватись зображення, методи для задавання координат і назви точки, для малювання пустої і заповненої точки, малювання всіє фігури, які будуть успадковуватися іншими класами.

Коло визначається його центром (точкою), та його радіусом(цілим числом). Тому клас Circle (Коло) як нащадок Pset (Точка) буде доповнений полем цілого типу R, та деякими методами які задають параметри кола, а також колір та стиль ліній, і режим та колір штриховки. Лінія визначається двома точками, тому клас Line(Лінія) містить і координати другої точки coord B, а також колір і стиль лінії. Відрізок також визначається двома точками, тому клас Length (Відрізок) є нащадком класу Line, тільки доповнений змінною яка містить назву другого кінця відрізка AnsiString NameB, Клас Ellips (Еліпс) є нащадком класу Line, оскільки для нього потрібні координати трьох точок і назви центру еліпса, тому він доповнений змінною coord C, та кольором і стилем штриховки. Клас Rectangle

(Прямокутник) є нащадком класу Length, так як для того щоб намалювати прямокутник потрібно всього дві точки. Цей клас доповнений ще двома текстовими змінними, так як у прямокутника є чотири точки з назвами AnsiString NameC, AnsiString NameD. Так як, для того щоб намалювати трикутник потрібно 3 точки, які також можуть мати назви, то клас Triangle(Трикутник) є похідним від класу Length, і має додаткову змінну яка містить координати третьої точки coord C, та її назву AnsiString NameC, а також параметри штриховки. Класи Arc_3 та Arc_Center (дуга) є нащадками класу Triangle, він доповнений новими змінними такими як цент та радіус дуги, також перевизначені методи малювання фігури. Клас Rhombus(Ромб) також є нащадком класу Triangle, та доповнений змінною яка містить координати четвертої точки coord D, та назву точки AnsiString NameD. Клас Tetragon (Чотирикутник) не містить нових полів, відмінних від полів класу Rhombus. Всі класи мають різні типи конструкторів. Для всіх класів один чи кілька разів перенавантажений оператор порівняння '='.

В кожному класі перевизначені такі функції : void Draw (double zoom) – для рисування об'єкту, void DrawName (double zoom) – рисує імена вершин об'єкту над вершинами, функція void SetNameFigure (AnsiString N1,AnsiString N2,AnsiString N3,AnsiString N4) надає імена вершинам фігур. bool Select (int X, int Y) повертає істину якщо точка з координатами (X, Y) знаходиться на фігурі, а функції bool SelectRect (TRect rect) повертає істину тільки тоді коли хоча б одна з її вершин знаходяться всередині прямокутника rect. Така функція як void Move(int tX, int tY, double zoom) переміщує фігуру на tX по осі абсцис, та на tY по осі ординат. void AroundPsets(int *x,int *y) надає значення точці (x, y) таке яке має точка яка знаходиться поруч, це дозволяє точно та легко нарисувати точки чи вершини фігури з однаковими координатами. А от функція void Calk_Par() переобчислює, якщо в об'єкта передбачені інші компоненти, такі як радіуси чи висоти або бісектриси та інші, значення цих компонентів, це потрібно робити коли користувач змінює положення якоїсь однієї вершини. А от функції void

WritePar(TFileStream* f) та void ReadPar(TFileStream* f) записують у файл та зчитують з нього дані-члени класу.

Отже згадані вище класи є досить гнучкими на зручними в роботі з ними, оскільки мають широкий вибір конструкторів та перенавантажених операторів порівняння, а також широкий набір функцій які дозволяють керувати об'єктом в залежності від його типу.

1.1.1. Батьківський клас Pset (точка)

Клас Pset містить поля: вказівник на об'єкт TImage, координати точки, та її назву, а також змінну типу TFont, яка зберігає вигляд шрифту та TPsetMode PsetMode і TLineMode LineMode, які зберігають налаштування точки та лінії, тобто колір та режим точки, товщину, колір та стиль лінії.

В класі створено ряд методів які надають значення даним класу. Так методи void SetA(int a1,int b1) та void SetNamePset(AnsiString N) що успадковується призначений для задавання координат точки. Методи void SetFont(TFont *t), void ChangePsetMode(TColor bcp,bool ps), void ChangeLineMode(TColor PenColor,int PenWidth,TPenStyle PenStyle), успадковуються повністю всіма нащадками і задають такі характеристики як шрифт, колір та стиль точки, а також колір, ширину та стиль лінії.

Методи void Draw (double zoom), void DrawName(double zoom),void DrawSelect(double zoom) перевизначаються в кожному класі а їх значення було наведено вище. А от функції – члени void DrawFPset(int x,int y,double zoom), void DrawEPset(int x,int y,double zoom), void Drag (double zoom), void DrawOll(double zoom) успадковуються класами нащадків і призначені для рисування. Так функції DrawFPset та DrawEPset рисують точку, а Drag і DrawOll задають режим в якому рисувати фігуру.

За допомогою функцій coord GetA(), AnsiString GetName() та int GetBrushColor() можна отримати значення координат точки, її ім'я та колір зафарбовування.

У класі існують функції - члени для запису і читання з та у файл void WriteToFile(TFileStream* f), void ReadFromFile(TFileStream* f), які використовують методи void WritePar(TFileStream* f), virtual void ReadPar(TFileStream* f), котрі уже і перевизначаються у кожному класі-нащадку. Ці функції в якості параметра приймають вказівник на потік TFileStream* f та записують до нього координати вершин, їх назви, кольори та стилі ліній, та їх товщини, режими штриховки та ін.

Метод Draw() реалізований за допомогою функції DrawEPset, яка в свою чергу зафарбовує піксель з координатами точки в потрібний колір, та малює навколо нього коло з радіусом 5 пікселів. В результаті цього отримуємо зображення наведене на Рис.2.1.

Рис. 1.1. Об'єкт Pset (Точка)

Для того щоб можна було виділити точку, потрібно перевірити чи координати «миші» потрапляють до її меж або чи потрапляє точка до заданої користувачем прямокутної області виділення в класі реалізовані функції virtual bool Select (int X,int Y) та virtual bool SelectRect (TRect rect), ці в кожному класі – нащадку мають різну реалізацію.

Для даного класу визначено кілька конструкторів, що дозволяє створювати об'єкт цього класу у різних ситуаціях. Також перенавантажений оператор присвоєння `=`.

1.1.2. Клас Circle (Коло)

Клас *Circle* є нащадком класу Pset. Він успадковує його поля та методи, а також має власні поля та методи. Клас Circle має додаткове поле цілого типу R, яке є радіусом кола.

Цей клас також має кілька конструкторів, та перенавантажених операторів присвоєння. Circle має два перевизначені методи: virtual void Draw() та virtual void DrawName() , які малюють фігуру та її назву. Вони перевизначені так як коло малюється інакше від точки, тому і методи малювання повинні бути іншими. Також даний клас має кілька нових методів: void SetCircle(int a1, int b1,int r),void SetR(int Xr,int Yr), які задають коло та радіус кола.

Метод Draw(), який відповідає за те як буде створюватись дана фігура, в даному випадку коло, реалізований за допомогою методу Ellipse(xl,yl,x2,y2) властивості об'єкту типу TImage Canvas, де xl,yl,x2,y2 координати прямокутника чи квадрата вселені якого вписано еліпс чи коло; а також за допомогою успадкованого методу DrawFPset, який малює точку в центрі кола. Так як для того щоб намалювати коло досить тільки координати точки та радіус, а для методу Ellipse(xl,yl,x2,y2) потрібні координати двох точок, то у функцію Ellipse задається так: Canvas->Ellipse(A.x-R,A.y-R,A.x+R,A.y+R), де A.x, A.y – координати центра кола, а R – радіус кола. В результаті та екрані рисується коло з точкою в центрі і радіусом заданим користувачем (Рис. 2.2).

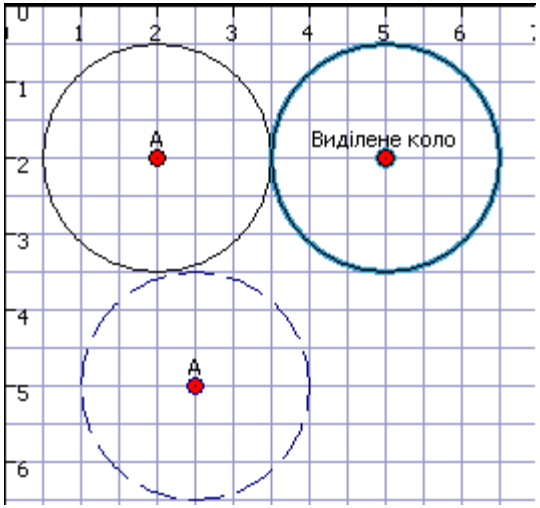


Рис.1.2. Коло. Масштаб: 1:1

1.1.2.1. Радіус, діаметр та хорда кола, їх запис у файл.

Задача знаходження координат радіуса, діаметра та хорди зводить до знаходження точок перетину кола та прямої. У випадку з діаметром та радіусом пряма проходить через центр кола.

Отже дано коло (задане координатами його центру та радіусом) і пряма (своїм рівнянням). Потрібно знайти точки їх перетину.

Замість формального розв'язування системи двох рівнянь підійдемо до задачі геометрично (до того ж, за рахунок цього ми отримаємо більш точне рішення з точки зору числової стабільності).

Допустимо, що центр кола знаходиться на початку координат. Тобто маємо коло з центром $(0,0)$ радіуса r і пряму з рівнянням $Ax + Bx + C = 0$.

Спочатку знайдемо найближчу до центру точку прямої - точку з деякими координатами (x_0, y_0). По-перше, ця точка повинна знаходитися на такій відстані від початку координат:

$$\frac{|C|}{\sqrt{A^2 + B^2}}$$

По-друге, оскільки вектор (A, B) перпендикулярний до прямої, то координати цієї точки мають бути пропорційні координатам цього вектора. Враховуючи, що відстань від початку координат до шуканої точки нам відомо, нам потрібно просто нормувати вектор (A, B) до цієї довжини, і ми отримуємо:

$$x_0 = -\frac{AC}{A^2 + B^2},$$

$$y_0 = -\frac{BC}{A^2 + B^2}$$

Дійсно, якщо відстань від (x_0, y_0) до початку координат більше радіусу, то відповідь - нуль крапок. Якщо ця відстань дорівнює радіусу, то відповіддю буде одна точка - (x_0, y_0).А ось у іншому випадку точок буде дві, і їх координати ще потрібно знайти.

Отже, ми знаємо, що точка (x_0, y_0) лежить усередині кола. Шукані точки (a_x, a_y) та (b_x, b_y), окрім того що повинні належати прямій, повинні лежати на одній і тій же відстані d від точки (x_0, y_0), причому цю відстань легко знайти:

$$d = \sqrt{r^2 - \frac{C^2}{A^2 + B^2}}$$

Відмітимо, що вектор $(-B, A)$ колінеарний до прямої, а тому шукані точки (a_x, a_y) і (b_x, b_y) можна отримати, додавши до точки (x_0, y_0) вектор $(-B, A)$, нормований до довжини d (ми отримаємо одну шукану точку), і віднявши цей же вектор (отримаємо другу шукану точку). Остаточне рішення таке [12]:

$$mult = \sqrt{\frac{d^2}{A^2 + B^1}},$$

$$a_x = x_0 + B \cdot mult,$$

$$a_y = y_0 - A \cdot mult,$$

$$b_x = x_0 - B \cdot mult,$$

$$b_y = y_0 + A \cdot mult.$$

Таким чином реалізувавши вищенаведені формули у функціях класу Circle void FindCrossPointRad(int X,int Y,int *crX,int *crY) та void FindCrossPointKhord (int r,int a,int b,int c,khord *Q) користувач може нарисувати хорду та ін. (Рис.2.3).

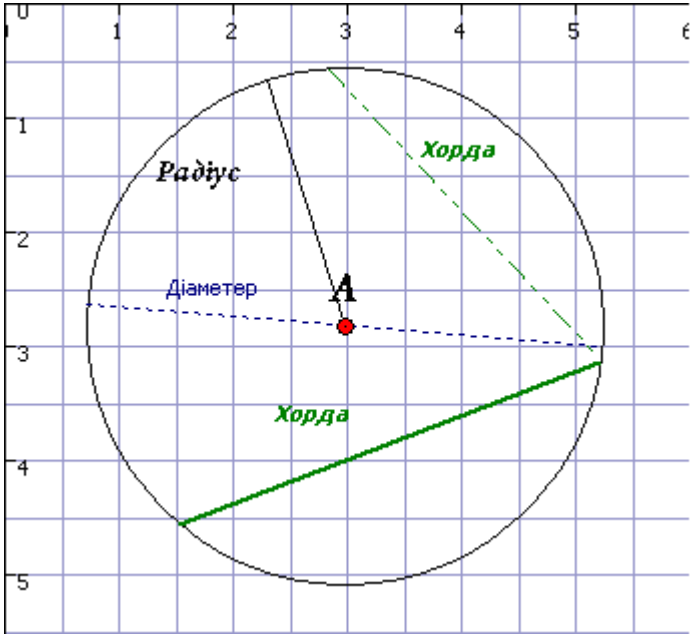


Рис.1.2. Коло з хордами, радіусом та діаметром

Кожен радіус або діаметр чи хорда можуть мати свій стиль (Рис.1.3), та можуть бути нарисовані чи виділені через контекстне меню.

Оскільки коло може мати будь яку кількість радіусів, діаметрів та хорд, то всі вони поміщені у контейнер стандартної бібліотеки шаблонів vector. При рисуванні та запису (читання) у (з) файл використовуються ітератори.

1.1.3. Клас TexT (Текст)

Клас TexT є нащадком від класу Pset, оскільки також характеризується парою координат та текстом. В основному цей клас відрізняється від батьківського способом рисування, та як Pset рисує точку у координатах і назву над нею, а TexT просто у відповідних

координатах виводить текст. Тому для цього класу перевизначені методи для рисування void Draw(double zoom), virtual void DrawName(double zoom), void DrawOll(double zoom) та void Drag(double zoom), а також методи для виділення bool Select(int,int),virtual bool SelectRect(TRect rect). Таким чином користувач може створювати на робочому полі текстові надписи (Рис.2.4).

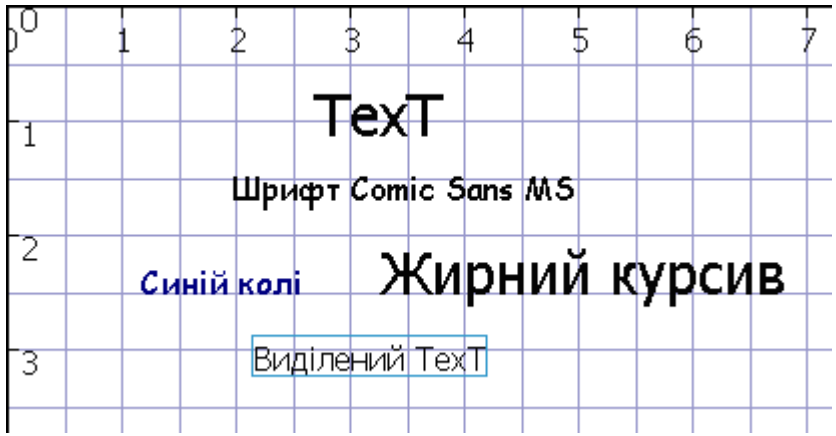


Рис.1.3. Рисування тексту в програмі

Функція – член bool Select (int x, int y), повертає істину коли точка з координатами (x, y) лежить в межах тексту, а bool SelectRect(TRect rect) повертає істину у випадку, коли будь яка з вершин прямокутника який уявно «обрамлює» текст знаходить усередині прямокутника rect. Для того щоб можна було визначити висоту і ширину тексту використані методи класу TCanvas int TextWidth (AnsiString) та NextHeight (AnsiString). Функці void DrawSelect(double zoom) рисує навколо тексту прямокутник, тобто при виділенні тексту він обрамлюється прямокутником (Рис. 1.4).

1.1.4. Клас Line (Лінія)

Батьківським класом для класу Line слугує також клас Pset. Цей клас доповнений однією змінною яка містить координати другої точки лінії. Клас має такі ж конструктори та перевантажений оператор присвоєння, як і попередні класи. Доповнений методами: void SetLine(int a1, int b1,int a2,int b2), void SetB(int x,int y), coord GetB(). SetLine(int a1, int b1,int a2,int b2) задає координати лінії a1,b1 – координати однієї точки a2,b2 – координати другої точки. SetB(int x,int y) задає значення другої точки, а функція SetA(int x, int y), яка задає координати першої точки успадковується, а GetB() повертає координати другої точки. Тут також перевизначені методи virtual void Draw(int zoom), virtual void DrawName(int zoom), bool Select(int,int),virtual bool SelectRect(TRect rect);

Об’єкт даного класу малює пряму, яка перетинає все поле малювання і проходить через дві дані точки. Для цього в метод Draw реалізовувався використовуючи рівняння прямої з кутовим коефіцієнтом [13, с.50]:

$$y = kx + b\ ,(4.1)$$

де $k = tg\alpha$, α - кут який утворює пряма з додатним напрямом осі Ox , b - відрізок, що відтинається прямою на осі Oy .

У методі оголошені додаткові змінні дійсного типу x, y, b та k. Спочатку обчислюється k за формулою:

$$k = \frac{B.y - A.y}{B.x - A.x},$$

де A.x, A.y – координати першої точки , а B.x і B.y – координати другої точки. Потім обчислюється b за формулою:

$$b = A.y - k * A.x.$$

Далі x присвоюється значення x=0, і обчислюється y, за формулою 4.1.

Переміщуємо перо в позицію (x,y) за допомогою методу канви MoveTo(X,Y), надаємо значення змінній x ширину поля, знов обчислюємо y за формулою (4.1) і методом канви LineTo(X,Y) малюємо пряму. Таким чином отримаємо пряму яка проходить від початку поля через дані точки до кінця поля (Рис. 1.5).

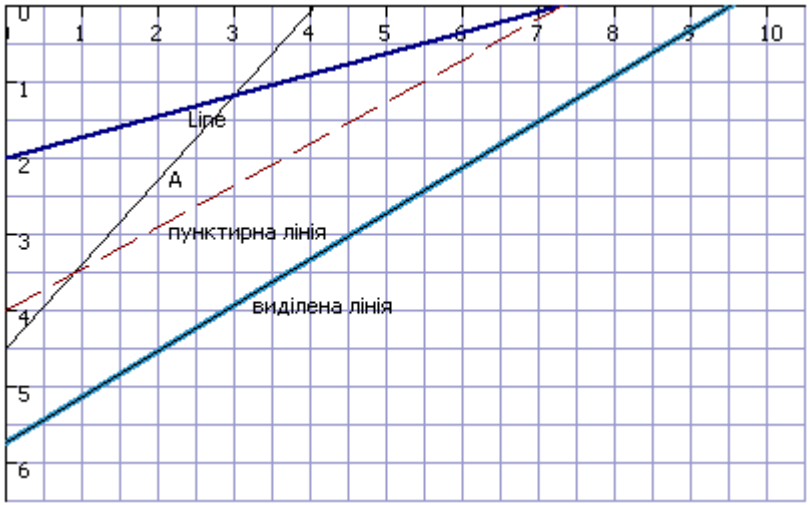


Рис.1.4. Лінія

Метод DrawName() також перевизначений, так як він виводить назву ліні. Щоб назва лінії знаходилась між двома вказаними точками в методі створюються дві змінні цілого типу xc, yc, які містять координати середини між двома даними точками і обчислюються за формулами:

xc=(A.x+B.x)/2,

yc=(A.y+B.y)/2,

де A.x, A.y – координати першої точки , а B.x і B.y – координати другої точки. Потім виводиться текст NameA на координатах (xc,yc) методом TextOutA (int X, int Y,const System::AnsiString Text).

Для того щоб визначити чи знаходиться точка на лінії, складається рівняння лінії, в яке підставляють координати точки. Таким чином функція Select(int,int) повертає істину коли точка з координатами (x,y) зводить до рівносильності рівняння лінії.

Результат роботи функції DrawSelect можна побачити на Рис.1.5.

1.1.5. Клас Ellips (Еліпс)

Клас Ellips (Еліпс) є нащадком класу Line, оскільки еліпс визначається двома точками і його центром, а також повинен мати назву свого центру. Нових методів він не має, крім методів призначених для роботи з півосями. Тут перевизначені методи: virtual void Draw() та virtual void DrawName(), має два конструктори і перенавантажений оператор присвоєння.

Метод Draw(int) реалізовується за допомогою методу Ellipse (B.x,B.y,C.x,C.y), який малює навколо точки еліпс (Рис 1.6), та успадкованого методу DrawEPset (A.x,A.y), який малює пусту точку в центрі еліпса, де A.x, A.y обчислюється за формулами:

$$A.x = \frac{B.x + C.x}{2},$$

$$A.y = \frac{B.y + C.y}{2}.$$

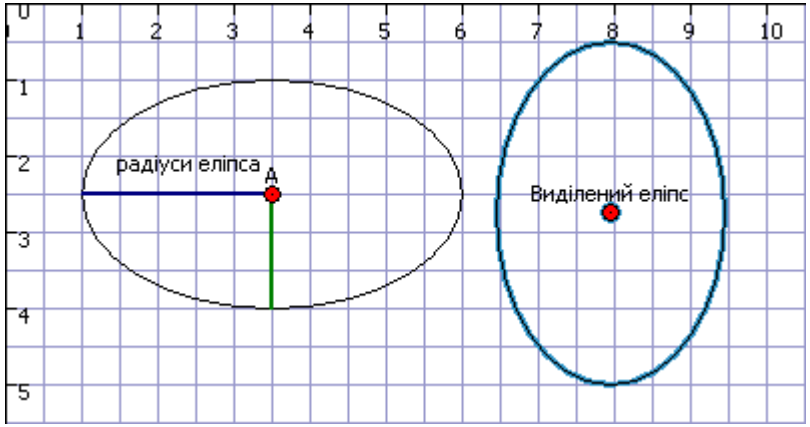


Рис.1.5. Еліпс

Метод DrawName(int) виводить назву центра еліпса над його центром, за допомогою методу TextOutA (int,int,AnsiStriang) (Рис.1.6).

В цьому класі також аналогічна реалізації функцій запису та зчитування з файлу і функції виділення: void WritePar (TFileStream*), void ReadPar(TFileStream*), bool Select (int x, int y)та bool SelectRect(TRect rect).

1.1.6. Клас Length (Відрізок)

Цей клас є нащадком класу Line (Лінія). Він доповнений трьома новими методами: void SetLenght(int a1, int b1,int a2,int b2), void SetNameLenght(AnsiString A,AnsiString B) та void DrawELine(int a1, int b1,int a2,int b2). SetLenght(int a1, int b1,int a2,int b2) задає координати відрізка. SetNameLenght(AnsiString A,AnsiString B) задає назви кінці відрізка, а DrawELine(int a1, int b1,int a2,int b2) малює лінію від точки (a1,b1) до точки (a2,b2), а потім малює точки на координатах (a1,b1) і (a2,b2) успадкованим методом DrawEPset(int x, int y) (Рис.1.7).

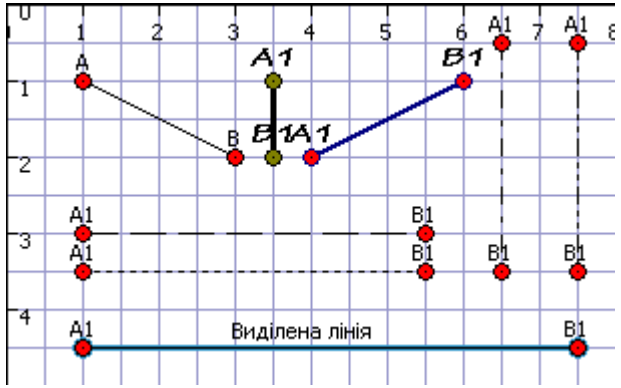


Рис. 1.6.Відрізок

Клас Length перевизначає методи Draw(int) та DrawName(int) а також методи для роботи з файлами void WritePar(TFileStream*), void ReadPar(TFileStream*) та методи для виділення bool Select (int x, int y)та bool SelectRect(TRect rect). Їх реалізація аналогічна реалізації в попередніх методах.

1.1.7. Клас Rectang (Прямокутник)

Батьківським класом для класу Rectangle є клас Length, оскільки для того щоб намалювати прямокутник достатньо, як і для відрізку, дві точки. Цей клас доповнений двома новими змінними текстового типу: AnsiString NameC та AnsiString NameD, а також двома новими методами: void SetRectang(int a1, int b1,int a2,int b2) та SetNameRectang(AnsiString A,AnsiString B,AnsiString C,AnsiString D). SetRectang задає значення координат двох точок за якими малюється прямокутник. SetNameRectang(AnsiString A,AnsiString B,AnsiString C,AnsiString D) надає значення змінним AnsiString NameC та AnsiString NameD, які є назвами двох інших точок прямокутника.

Перевизначений метод Draw (int zoom) реалізується за допомогою методів канви Polygon , які малюють прямокутник, та

успадкованого методу DrawEPset (X,Y), який малює точки на вершинах прямокутника (Рис. 1.7).

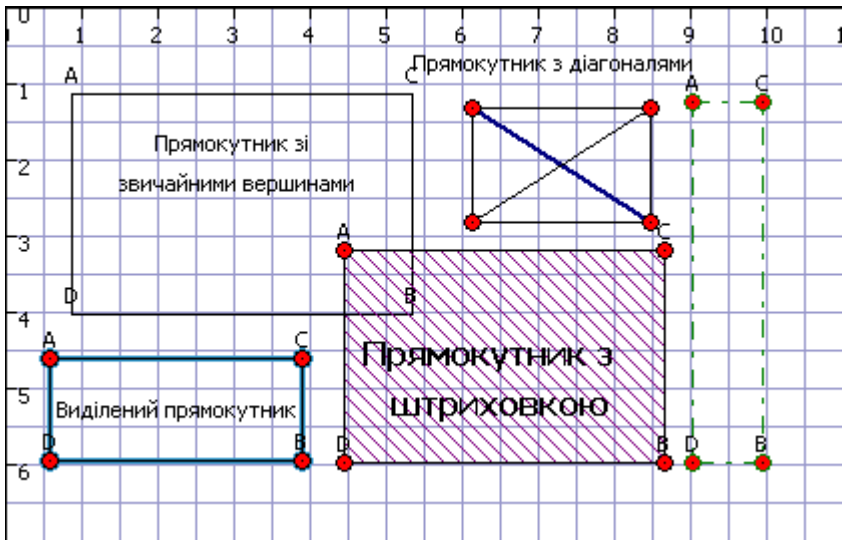


Рис.1.7. . Прямокуник.

Ще один перевизначений метод DrawName(int zoom) виводить назви вершин справа або зліва вершини, враховуючи масштаб.

В класі Rectang передбачені дві діагоналі, які можуть бути нарисовані різними стилями (Рис. 1.8). Для цього в класі оголошені дві змінні типу diag, вони складаються з поля типу bool та поля типу TLineMode. Таким чином користувачеві досить вибрати відповідник пункт контекстного меню, попередньо встановивши параметри лінії, і на рисунку з'явиться відповідна діагональ (Рис.1.8).

1.1.8. Клас Triangle (Трикутник)

Клас Triangle є нащадком класу Length і доповнений двома змінними coord C та AnsiString NameC, що містять координати та назву третьої точки. Він також доповнений методами void

SetTriangle(int a1, int b1,int a2,int b2,int a3,int b3),void SetC(int x,int y) ,coord GetC().SetTriangle(int a1, int b1,int a2,int b2,int a3,int b3) ініціалізує координати всіх трьох точок, а SetC(int x,int y) тільки координати третьої точки. GetC() повертає значення третьої точки. Інші методи успадковуються від батьківських класів.

Методи Draw(int) та DrawName(int) в цьому класі також перевизначені, і реалізовані відповідно до класу, результат цих функцій наведено на Рис.1.9. Так само і методи bool Select(int X, int Y), bool SelectRect(TRect rect);

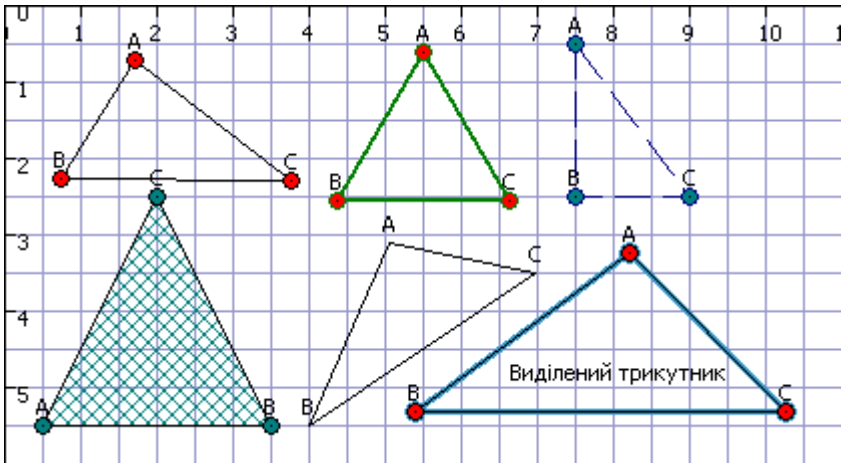


Рис.1.8. Трикутник.

Як і в попередніх класах в класі Triangle передбачено ряд конструкторів та перевантажено оператор присвоєння, а також перевизначено функції для роботи з файлами та виділенням.

1.1.8.1. Висота, медіана, бісектриса та середня лінія трикутника

У кожному трикутнику можна провести три висоти, три медіани, три бісектриси та три середні лінії. Для того щоб користувач

легко вибравши відповідний пункт контекстного меню міг створити в трикутнику одну з цих ліній, у класі визначено ряд функцій для їх створення.

Що стосується медіани та середньої лінії трикутника, то тут задача не є складною, оскільки для їх рисування потрібно знати середину сторони трикутника щоб провести медіану і дві середини трикутника для середньої лінії. Знайти координати середини відрізка можна за наступними формулами [13, с.49]:

$$x = \frac{x_1 + x_2}{2}; y = \frac{y_1 + y_2}{2},$$

де точка (x, y) – середина відрізка, кінці якого мають координати (x_1, y_1) та (x_2, y_2). Тому для медіан трикутника оголошені три змінні типу Some_build, який містить координати кінця медіани та стиль лінії, а для середніх ліній визначено також три змінні типу CenterLine, який містить координати пари точок, що лежать на серединах сторін трикутника, та стиль ліній. Їх рисування на трикутнику відбувається у методі Draw, результат рисування трикутника з медіанами, середніми лініями видно на Рис.1.10.

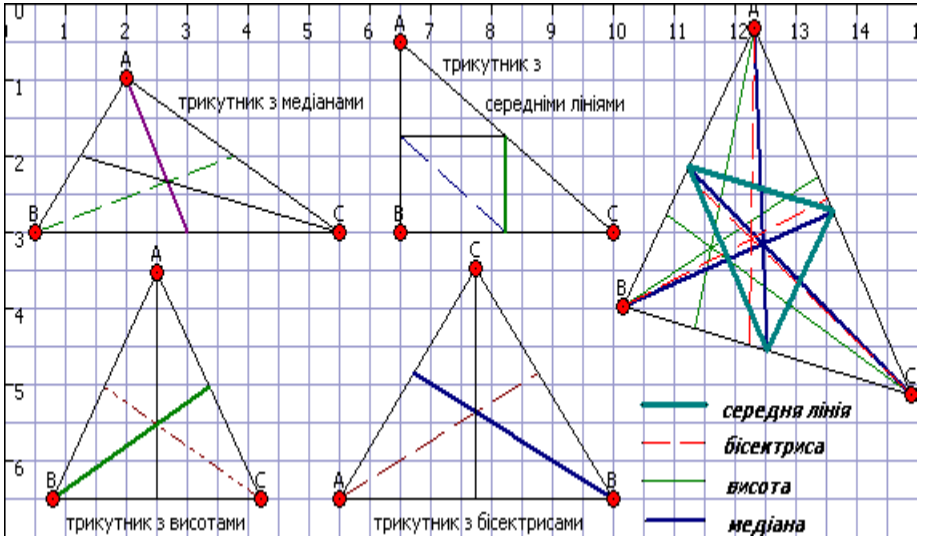


Рис.1.9.Трикутники з висотами, медіанами, бісектрисами, середніми лініями.

Що стосується висоти та бісектриси то тут задача складніша. По перше потрібно скласти рівняння сторони трикутника до якої опущена висота чи бісектриса, потім скласти рівняння самої висоти чи бісектриси, і вже тоді знайти точку їх перетину.

Рівняння прямої має вигляд: $Ax + By + C = 0$. Для знаходження коефіцієнтів скористаємося наступними формулами [14]:

$$A = y_1 - y_2,$$
$$B = x_2 - x_1,$$
$$C = -(Ax_1 + By_1).$$

В програмі ці формули реалізовані у методі класу Triangle FindIndex.

Скласти рівняння висоти, знаючи рівняння прямої не складно. Оскільки умова перпендикулярності прямих l_1 (сторона) та l_2 (висота),

заданих загальним рівняннями $A_1x + B_1y + C_1 = 0$ та $A_2x + B_2y + C_2 = 0$ має вигляд: $A_1A_2 + B_1B_2 = 0$, то $A_2 = -B_1, B_2 = A_1$. Для того щоб знайти C_2 досить підставити координати висоти з якої опущено перпендикуляр, тобто $C_2 = -(A_2x + B_2y)$ *((x,y)* – координати вершини). Метод класу Triangle який знаходить точку перетину сторони та висоти має назву FindHeight.

Рівняння бісектриси між прямим l_1 та l_2 заданих загальним рівняннями $A_1x + B_1y + C_1 = 0$ та $A_2x + B_2y + C_2 = 0$ має вигляд [13, с.52]:

$$\frac{A_1x + B_1y + C_1}{\sqrt{A_1^2 + B_1^2}} = \pm\frac{A_2x + B_2y + C_2}{\sqrt{A_2^2 + B_2^2}}.$$

Звідси можна вивести наступні коефіцієнти рівняння бісектриси $A_bx + B_by + C_b = 0$:

$$\begin{aligned} A_b &= A_1k_2 \pm A_2k_1, \\ B_b &= B_1k_2 \pm B_2k_{1,} \\ C_b &= C_1k_2 \pm C_2k_1. \end{aligned} \text{ де } \begin{aligned} k_1 &= \sqrt{A_1^2 + B_1^2}, \\ k_2 &= \sqrt{A_2^2 + B_2^2}. \end{aligned}$$

Функція класу Triangle яка знаходить точку перетину сторони та бісектриси має назву FindBisector.

Маючи рівняння сторони трикутника та прямої можна знайти їхню точку перетину. Щоб знайти точку перетину, достатньо скласти з двох рівнянь систему і розв'язати її:

$$\begin{cases} A_1x + B_1y + C_1 = 0, \\ A_2x + B_2y + C_2 = 0. \end{cases}$$

Користуючись формулою Крамера, знаходимо розв'язок системи, який і буде шуканою точкою перетину:

$$x = -\frac{\begin{vmatrix} C_1 & B_1 \\ C_2 & B_2 \end{vmatrix}}{\begin{vmatrix} A_1 & B_1 \\ A_2 & B_2 \end{vmatrix}} = -\frac{C_1B_2 - C_2B_1}{A_1B_2 - A_2B_1},$$

$$y = -\frac{\begin{vmatrix} A_1 & C_1 \\ A_2 & C_2 \end{vmatrix}}{\begin{vmatrix} A_1 & B_1 \\ A_2 & B_2 \end{vmatrix}} = -\frac{A_1C_2 - A_2C_1}{A_1B_2 - A_2B_1}.$$

Якщо знаменник дорівнює нулю, то система не має розв'язку (прямі паралельні і не співпадають) або має безліч розв'язків (прямі співпадають). [15].

Реалізація знаходиться у функції класу Triangle CrossPoint.

В результаті користувач може легко та точно нарисувати бісектриси та висоти в трикутнику (Рис. 1.10).

1.1.9. Класи ARC_3 та ARC_Center (Дуга)

Класи ARC_3 та ARC_Center є похідним класом від класу Triangle. Класи, як і інші класи, мають конструктори і перенавантажені оператор присвоєння.

Методи Draw (int) малює дугу, використовуючи метод Arc(xl,yl,x2,y2,x3,y3,x4,y4), де параметри x1, y1, x2, y2 визначають еліпс, частиною якого є дуга. Параметри x3 и y3 задають початкову, а x4 и y4 — кінцеву точку дуги. Початкова і кінцева точка дуги — це точка перетин меж еліпса і прямої, проведеної із центра еліпса в точку з координатами x3, y3 (x4, y4). Метод Arc рисує дугу проти часової стрілки від початкової точки до кінцевої (Рис.1.11).

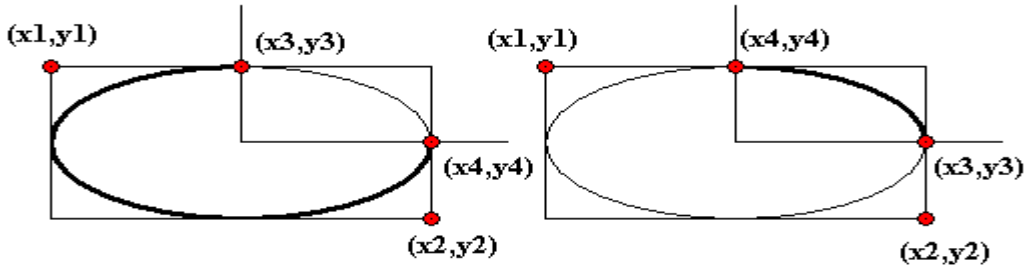


Рис.1.10. Робота функції Arc

Метод Draw(int) класу ARC_3 рисує дугу за трьома точками що лежать на даній дузі, а цей самий метод класу ARC_Center рисує дугу як частину кола, центр якого вказує користувач (Рис 1.12).

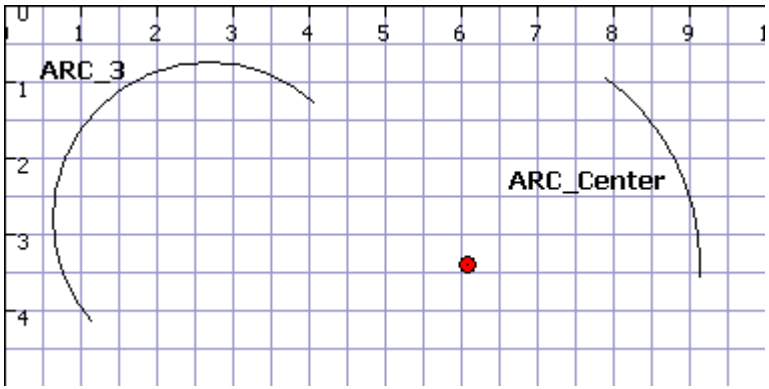


Рис.1.11 Нарисовані об'єкти класів ARC_3 та ARC_Center

Метод DrawName(){} пустий, оскільки написів на дузі не передбачено. Як і в інших класах в даних класах існують функції для рисування фігур та роботи з файлами.

1.1.9.1. Побудова дуги за трьома точками

Для того щоб користувачу було легко нарисувати дугу потрібно було максимально скоротити і спростити спосіб задавання точок за якими будується дуга. Таким чином, щоб нарисувати дугу вписану в квадрат досить 3 точки. Для цього потрібно мати координати центра кола, частиною якого є дуга. Таким чином дугу (Рис.1.13) користувач може нарисувати три рази клацнувши на полі. Як видно всі ці точки (на які користувач задає клацанням на полі), тому задача зводиться до побудови кола за трьома точками, а також знаходження його центру та радіусу, це потрібно щоб задати прямокутну область у функції Arc (Рис. 1.13).

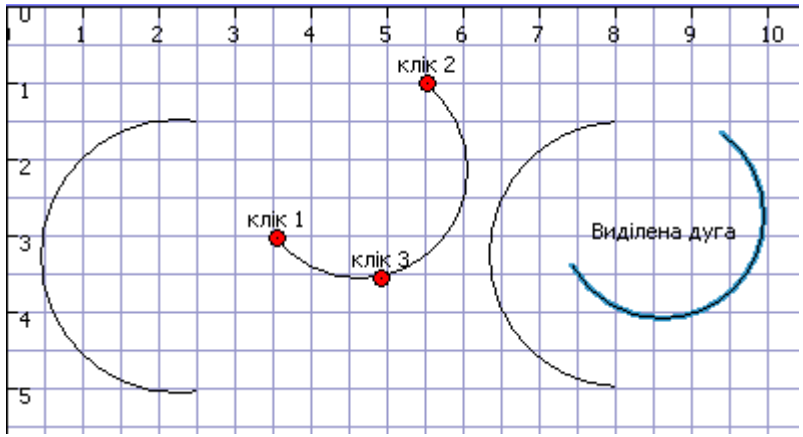


Рис.1.12. Інструмент дуга.

Отже, маємо три точки на площині p_1, p_2 та p_3 (Рис.1.14).

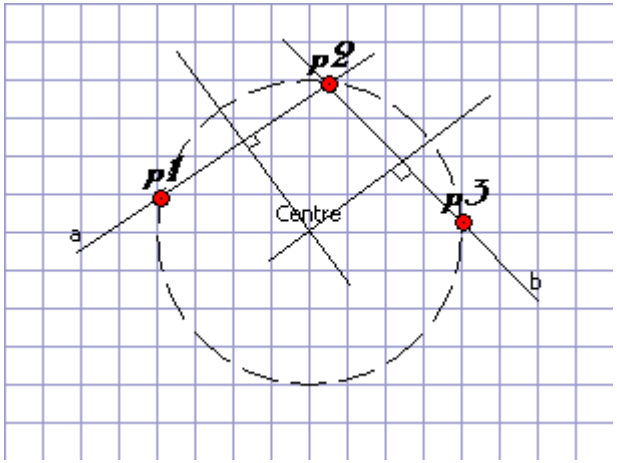


Рис.1.13. Положення точок.

Проведемо через пари точок дві прямі. Нехай лінія a проходить через p_1 і p_2, а пряма b - через p_2 та p_3 (Рис 1.14). Рівняння цих прямих будуть мати вигляд:

$$y_a = m_a(x - x_1) + y_1, \text{ та } y_b = m_b(x - x_2) + y_2,$$

де m - коефіцієнт нахилу лінії, який отримуємо з:

$$m_a = \frac{y_2 - y_1}{x_2 - x_1}, \text{ та } m_b = \frac{y_3 - y_2}{x_3 - x_1}.$$

Центр кола – знаходиться на перетині двох перпендикулярних прямих, які проходять через середини відрізків p_1p_2 та p_2p_3. За ознакою перпендикулярності прямих, пряма перпендикулярна до лінії з коефіцієнтом нахилу m має коефіцієнт нахилу $1/m$, отже рівняння прямих, перпендикулярних a і b, зо проходять через середини p_1p_2 і p_2p_3 будуть:

$$y_a' = -\frac{1}{m_a}\left(x - \frac{x_3 + x_2}{2}\right) + \frac{y_1 + y_2}{2},$$

$$y_b' = -\frac{1}{m_b}\left(x - \frac{x_3 + x_3}{2}\right) + \frac{y_3 + y_2}{2}.$$

Вони перетинаються в центрі , і розв'язок відносно x дає:

$$x = \frac{m_a m_b(y_1 - y_3) + m_b(x_1 + x_2) - m_a(x_2 + x_3)}{2(m_b - m_a)}.$$

Значення y обчислюємо підстановкою отриманого x у рівняння одного з перпендикулярів. Радіус знаходиться елементарно, наприклад, потрібно знайти відстань від точки p_1 до центру.

Тут слід врахувати наступні зауваження:

- Знаменник $(m_b - m_a)$ дорівнює нулю, коли прямі паралельні. У цьому випадку вони співпадають, тобто кола не існує.
- Якщо яка-небудь із прямих вертикальна, то її коефіцієнт нахилу дорівнює нескінченості. Цього можна уникнути змінивши, змінивши порядок точок так, щоб вертикальних ліній не з'являлося. [16]

Всі вищенаведені формули реалізований в методі coord CalcCenter(coord A1,coord A2, coord A3) класу ARC_3.

1.1.10. Класи Rhombus (Паралелограм) та Tetragon (Чотирикутник)

Клас Rhombus, що є нащадком класу Triangle, призначений для створення паралелограмів. Для того щоб нарисувати паралелограм достатньо мати координат трьох вершин, а четверту вершину можна обчислити за формулою:

$$x = (A.x + C.x) - B.x;$$
$$y = (A.y + C.y) - B.y.'$$

де A(A.x,A.y), B(B.x,B.y) і C(C.x,C.y) – координати вершин паралелограма. Ці формули реалізовані в методах класу Rhombus int GetDx() та int GetDy(), які в свою чергу викликаються функцією coord GetD(), що повертає координати четвертої точки паралелограма.

У цьому класі додано нову змінну текс того типу NameD, яка зберігає ім'я четвертої вершини. Відповідно метод DrawName(), який відповідає за напис над вершинами, перевизначений. Також додані змінні для збереження інформації про діагоналі та висоти і відповідно перевизначені ряд методів для висот, діагоналей та середніх лінії.

Метод Draw(int), після виклику методу класу coord GetD(), за допомогою функції Polygon рисує паралелограм (Рис. 1.15).

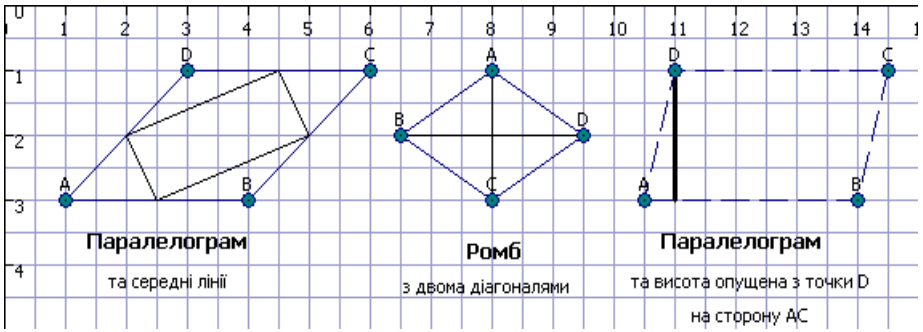


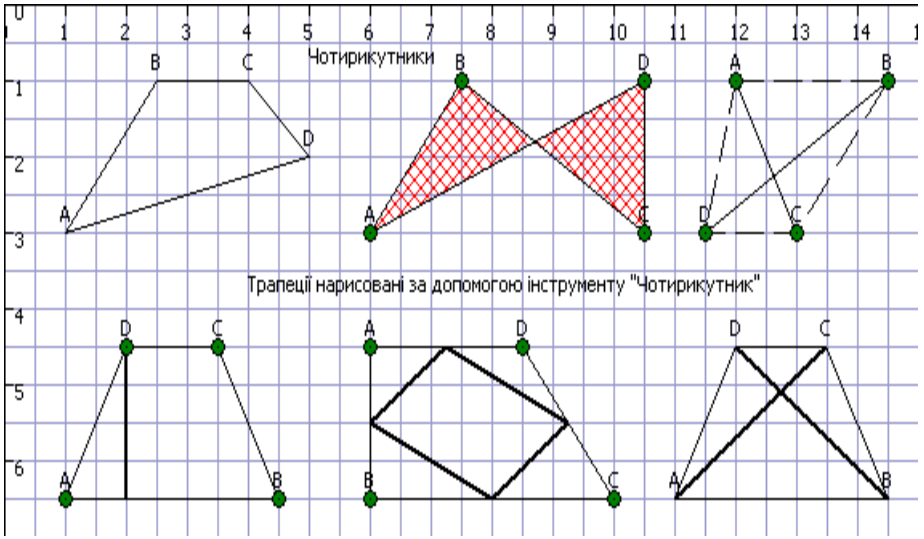
Рис.1.14. Обєкти класу Rhombus

Як і в інших класах методи для роботи з файлами та виділеннями фігур перевизначені. Тут також існують конструктори різних типів та перевантажені оператори присвоєння.

Наступник класом в даній ієрархії є клас Tetragon, нащадок класу Rhombus. Цей клас мало чим відрізняється від попереднього, такі ж самі методи перевизначені і додано такі самі конструктори та

перенавантажені оператори. Однак в класу Tetragon додано нову змінну типу coord – D, яка зберігає координати четвертої вершини.

Як і в батьківському класі, в метод Draw класу Tetragon рисує чотирикутник методом Polygon (Рис. 1.16). Висоти, діагоналі, середні лінії успадковуються від попередніх класів, а їх графічне відтворення



успадковується від класу Rhombus.

Рис.1.15. Обєкти класу Tetragon

Таким чином класи Rhombus та Tetragon є останніми класами в даній ієрархії класів.

1.2. Створення графічного інтерфейсу користувача

Графічний інтерфейс користувача (ГІК, англ. GUI, Graphical user interface) — інтерфейс між комп'ютером і його користувачем, що використовує піктограми, меню, і вказівний засіб для вибору функцій та виконання команд.

ГІК — система засобів для взаємодії користувача з комп'ютером, заснована на представленні всіх доступних користувачеві системних об'єктів і функцій у вигляді графічних компонентів екрану (вікон, значків, меню, кнопок, списків і т. п.). При цьому, на відміну від інтерфейсу командного рядка, користувач має довільний доступ (за допомогою клавіатури або пристрою координатного введення типу «миша») до всіх видимих екранних об'єктів.

Вперше концепція ГІК була запропонована вченими з дослідницької лабораторії Xerox PARC в 1970-х, але отримала комерційне втілення лише в продуктах корпорації Apple Computer. У операційній системі AMIGAOS ГІК з багатозадачністю був використаний в 1985 р. В даний час ГІК є стандартній складовій більшості доступних на ринку операційних систем. [17]

1.2.1. Створення головної форми та розміщення к

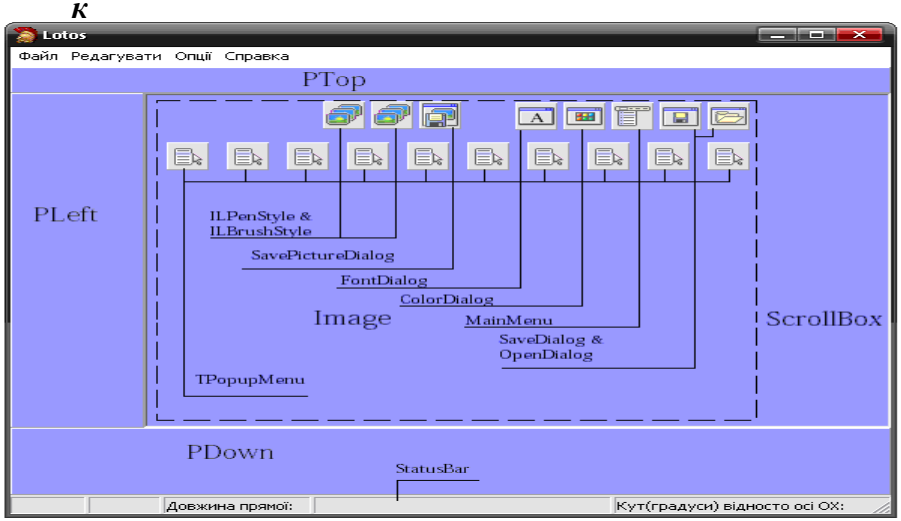


Рис. 1.16. Головна форма

Головне вікно графічного редактора повинно мати робочу область, тобто область на якій створюється зображення, також панелі інструментів та палітру кольорів (Рис. 2.17, додаток В). Панель швидкого доступу робить головне вікно зручнішим під час роботи.

Як видно з Рис. 2.17 на формі Form1 розміщено як візуальні так і невізуальні компоненти. З візуальних компонентів маємо: Image типу TImage, що знаходиться на панелі з прокруткою ScrollBox, панелі PLeft, PDown та PTop, на яких в свою чергу розміщені кнопки, поля вводу та палітра кольрів, рядок стану StatusBar. Також на формі розміщені такі не візуальні об’єкти: списки зображень ILPenStyle та ILBrushStyle типу TImageList, головне меню MainMenu, ряд контекстних меню типу TPopupMenu, діалоги: ColorDialog, FontDialog, SavePictureDialog, SaveDialog та OpenDialog (Рис. 2.17).

Об’єкт Image є полем для малювання. На панелях розміщені кнопки, палітра та інше. За допомогою діалогів, забезпечується можливість зберігання та відкриття файлів з розширенням .qp, а також експортувати створене зображення в файл з розширенням .bmp. Рядок стану StatusBar відображання інформацію, таку як координати курсору, кут нахили прямих або їх довжини.

1.2.2. Панелі та розміщення компонентів на них

Як уже було сказано, в головному вікні програми розміщені три панелі, та панель з прокруткою. Панелі є контейнерами, які служать для об’єднання інших управляючих елементів. Вони можуть виконувати чисто декоративні функції, наочно об’єднуючи компоненти, зв’язані між собою за призначенням, так і функції управління, організовуючи смільну роботу своїх дочірніх компонентів.[1,с.217]

Кожна з панелей має своє розташування: панель PDown розміщена внизу, PLeft – зліва, а PTop, відповідно, нагорі. Таким

чином при зміні розмірів вікна панелі з встановленими на них компонентами будуть розташовуватися відповідно.

На панелі PDown, що знаходиться внизу вікна, розташовані компоненти які дозволяють задавати параметри створюваної фігури: колір, стиль лінії та заливки, а також додаткові параметри (Рис.2.18).

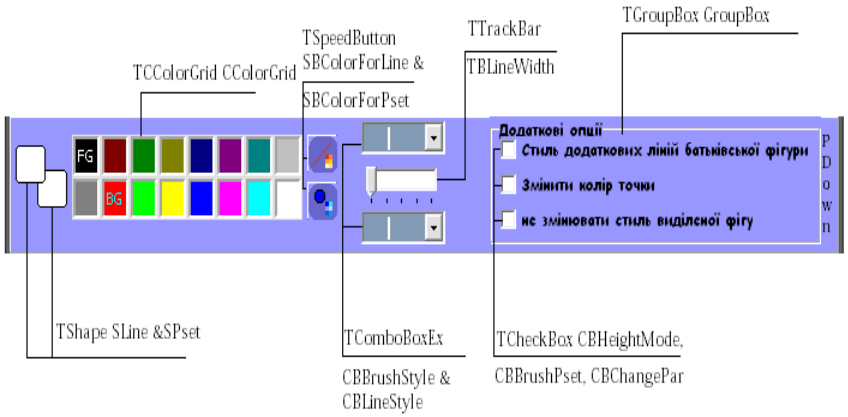


Рис. 1.17. Панел PDown та компоненти, що розміщені на ній .

Для вибору кольору на панелі розміщений об'єкт CColorGrid типу TCColorGrid та кнопки SBColorForLine, SBColorForPset. При натиску на цих кнопках лівою кнопкою миші викликається діаголог ColorDialog типу TColorDialog, він і забезпечує наявність стандартної розширеної панелі кольорів Windows. Для відображення поточного кольору кисті та штриховки використані об'єкти типу TShape SLine та SPset.

Встановлення ширини лінії відбувається за допомогою компоненту TBLineWidth типу TTrackBar (Рис.2.18). В обробнику події TBLineWidthChange (TObject *Sender) цього компоненту ширина ліній виділеного об'єкту встановлюється значенням TBLineWidth->Positio. Це ж значення товщини лінії встановлюється під час створення нової фігури.

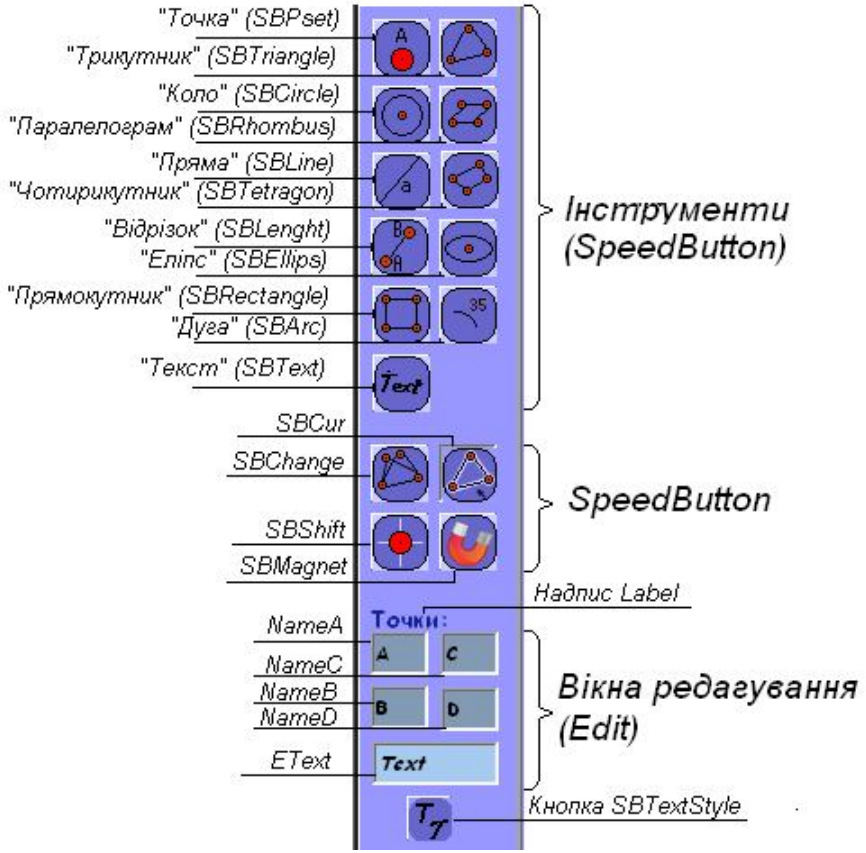


Рис. 1.18. Панель PLeft

Для встановлення типу лінії та штриховки використані випадних списків з зображеннями та текстом – CBBrushStyle та CBLineStyle. В їх головній властивості Items вказаний список строк. Індекс вибраної строки повертається за допомогою властивості ItemIndex. Зображення у списки завантажуються з списків зображень ILPenStyle та ILBrushStyle.

Для зручності користування програмою на панелі знаходяться деякі часто використовувані опції. Індикатори опції CBHeightMode, CBBrushPset і cBChangePar знаходяться на компоненті типу TGroupBox, що дозволяє об'єднати індикатори (Рис.2.18).

На панелі зліва – PLeft знаходиться кнопки інструментів та поля вводу текстової інформації такої, як назви вершин фігур чи текст для інструменту «Текст» та ін. (Рис.2.19).

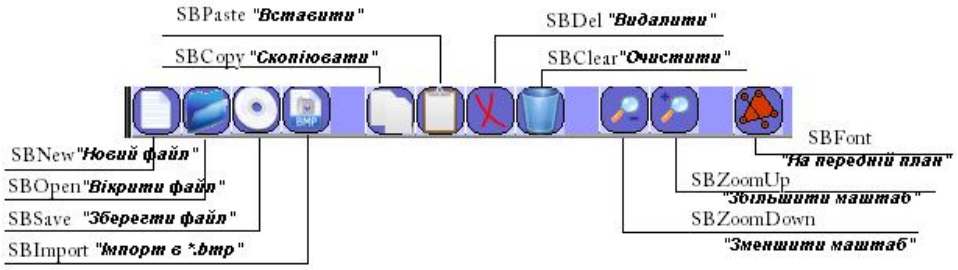


Рис. 1.19 Панель PTop

Всі кнопки на панелі є кнопками з фіксацією SpeedButton. Такі кнопки часто використовуються для інструментальних панелей кнопок з фіксацією натиснутого стану і в якості швидких кнопок, дублюючих команди меню. Щоб завантажити рисунок для кнопок використовується властивість Glyph. Для того щоб тільки одна кнопка могла знаходитися у натиснутому стані потрібно у властивість GroupIndex всіх кнопок групи вказати одне і те саме число. У такій групі з індексом – «1» знаходяться всі кнопки крім SBShift, SBMagnet та SBTextStyle. Кнопка SBTextStyle викликає діалог вибору шрифту – FontDialog, таким чином при натисненні кнопки користувач побачить стандартний діалог вибору шрифту Windows.

Для введення назв вершин та тексту для виведення використані вікна редагування типу TEdit, оскільки це зручний компонент для вводу і редагування тексту. Основна властивість – Text. А от для найпростішого виведення тексту використовують

компонент Label (мітка), основна властивість якого - Caption. Такий компонент знаходиться на панелі PLeft (Рис.2.19).

Панель PTop знаходиться уверху вікна та містить кнопки, що дублюють функції меню і деякі інші кнопки: «Новий файл», «Відкрити Файл», «Зберегти файл», «Імпорт в *.bmp», «Вставити», «Скопіювати», «Видалити», «Очистити» та «Зменшити масштаб», «Збільшити масштаб», «На передній план» (Рис.2.20).

Всі кнопки на панелі PTop типу SpeedButton. Щоб кнопки не залишалися в натиснутому стані властивість GroupIndex всіх кнопок має значення – «0». Як було сказано, більшість кнопок дублюють пункти меню, а от кнопки SBZoomDown та SBZoomUp зменшують і збільшують масштаб зображення, кнопка SBFont виводить на передній план виділені фігури.

Панелі з прокруткою ScrollBox використана так як розмір поля для рисування може бути змінено користувачем до великих розмірів. Таким чином для того щоб мати змогу створювати та редагувати зображення великого розміру компонент Image поміщений в панель з прокруткою.

1.2.3. Малювання курсором на канві

Для того щоб користувач міг створювати певне зображення на полі малювання (канві) за допомогою маніпулятора, потрібно задати певний код в таких оброблювачах події компоненту Image: OnMouseDown, OnMouseMove, OnMousUp. При цьому потрібно враховувати такі дані:

- яка кнопка панелі інструментів знаходиться в натиснутому положенні.
- яка кнопка «миші» натиснута
- які клавіші утримуються
- чи виділена на даний момент якась фігура
- яка вершина вказується

Коли користувач натискає кнопку «миші» наступає подія OnMouseDown. В цей час, у випадку коли була натиснута права кнопка «миші», визначається чи вона натиснута над якоюсь фігурою, або в даний момент виділена фігу і яка це фігура. В залежності яка фігура виділена компоненту Image назначається відповідне контекстне меню.

Якщо ж була натиснута ліва кнопка то лічильник кліків потрібно збільшити на одиницю. Далі потрібно визначити яка фігура створюється (змінюється або переміщається) шляхом визначення затиснутої кнопки на панелі інструментів (на клавіатурі. Далі в залежності від того яке значення має лічильник кліків вже індивідуально для кожної фігури задаються координати відповідної вершини фігури. Коли остання вершина фігури задана відбувається остаточне рисування фігури в режимі pmCopy та обнулення лічильника.

Подія OnMouseMove відбувається під час руху вказівника «миші» по компоненту Image. Тут аналогічним чином відбувається дії у залежності від того яка фігури рисується та яке значення має лічильник кліків. Під час рисування фігур користувач не зобов'язаний утримувати кнопки «миші», однак перерисування фігур в залежності від положення курсору відбувається з кожним його рухом. Якщо під час руху стрілки виділені фігури і затиснула ліва кнопка «миші» то відбувається зміщення фігур відносно попереднього їх положення.

В обробнику події OnMouseUp вписані команди, котрі повинні відбуватися в той час коли кнопка «миші» відпущена. Реагувати на цю подію програма повинна у випадку тільки якщо відпущена ліва кнопка «миші». У випадку коли користувач задавав квадратну область виділення то відбувається заповнення контейнеру для виділених об'єктів тими фігурами, вершини яких потрапляють до області видалення. Якщо ж користувач створює точку або текст, то тут задають їх координати. Також у обробнику подій OnMouseUp відбувається перерисування фігур які переміщувалися.

Тами чином для того щоб намалювати точку або текст, потрібно натиснути ліву клавішу «миші», встановити курсор в потрібне положення і відпустити клавішу миші. Для малювання решти фігур потрібно один раз натиснути ліву клавішу миші, потім рухаючи курсором отримати бажане зображення і зафіксувати це зображення вдруге натиснувши на ліву клавішу миші, так допоки не буде задана фігура. Для того щоб виділити фігуру досить натиснути на ліву кнопку «миші» коли стрілка знаходиться над фігурою, або обвести хоча б одну її вершину прямокутником інструменту «Виділення». Щоб перемістити виділені фігури слід у переміщувати курсор і утримувати ліву кнопку маніпулятора.

1.2.4. Створення сітки та лінійки

На полі якому створюється зображення доцільно створити сітку та лінійку яка б допомагала наочно оцінити розміри, відстані та пропорції. Функція що рисує сітку на полі TImage Image1 має наступний прототип:

```
void Table(TImage *Image1,double x,double y,double zoom);
```

Змінні double x та double y вказують кількість пікселів в одному сантиметрі по горизонталі та вертикалі, double zoom – вказує масштаб. Ця функція рисує горизонтальні та вертикальні лінії кожні 0,5 сантиметра при масштабі 1:1 (Рис.2.21).

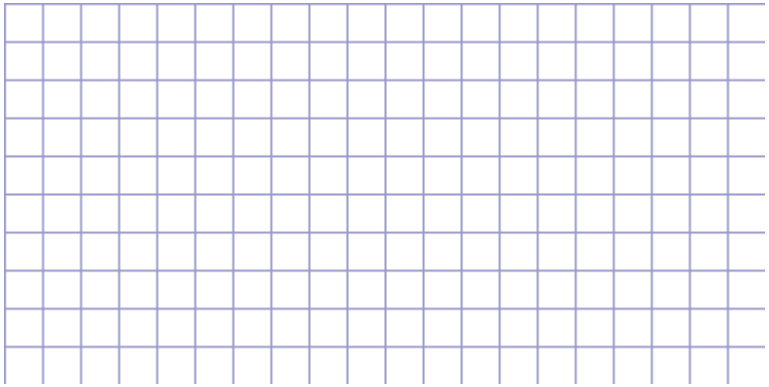


Рис. 1.20. Сітка на полі для рисування

Для рисування лінійки, тобто позначок та цифр кожного сантиметру на полі (Рис.2.22) Image1 призначена функція з таким прототипом:

void Ruler(TImage *Image1,double x,double y,double zoom);

Тут також змінні x та double y вказують кількість пікселів в одному сантиметрі по горизонталі та вертикалі та double zoom вказує масштаб.

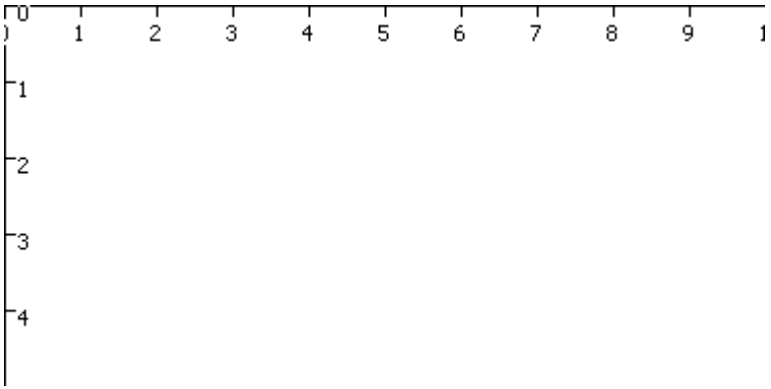


Рис. 1.21. Лінійка на полі для рисування

Функція void Table_Ruler викликає спочатку функцію Table а потім Ruler, тобто рисує спочатку сітку а потім лінійку(Рис. 2.23).

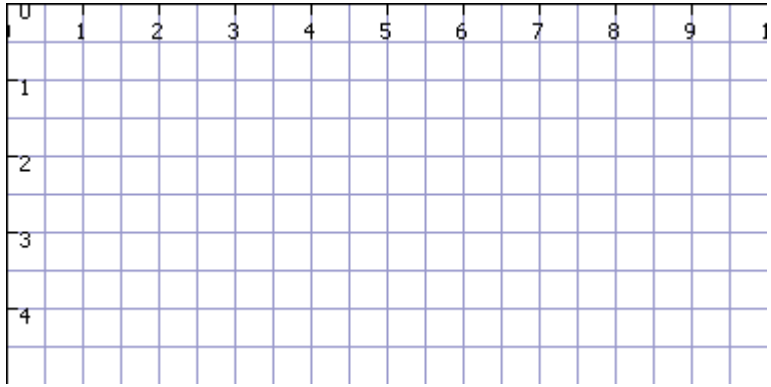


Рис. 1.22. Сітка і Лінійка на полі для рисування.

Таким чином існують три функції з однаковими параметрами, отже можна створити вказівник яким може вказувати на будь – яку з цих функцій. Так в програмі створено вказівник наступним чином (докладніше пункт 1.5):

void (*Tabl) (TImage *Image1,double x,double y,double zoom);

У випадку коли потрібно нарисувати фон програма звертається до вказівника, котрий в свою чергу звертається до одної з трьох функцій. На початку програми він вказує на функцію Table_Ruler. Коли користувач, використовуючи головне меню (Рис.2.24) встановлює потрібну комбінацію вказівнику присвоюється нове значення.

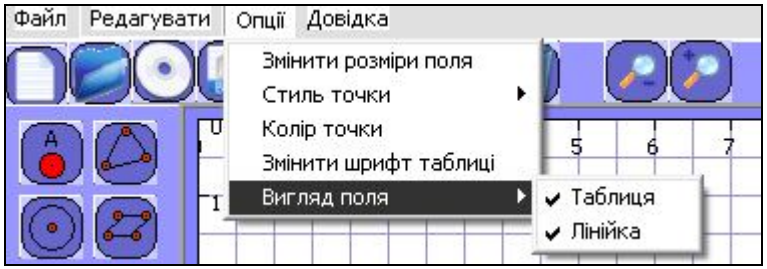


рис. 1.23. Налаштування вигляду поля для рисування за допомогою головного меню

Для того щоб визначити кількість пікселів на один сантиметр враховуючи особливості кожного монітору скористаємося функцією Windows API GetDeviceCaps, що використовує в якості вхідного параметру контекст графічного пристрою. За допомогою цієї функції можна отримати специфічну інформацію щодо екранного пристрою[2, с.345].

Для знаходження кількості пікселів в сантиметрі по горизонталі та вертикалі скористаємося наступним кодом:

```
double pixXFact
=GetDeviceCaps(GetDC(HWND(NULL)),LOGPIXELSX)/2.54;

double pixYFact
=GetDeviceCaps(GetDC(HWND(NULL)),LOGPIXELSY)/2.54;
```

Оскільки значення, що повертає функція це число пікселів на дюйм, то потрібно поділити його на 2.54, оскільки саме стільки сантиметрів у одному дюймі.

1.2.5. Обробка координат

Положення фігури визначається її положенням на канві, тобто по координатами по осям. Координати задаються під час вищерозглянутих подій OnMouseDown, OnMouseMove, OnMouseUp. У обробники цих подій передаються параметри різних типів в тому числі і положення курсору, тобто число пікселів від лівого краю – int X, та від верхнього – int Y. Ці значення і приймають вершини фігур, однак вони залежать від масштабу, а також зазнають інших перетворень.

Що стосується масштабу, то тут координати перетворюються наступним чином:

$$X = \frac{X}{zoom},$$
$$Y = \frac{Y}{zoom.},$$

де zoom – масштаб.

Щоб створювати точний рисунок по перетинам ліній на полі (Рис. 2.25) потрібно затримати Shift або встановити у натиснуте положення кнопку з піктограмою:  . Для того щоб надати вершині фігури координати перетину також слід виконати перетворення над координатами за наступними формулами:

$$X = \left\lfloor \frac{PixXFact}{2} + 0.5 \right\rfloor \cdot n_x$$

$$Y = \left\lfloor \frac{PixYFact}{2} + 0.5 \right\rfloor \cdot n_y,$$

де PixXFact – кількість пікселів на сантиметр по горизонталі, PixYFact – кількість пікселів на сантиметр по вертикалі, а n_x та n_y обчислюються за формулами:

$$n_x = \left\lfloor (X \div \left\lfloor \frac{PixXFact}{2} + 0.5 \right\rfloor) + 0.5 \right\rfloor,$$
$$n_y = \left\lfloor (Y \div \left\lfloor \frac{PixYFact}{2} + 0.5 \right\rfloor) + 0.5 \right\rfloor.$$

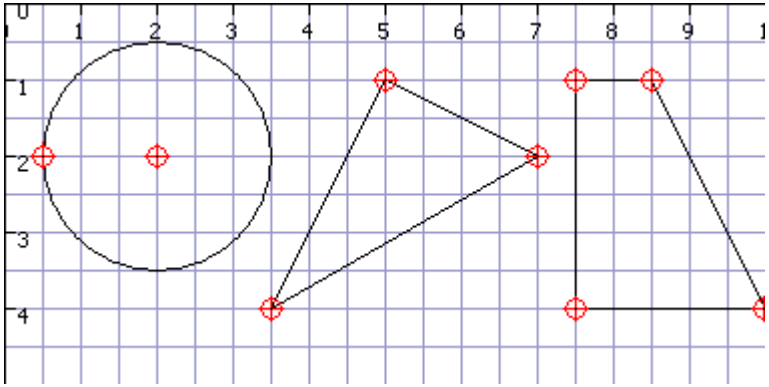


Рис. 1.24. Рисунок, що створено за допогомю клавіші Shift

При створенні креслення часто виникає потреб нарисувати точку чи вершину фігури точно на місці іншої точки або вершини (Рис. 2.25). Для цього достатньо встановити кнопку «Магніт» з піктограмою  у натиснуте положення. У цьому випадку програма перевіряє поруч з якими координатами було натиснуто на ліву кнопку. Тобто якщо координати потрапляють у квадрат зі стороною 20 пікселів та центр якого лежить на шуканій вершині, то змінні X та Y приймають значення що встановлені у цієї вершини.

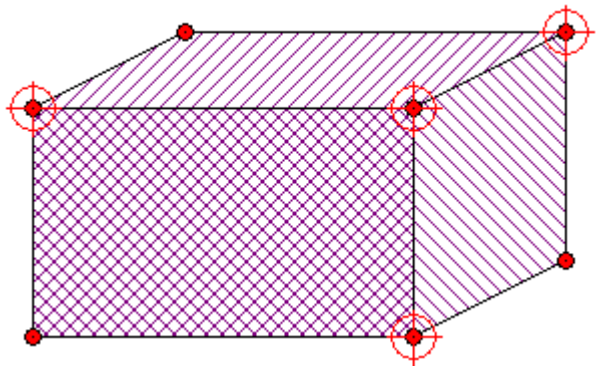


Рис. 1.25. Рисунок створений з використанням функції "Магніт"

Таким чином перед тим як передати координати при потребі вони зазнають вищеописаних перетворень. Дані перетворення було максимально оптимізовано і вони не сповільнюють роботу програми.

1.2.6. Створення головного та контекстного меню

Головне та контекстне меню створимо за допомогою об'єктів MainMenu типу TMainMenu і ряду об'єктів типу TPopupMenu.

MainMemu – не візуальний компонент, тобто місце його розміщення на формі не має ніякого значення для користувача – він все рівно побачить не сам компонент, а тільки маню згенероване ним. Під час проектування властивості Menu форми присвоюється спилка на цей компонент. Основна властивість компоненту – Items. Його заповнення проводиться за допомогою Конструктора Меню, що викликається подвійним клацанням на компоненті MainMenu. В результаті з'явиться вікно, вид якого представлений на Рис. 2.27. В створюваному меню є необхідність створення підменю. Для того щоб його створити потрібно із контекстного меню вибрати команду Create Submenu. Кожен створений пункт меню, тобто кожен елемент властивості Items, є об'єктом типу TManuItem, що володіє своїми властивостями, методами, подіями. Також дане меню має пункти для яких передбачена наявність клавіш швидкого доступу. Для того щоб їх визначити , потрібно відкрити випадний список властивості Shorcut у вікні інспектору об'єктів і вибрати із нього потрібну комбінацію клавіш. Ця комбінація з'явиться в рядку розділу меню.[1, с.211]

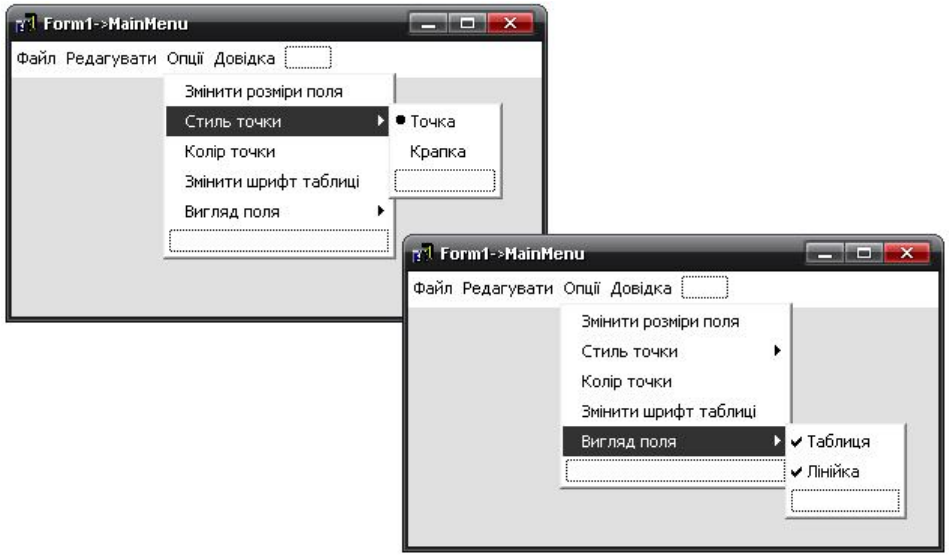


Рис. 1.26. Конструктор головного меню

Властивість Checked, встановлена в true, вказує, що розділі меню буде відображатися маркер прапорця, котрий показує, що даний розділ вибраний. В C++ Builder 6 для розділів меню введено нову властивість – AutoCheck. Якщо гойо встановити в true, то при кожному виборі користувачем даного розділу маркер буде автоматично перемикатися, вказуючи то на вибраний стан, то на відсутність вибору. Ще однією властивістю, що дозволяє вводити маркери в розділи меню є RadioItem. Ця властивість, встановлена в true, визначає, що даний розділ повинен працювати в режимі радіо кнопки спільно з іншими розділами, що маю такий самий GroupIndex. Такі розділи меню знаходяться на вкладці «Опціїї» (Рис 2.27). [1, с.212-213]

Головне меню містить вкладки: «Файл», «Редагувати», «Опції», «Довідка». У вкладці файл розміщені команди, для роботи з файлами: «Створити», «Відкрити», «Зберегти», «Зберегти як», «Імпорт в *.bmp» та «Вихід». Вкладка «Редагувати» включає в себе

основні команди для роботи з зображенням: «Введення координат», «Скопіювати», «Вставити», «Виділити всі об’єкти», «Виділити:», «Видалити виділені об’єкти», «Очистити поле», «Зняти виділення»; вкладка опції містить наступна поля: «Змінити розміри поля», «Стиль точки», «Колір точки», «Змінити шрифт таблиці» та «Вигляд поля». Вкладка «Довідка» містить поле «Інструкція користувача» та «Про програму».

Контекстне меню прив’язують до певних компонентів. Так всі контекстні меню створені в програмі прив’язані до компоненту Image. Воно випливає, якщо під час, коли даний компонент в фокусі, користувач клацне правою кнопкою «миші». Форматування контекстного меню також створюється за допомогою конструктора меню (Рис. 2.28), який викликається подвійним кліком на об’єкті. В основному робота з контекстним меню не відрізняється від роботи з головним вікном.

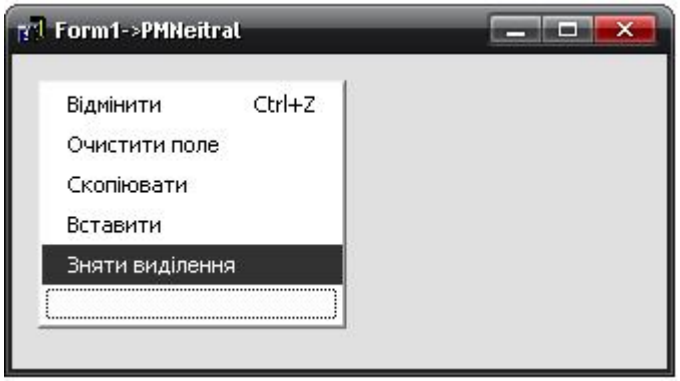


Рис. 1.27. Створення контекстного меню PMNeitral за допомогою конструктора меню

В програмі існують кілька контекстних меню: PMNeitral, PMCircle, PMTriangle, PMRhombus, PMRectang, PMEllips, PMLine, PMCorner, PMLenght. Всі контекстні меню призначені для роботи з певними фігурами, тобто якщо клацнути правою кнопкою над певною фігурою, чи коли виділена якась одна фігура, з’явиться відповідне контекстне меню. PMNeitral контекстне меню, що викликається у

випадку коли користувач виділив кілька об’єктів та у всіх інших випадках. Зміна контекстного меню компоненті відбувається шляхом змінити властивості компоненту Image - PopupMenu у обробнику події OnMouseDown. У випадку коли натиснута права кнопка, програма визначає який об’єкт виділений і скільки, а далі призначає потрібне меню. Основне контекстне меню PMNeitral викликається найчастіше. Це меню випливає у випадку виділення кількох фігур або точки. Поля у меню PMNeitral: «Відмінти», «Очистити поле», «Скопіювати», «Вставити» та «Зняти виділення» (Рис. 2.28). Для роботи з колом потрібно передбачити можливість роботи з хордами, радіусами, діаметрами. Тобто потрібно надати користувачу можливість додавати ці прямі і видаляти їх. Тому в меню кола PMCircle з’явилися такі пункти як «Змінити радіус», «Нарисувати хорду», «Видалити діаметр» то ін (Схема 2.1). Також змінити радіус кола можна за допомогою того ж самого контекстного меню, вибравши пункт «Змінити радіус»(Схема 2.1).

Схема 1.1 Контекстне меню кола

В трикутнику користувач може провести такі прямі як висота, медіана, бісектриса та середня лінія трикутника, кількість яких обмежена. Також в контекстному меню передбачена можливість змінити положення вершини тієї над якою було викликане дане контекстне меню. Схема контекстно меню трикутника подана на Схемі 2.2.

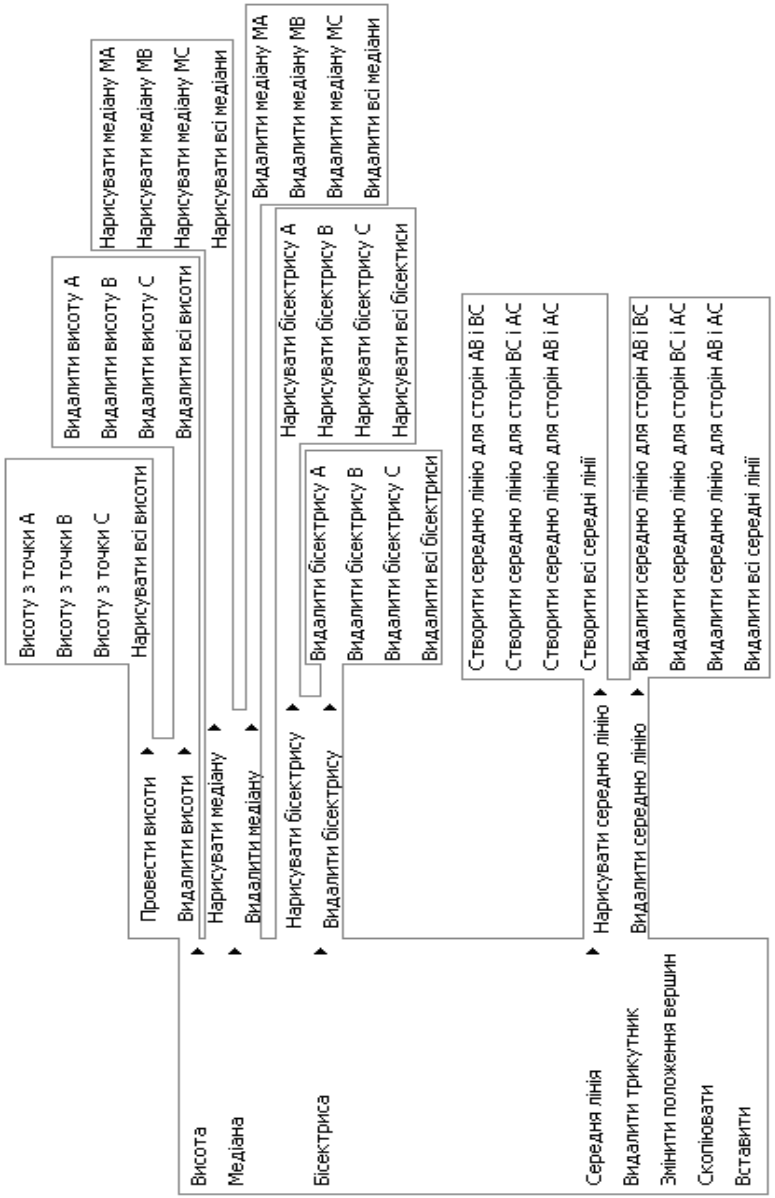


Схема 1.2. Контекстне меню трикутника

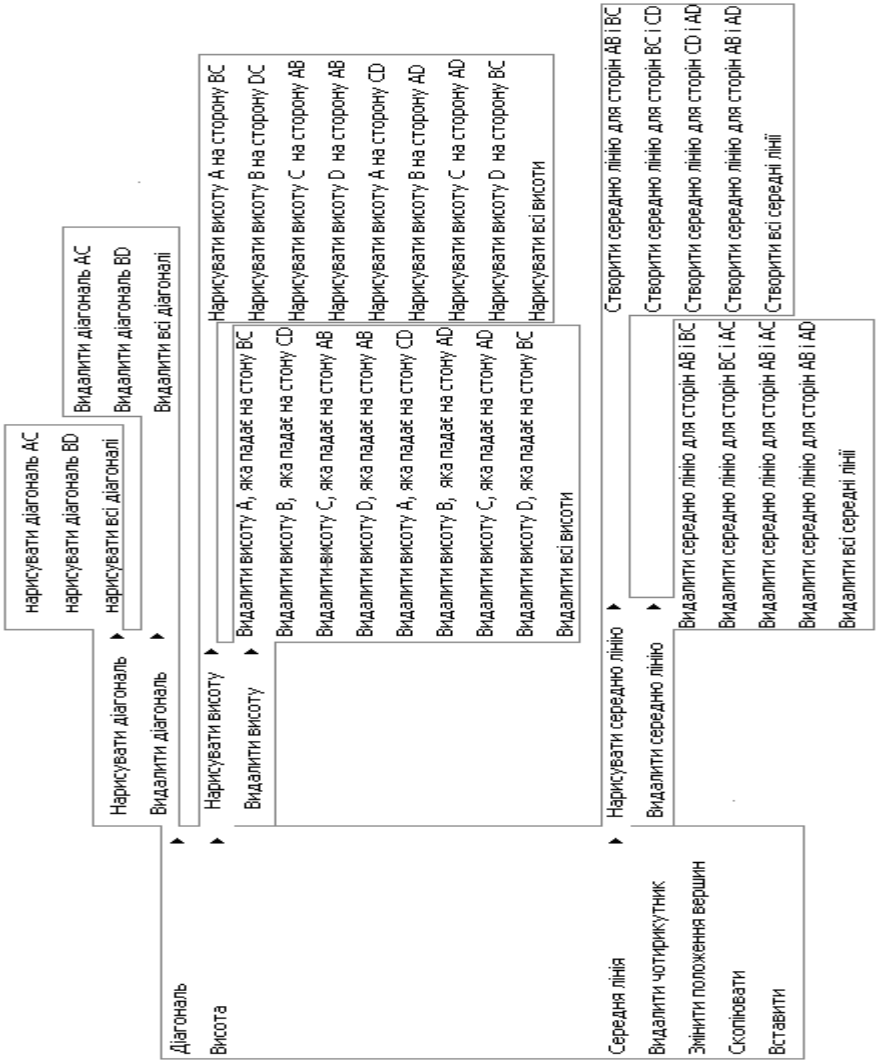


Схема 1.3. Контекстне меню чотирикутника

Для чотирикутника та паралелограма створене одне контекстне меню PMRhombus, оскільки паралелограм являє собою чотирикутник. Як і в трикутнику, для чотирикутників також передбачена можливість змінювати вершини, видаляти та копіювати. Також за допомогою контекстного меню можна додавати висоти та середні лінії, однак, порівняно з трикутником, їх кількість дещо більша. Схема контекстного меню PMRhombus наведена на Схемі 2.3.

Прямокутник також є чотирикутником, однак він не потребує проведення висот чи середніх ліній. Саме тому, для прямокутника було створено спрощене меню PMRectang. Схема контекстного меню прямокутника наведена на Схемі 2.4.

Схема 1.4. Контестне меню прямокутника

PMEllips – контекстне меню еліпса (Схема 2.5). В цьому меню додані такі пункти як «Змінити півосі» та «Півосі». Для того щоб можна було змінити нарисований еліпс призначений пункт «Змінити півосі» та його підпункти. Пункт «Півосі» та його підпункти призначені для рисування горизонтальних та вертикальних ліній, від центру еліпса.

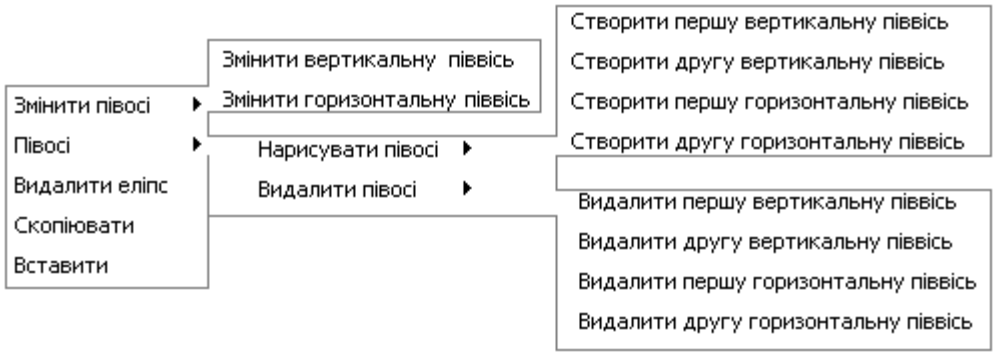


Схема 1.5. Контестне меню еліпса

Контекстні меню для лінії, дуги та відрізка мають назви відповідно PMLine, PMCorner, PMLenght (Рис. 2.34). Контекстне меню PMLine має новий пункт «Змінити кут нахилу прямою». Вибравши цей пункт користувач має можливість змінити нахил прямої шляхом зміни координат другої точки прямої. Для дуги передбачена можливість змінити її радіус, саме для цього у контекстне меню PMCorner був доданий пункт «Змінити радіус дуги». Що ж стосується контекстного меню відрізка – PMLenght, то тут, з'явився новий пункт «Змінити положення кінців відрізка», що є аналогом пункту «Змінити положення вершини», що з'являється в контекстних меню трикутника, чотирикутника і прямокутника.

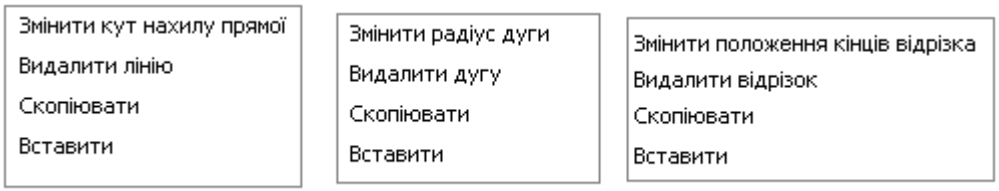


Рис. 1.28 Контекстні меню лінії, дуги та відрізка

Таким чином в програмі створено ряд контекстних меню, що помітно збільшують можливості користувача і надають йому змогу корегувати створені фігури.

1.2.7. Поля введення текстової інформації

В бібліотеці візуальних компонентів C++Builder існує багато компонентів, які дозволяють відображати, вводити і редагувати текстову інформацію. В програмі для введення тексту використовуються компоненти типу TEdit. В таких вікнах текст міститься у властивості Text типу AnsiString. Цю властивість можна встановлювати в процесі проектування або задавати програмно. Вирівнювання та перенесення тексту неможливі.

Вікна редагування оснащені багатьма іншими функціями, властивим більшості редакторі. Наприклад, в них передбачені типові комбінації «гарячих» клавіш: Ctrl-C – копіювання виділеного тексту в буфер обміну Clipboard (команда Copy), Ctrl-X – виріз виділеного тексту в буфер обміну Clipboard (команда Cut), Ctrl-V – вставка тексту з буферу Clipboard в позицію курсору (команда Paste), Ctrl-Z – відміна останньої команди редагування. [1, с.150-151]

Вікна такого типу знаходяться на панелі PLeft (Рис. 2.19). Також такі поля вводу інформації знаходяться у вікні «Введення координат» та «Зміна розмірів поля», що використовуються для введення числової інформації – координат вершин (Рис. 2.30).

Рис. 1.29. Поле для введення числової інформації

Як було сказано, властивість вікон редагування має тип строки AnsiString. При присвоюванні цьому типу числової інформації відбувається її автоматичне перетворення в строку. Для того щоб зчитати число потрібно скористатися функцією StrToInt. Однак, якщо

ведений текст не буде відповідати числу (наприклад, містити недопустимі символи), то виникне помилка. Щоб цього уникнути ці вікна спроектовані таким чином, що користувач просто не може ввести в нього неправильні символи.

Як видно з Рис. 2.30, поруч з полями вводу знаходиться кнопка-лічильник – UpDown. Цей компонент перетворю вікно редагування Edit в компонент, в якому користувач може вибрати ціле число, змінюючи його кнопками зі стрілками. Основна властивість компоненті UpDown – Associative, яка зв'язує кнопки зі стрілками з одним із компонентів, зазвичай Edit.

Властивість AlignButton компоненту UpDown, яке може приймати значення udLeft або udRight, визначає, зліва чи справа від вікна будуть розміщені кнопки. Властивість Orientation, яка може приймати значення udHorizontal або udVertical, визначає, чи знаходяться кнопки по вертикалі (одна під одною Рис. 2.30), чи по горизонталі (одна поруч з іншою).Властивість ArrowKeys визначає, чи будуть управляти компонентом клавіші клавіатури зі стрілками. Властивість Thousands визначає наявність або відсутність розділового пробілу між кожними трьома цифрами розрядів числа, яке вводиться.

Властивості Min і Max компоненту UpDown задають відповідно мінімальне та максимальні значення чисел, властивість Increment задає відповідно приріст числа при натиску на кнопці. Властивість Position визначає поточне значення числа. [1, с.171].

В компонентах типу TEdit властивість ReadOnly рівна false, тобто користувач може водити числа також і з клавіатури. Це зручно, оскільки потрібне число може бути далеко від зазначеного в кроці приросту Increment в UpDown.

1.2.8. Палітра кольорів

Для того щоб користувач міг змінювати колір ліній та колір зафарбовування використаний об'єкт CColorGrid типу TCColorGrid, за допомогою якого користувач може вибрати колір з шістнадцяти основних. Натиснувши лівою клавішею миші на потрібному кольорі користувач вибирає колір ліній, а правою кнопкою колір зафарбовування. Цей компонент знаходиться на панелі PDown (Рис. 2.18),а також у вікні «Колір точки» (Рис. 2.31).

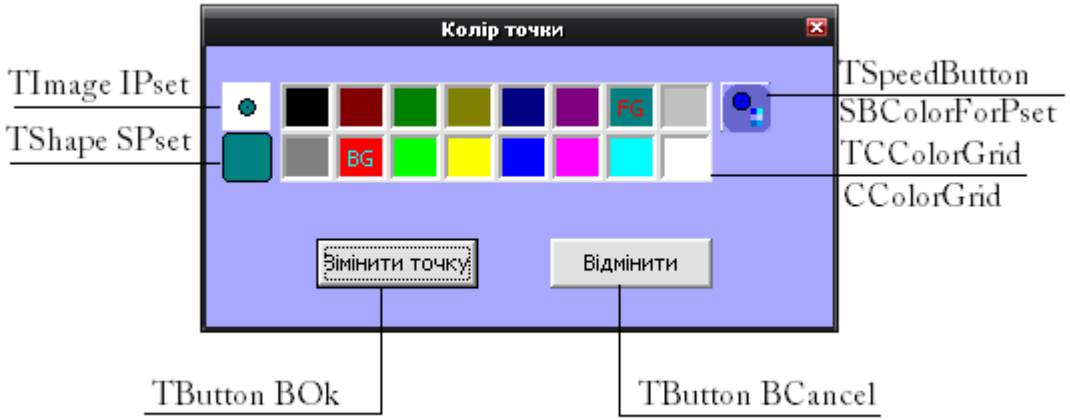


Рис. 1.30 Вікно "Колір точки"

Основними властивостями компоненту CColorGrid є ForegroundColor і BackgroundColor. Саме їхні значення зчитуються під час встановлення кольору. Коли користувач натискає ліву клавішу миші над певним кольором то властивість ForegroundColor приймає значення. А у випадку натиснення правої кнопки значення приймає BackgroundColor. Для того щоб задати спосіб розміщення клітинок кольорів слід вибрати з випливаю чого списку потрібний пункт властивості GridOrdering (на Рис.2.18. та Рис.2.31. GridOrdering має значення go8x2).

Якщо користувач захоче вибрати колір з цілої паліти, то для цього в програмі передбачено дві кнопки на панелі PDown – SBColorForLine та SBColorForPset, для кольору ліній та кольору зафарбовування і кнопка у вікні «Колір точки» - SBColorForPset. Ці кнопки викликають діалог ColorDialog типу TColorDialog, він і забезпечує наявність стандартної розширеної панелі кольорів Windows.

Активні кольори відображаються на об'єктах SPset та SLine класу TSharpe, що мають форму квадрату з заокругленими вершинами (Рис. 2.18, Рис. 2.31.).

1.2.9. Додаткові форми

Основними елементами будь-якого додатку є форма – контейнер в якому розміщені інші візуальні і не візуальні компоненти.

Головна форма відрізняється від інших рядом властивостей. По-перше, саме цій формі передається управління на початку виконання додатку. По-друге, закриття користувачем головної форми означає закінчення роботи програми. По – третє, головна форма так як і інші, може бути спроектована невидимою, але якщо всі інші вікна закрити, то головна форма стає у будь-якому випадку видимою (інакше користувач не зміг би продовжувати роботу з додатком і навіть не зміг би його завершити). [1, с.340]

В потрібний момент форму можна зробити видимою методами Show або ShowModal. Останній метод відкриває форму як модальну. Це означає, що управління передається цій формі і користувач не може передати фокус іншій формі даного додатку до того часу, поки не закриє модальну форму.[1, с.341]. Модальними форми у програмі є форми «Розміри поля» (Рис.2.32.), «Колір точки» (Рис.2.31) та «Про програму».

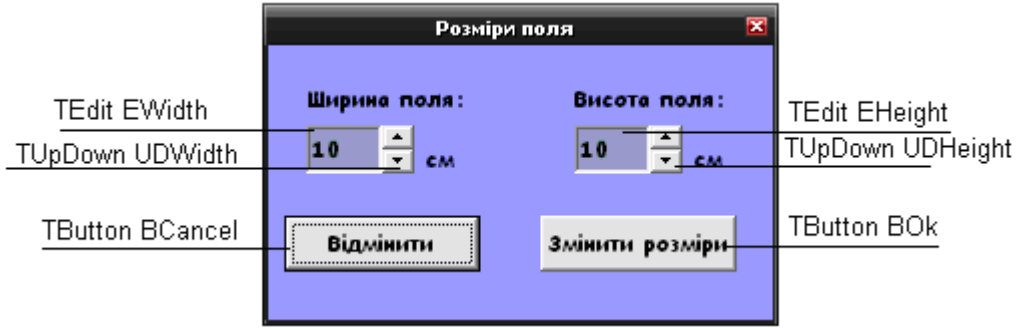


Рис. 1.31. Вікно "Розміри поля"

Поведінка модальної форми визначається її основною властивістю ModalResult. Ця властивість доступна тільки під час виконання додатку. Під час відкриття форми методом ShowModal спочатку властивість ModalResult дорівнює нулю. Як тільки при обробці якихось подій на формі властивості ModalResult буду присвоєно додатне значення, модальна форма закриється. А значення властивості ModalResult можна буде прочитати як результат методом, що повертається методом ShowModal. Таким чином, програма, що викликала модальну форму може знати, що зробив користувач, працюючи з цьою формою, наприклад, на якій кнопці він клацнув.

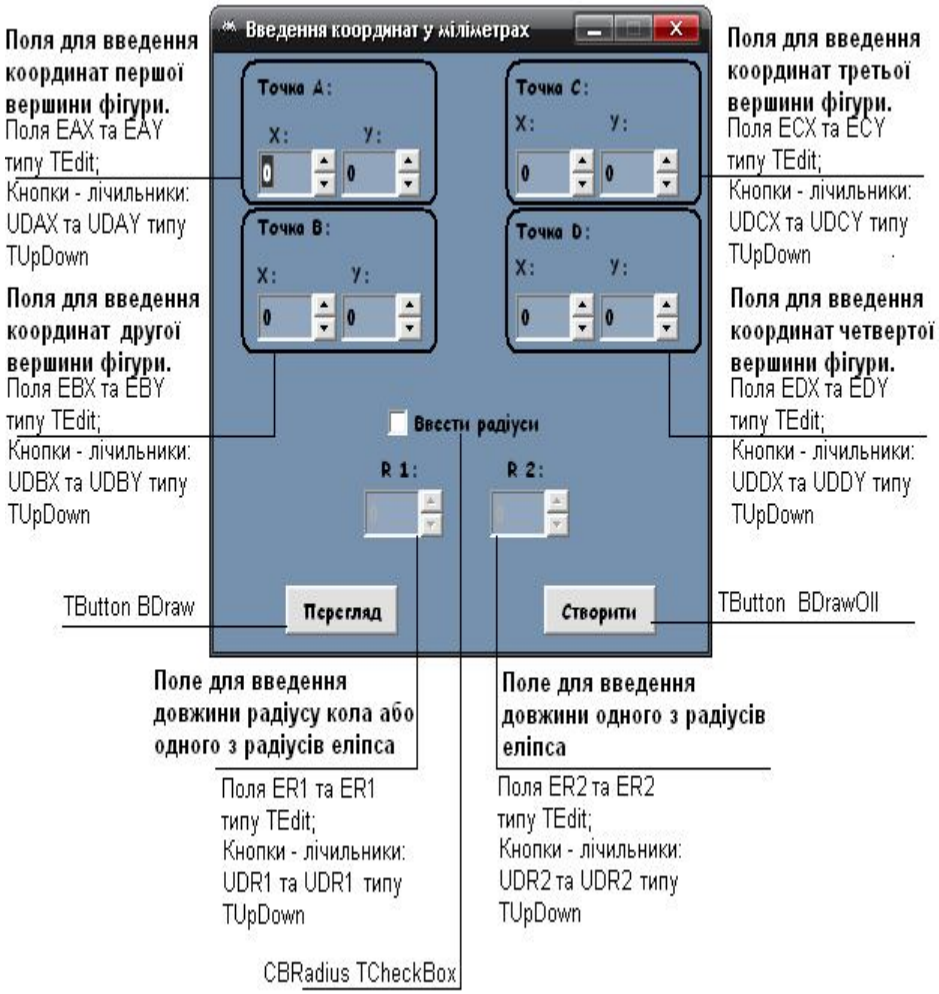
Кнопки типу TButton TBitBtn мають властивість ModalResult, що по замовчуванню дорівнює mrNone. Для конопок, розміщених на модальні йформа, значення цієї властивості можна змінити.[1, с.345]. Так у вікнах «Колір точки» та «Розміри поля» властивість ModalResult кнопок «Змінити точку» і «Змінити розміри» ModalResult має значення – mrOk, а кнопок «Відмінити» - mrCancel.

Методом Show відкривається форма «Введення координат» (Рис. 2.33), та в цей момент поле Enabled компонента Image отримує значення false, тобто стає недоступним. В цьому вікні розташовані полі для введення координат типу TEdit з конопками – лічильниками типу TUpDown. На формі знаходиться індикатор CBRadius. У випадку коли він позначений стають доступними поля для введення

довжини радіусів. Кнопка «Перегляд» рисує фігуру за введи ми координатами коли користувач натискає на неї, і витирає фігуру коли відпускаю цю кнопку. Кнопка «Створити» створює та рисує фігуру за вказаними координатами.

Рис. 1.32. Вікно "Введення координат"

Також у програмі існують вікна, що з'являються біля панелі інструментів в момент коли користувач вибрав інструмент. В



залежності від того який інструмент був вибраний така форма і викликається методом Show. Це форми: FTrianglMode, FRhombusMode, FTrapezoid, FEllipsMode, FArc та FSelect. При бажанні користувач може закрити ці форми, але при наступному виборі інструменти відповідна форма з'явиться поруч з інструментом. Під час створення фігури форми є недоступними. На них розташовані кнопки типу SpeedButton.

Форма FTrianglMode з'являється поруч з інструментом «Трикутник» (Рис.2.34). Кнопки SBrsTriangle, SBprTriangle та SBrbTriangle дозволяють користувачу нарисувати точний відповідно рівносторонній, прямокутний, або рівнобедрений трикутник.

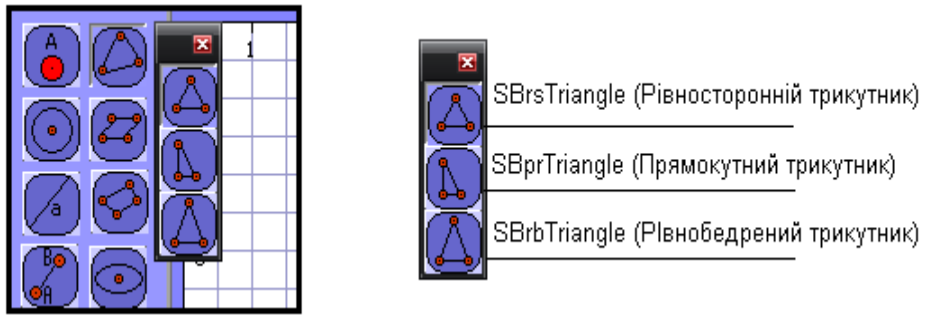


Рис. 1.33. Форма FTrianglMode

Коли користувач вибирає інструмент «Паралелограм» поруч з ним з'являється форма FRhombusMode (Рис.2.35). Якщо користувач натисне на кнопку в вікні, що з'явилося, він зможе нарисувати паралелограм з рівними сторонами.

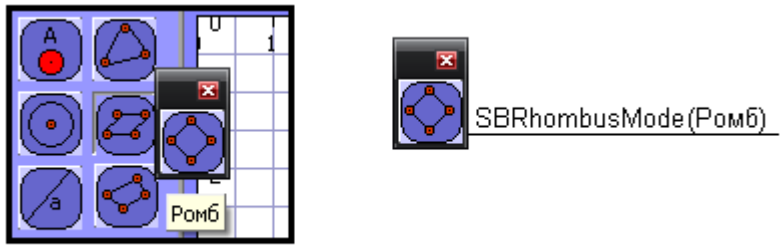


Рис. 1.34. Форма FRhombusMode

При виборі інструменту «чотирикутник» програма викликає форму FTrapezoid (Рис.2.36). Для того щоб користувач міг нарисувати трапецію він повинен натиснути на одну з кнопок на формі: SBrbTrapezoid, SBTrapezoid, SBprTrapezoid, що відповідають: звичайні трапеції, прямокутній та рівнобічні трапеції.

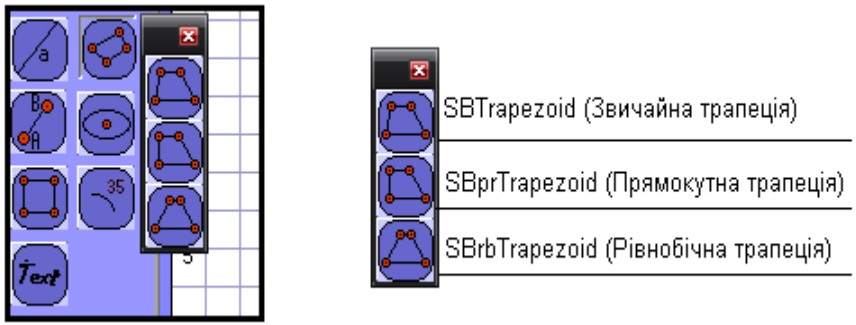


Рис. 1.35. форма FTrapezoid

У випадку коли потрібно створити еліпс задавши його радіуси, слід натиснути на кнопку на формі FEllipsMode, що з'явиться поруч з інструментом «Еліпс» (Рис. 2.37). А от коли потрібно нарисувати дугу вказавши її центр то слід аналогічно на формі FArc (з'явиться поруч з інструментом «дуга») натиснути на кнопку (Рис. 2.38).

Рис. 1.36. форма FEllipsMode

Рис. 1.37. форма FArc

Поруч з інструментом «виділення» з'являється форма FSelect (Рис. 2.39). Натиснувши на кнопку SBSelectRect, при виділенні потрібно вказати прямокутну область. Ті фігури в котрих хоча б одна вершина потрапляє у вказану область виділяться.

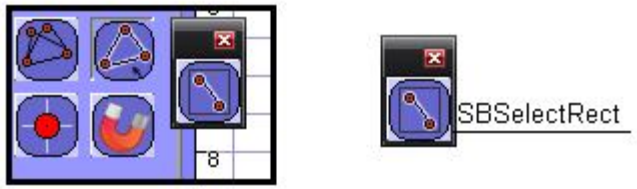


Рис. 1.38. Форма FSelect

Таким чином у програмі є ряд форм що розширює можливості користувача та дозволяють зручно створювати фігури.

1.2.10. Діалоги

В створеній програмі у користувача може виникнути потреба виконати стандартні дії: відривати і зберігати файли, давати атрибути шрифту, вибирати колір палітри. У C++ Builder в бібліотеку були включені компоненти, що реалізують відповідні діалогові вікна.

Основний метод, який виконується звернення до будь – якого діалогу, - Execute. Ця функція відкриває діалогове вікно і, якщо користувач виконав в ньому якийсь вибір, то функція повертає true. При цьому у властивостях компоненту – діалогу запам'ятовується вибір користувача, який можна прочитати і використати у наступних операціях. Якщо користувач в діалозі натиснув кнопку «Отмена» або клавішу «Esc», то функція Execute повертає false. [1, с.238].

Компонент OpenDialog – діалог в «Відкрити файл» і SaveDialog – діалог «Зберегти файл як…» призначені для роботи з файлами. Приклади вікон цих діалогів приведені на Рис. 2.40 и Рис. 2.41.

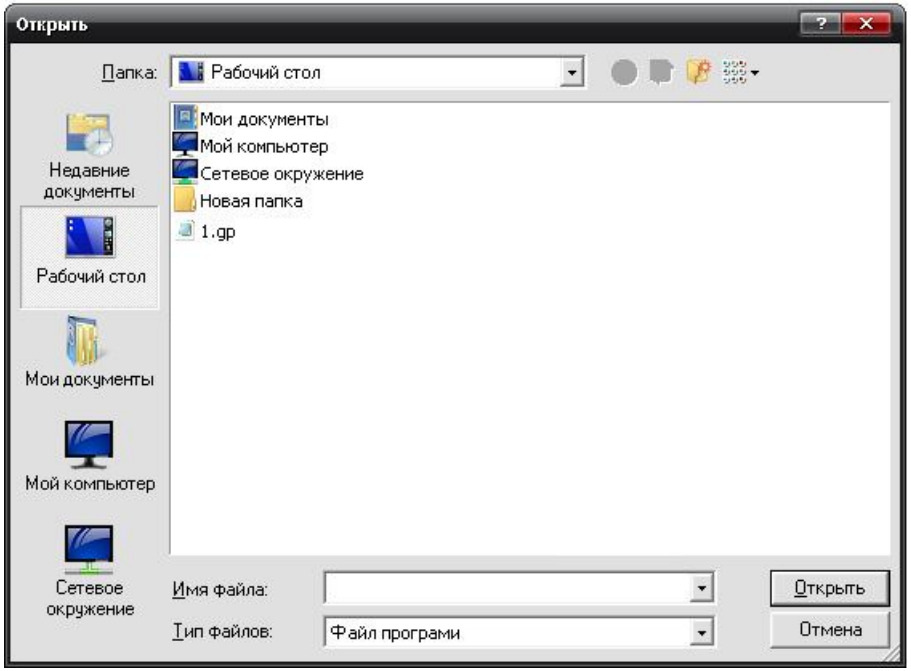


Рис. 1.39. Діалог відкриття файлу

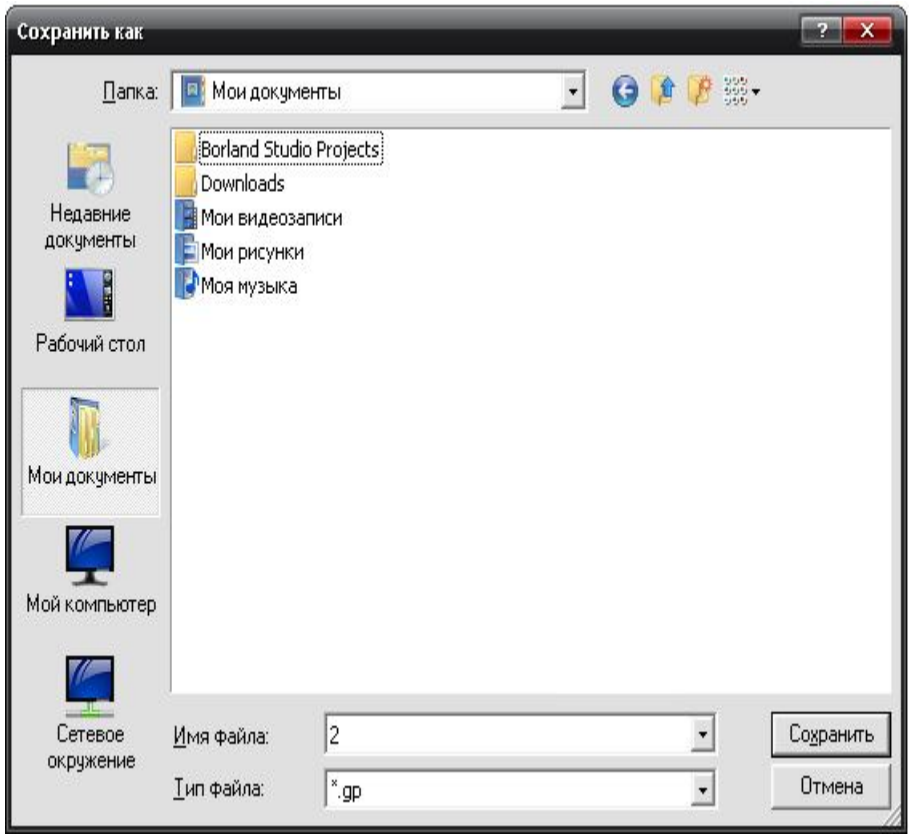


Рис. 1.40. Діалог збереження файлу

Всі властивості цих компонентів однакові, тільки їх сенс дещо відрізняється для відкриття і збереження файлів. Основна властивість, в якій повертається у вигляді строки вибраний користувачем файл, - FileName. Значення цієї властивості можна задавати і перед зверненням до діалогу.

Типи шуканих файлів, що з'являються в діалозі в випадному списку Тип файлу (Рис. 2.40), задаються властивістю Filter. В процесі проектування цю властивість легше задати за допомогою редактора фільтрів, який викликається натисненням кнопки з крапками поруч імені цієї властивості.[1, с.239]

Властивість FilterIndex визначає номер фільтру, яки буде по замовчуванню показаний користувачем в момент відкриття діалогу. Наприклад, у програмі значення FilterIndex = 1, задає по замовчуванню перший фільтр.

Властивість InitalDir визначає початковий діалог, який буде відкриватий в момент початку роботи користувача з діалогом. Якщо значення цієї властивості не задано, то відкривається поточний діалог або той, який був відкритий при останньому зверненні користувача до відповідного діалогу в процесі виконання даного додатку.

Властивість DefaulExt визначає значення розширення файлу по замовчуванню. Якщо значення цієї властивості не задано, користувач повинен вказати в діалозі повне ім'я файлу з розширенням. Якщо ж задати значення DefaulExt, то користувач може писати в діалозі ім'я без розширення. В цьому випадку буде прийняте задане розширення. В програмі властивість DefaulExt = .gp.

Властивість Title дозволяє задавати заголовок діалогового вікна. Якщо ця властивість не задана, вікно відкривається з заголовком, який визначається в системі. [1, с.240]

Властивість Options визначає умови вибору файлу. Множина опції, які можна встановити програмно або під час проектування, є досить великою. Так опції в діалозі SaveDialog яким задано значення true наведено в Таблиці 2.1.

ofOverwritePrompt	У випадку, якщо при збереженні файлу користувач написав ім'я неіснуючого файлу, з'явиться попередження, що файл з таким

	іменем уже існує, і зарошується бажання користувача переписати існуючий файл
ofHideReadOnly	Видаляє із діалогу індикатор Відкрити тільки для читання.
ofPathMustExist	Генерує повідомлення про помилку, якщо користувач вказав в імені файлу неіснуючий каталог
ofEnableSizing	Дозволяє користувачу змінювати розмір діалогового вікна

Таблиця 1.1. Опції діалогу SaveDialog

В діалозі Open Dialog значення true приймають тільки опції ofHideReadOnly та ofEnableSizing.

У діалогів в C++ Builder 5 введена властивість OptionsEx, яке визначає додаткові умови вибору файлу. Поки що в цій властивості тільки одна опція – ofExNoPlacesBar. Вона забороняє появу в діалозі звичайних кнопок Windows. В наведених вище прикладах на Рис. 2.39 та Рис. 2.40 ця опція вимкнута [1, с.341].

В компонентах діалогів відкриття та збереження файлів передбачена можливість обробки ряду додій. Та при створенні програми достатньо було вищенаведених опцій.

Компонент FontDialog викликає діалогове вікно вибору атрибутів шрифту, представлене на Рис. 2.42. В ньому користувач може вибрати ім'я шрифту, його стиль, розмір та інші атрибути. У програмі це вікно користувач бачить коли вибирає пункт головного меню «Змінити шрифт таблиці» та при натисненні кнопки з піктограмою  , для того щоб змінити шрифт назв фігур чи стиль тексту.

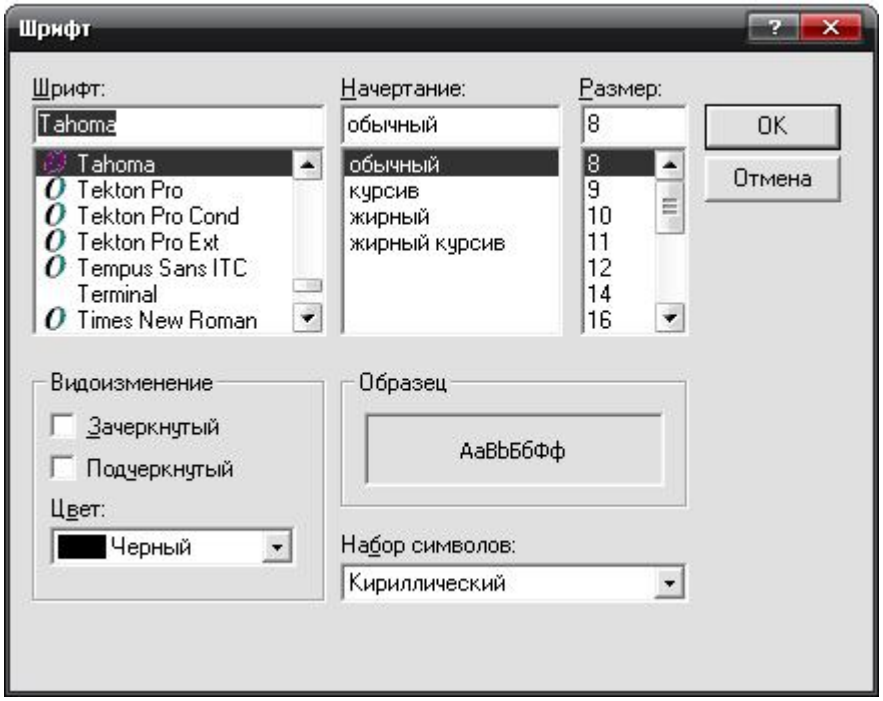


Рис. 1.41. Діалог вибору шрифту

Основна властивість компоненту – Font типу Font, в якому при бажанні можна задати початкові установки атрибутів шрифту і я якому можна прочитати значення атрибутів, вибрані користувачем в процесі діалогу.

Властивості MaxFontSize і MinFontSize встановлюють обмеження на максимальний і мінімальний розмір шрифту. При спробі користувача задати недопустимий розмір йому буде видано попередження виду «Розмір повинен лежати в інтервалі …» і вибір користувача відміняється. Правда це працює не на всіх версіях Windows. Властивості MaxFontSize і MinFontSize діють тільки при увімкнутій опції fbLimitSize.[1, с.246]

Властивість Options містить множину опцій, але всі опції були залишені по замовчуванню.

Компонент ColorDialog викликає діалогове вікно вибору кольору, представлене на Рис. 2.43. В ньому користувач може вибрати колір з базової палітри, бао синтезувати колір.Синтез кольору робиться шляхом переміщення курсору по горизонталі для вибору відтінку і по вертикалі для вибору контрастності. Яскравість регулюється переміщенням по вертикальній шкалі справа від матриці кольорів. Синтезований колір можна додати кнопкою «Добавить в набор» в палітру додаткових кольорів на лівій панелі і використовувати його надалі.

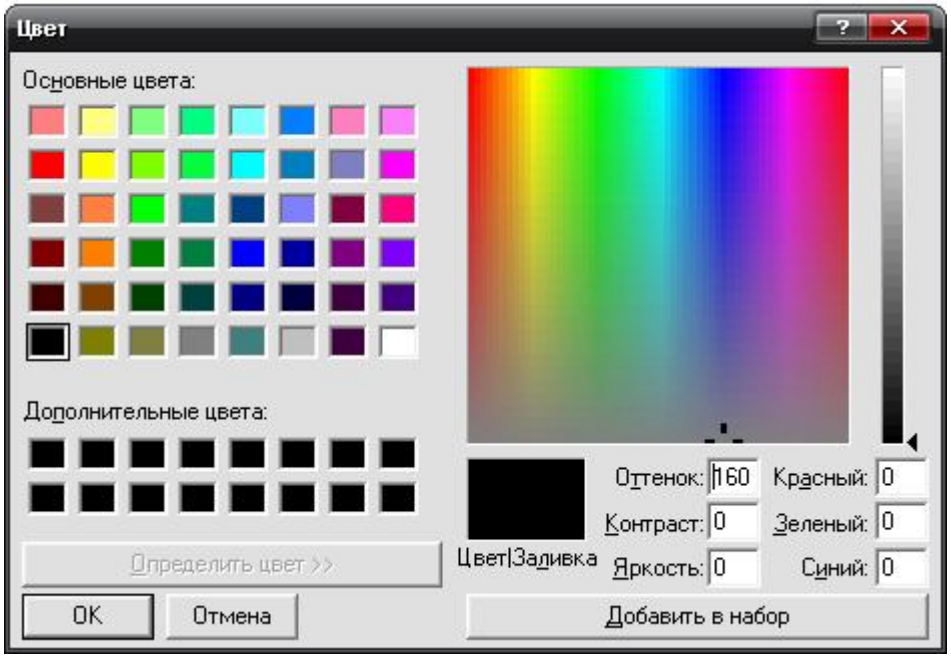


Рис. 1.42. Діалог вибору кольору

Основна властивість компоненту ColorDialog – Color. Ця властивість відповідає тому кольору, який вибрав в діалозі користувач. Властивість CustomColor типу TString дозволяє задати заказані кольори додаткової палітри або прочитати ці кольори, сформовані користувачем в діалозі.[1, с.248]

У випадках закриття програми чи створенні нового документу та ін доцільно вивести діалогове вікно про запит на збереження файлу (Рис. 2.44). Для цього скористаємося функцією MessageBox

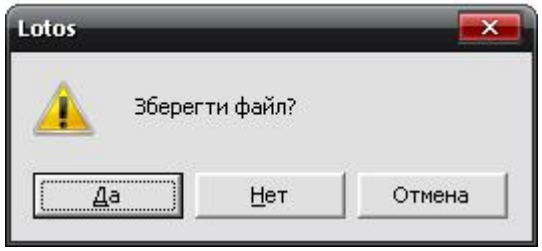


Рис. 1.43. Діалогове вікно, що містить запит на збереження файлу

Функція MessageBox створює, відображає на екрані і оперує вікном повідомлень. Вікно повідомлень містить вказане повідомлення і заголовок, а також будь – яку комбінацію визначених піктограм і командних кнопок. Синтаксис:

```
int MessageBox

(HWND hWnd, LPCTSTR lpText, LPCTSTR lpCaption, UINT
uType );
```

Тут HWND hWnd – дескриптор вікна власника, LPCTSTR lpText – адреса тексту у вікні повідомлення, LPCTSTR lpCaption адреса заголовку у вікні повідомлення і UINT uType – стиль вікна повідомлення.[18]

Для того щоб створити таке повідомлення як повідомлення на Рис. 2.44 слід написати такий рядок коду:

```
int l=Application->MessageBox

("Зберегти
файл?","Lotos",MB_YESNOCANCEL|MB_ICONWARNING);
```

Щоб визначити яку саме кнопку натиснув користувач достатньо зчитати значення змінної l, наприклад:


```
if (l==IDYES)MMSaveClick(this);

else if (l==IDCANCEL) CanClose=false;
```

Тут наведено роботу програми у випадку коли користувач закриває головне вікно. У випадку коли користувач натиснув кнопку «Да» програма виконує так ж дії як і при виборі пункту головного меню «Зберегти як», тобто запускає діалог та зберігає файл. Якщо ж відбулося натиснення кнопки «Отмена» форма не закривають.

1.2.11. Збереження та відкриття файлів

Для роботи у графічному редакторі часто виникає потреба зберегти створене зображення, а потім відкрити і зробити можливо якісь зімни і так далі. Для того щоб користувач міг зберігати і відкривати файли та в подальшому їх редагувати потрібно записати всі дані про створені фігури у файл.

Оскільки кожна фігура характеризується координатами вершин, їх назвою, кольором і стилем ліній та заливки, наявність діагоналей, медіан та ін, всі ці дані потрібно записати у файл. Таким чином у класі Pset передбачені методи для запису та читання полів класу у файл, що успадковуються: void WriteToFile(TFileStream* f),void ReadFromFile(TFileStream* f). Ці методи записують та зчитують основні дані та використовують методи virtual void WritePar(TFileStream*), virtual void ReadPar(TFileStream*). Функції що записують та зчитують дані у потік TFileStream* f були розглянуті у пункті 1.6.

Пункти головного меню «Зберезги» та «Відкрити» запускають діалоги SaveDialog та OpenDialog відповідно (див. п. 2.2.10).

Оскільки програма записує дані у файл в певному порядку то зчитування також повинно відбуватися у такому ж порядку. Для того щоб програма могла відрізняти файли створені нею та коректно їх

відкривати потрібно збур угати файли з певним розширенням, що відрізняється від уже існуючих. Таким чином користувач зберігає дані про створені фігури у файлі з розширенням *.qp і відповідно може відкрити файли тільки з таким же розширенням.

Для того щоб можна було перевести створене зображення у загально відомі формати використано діалог SavePictureDialog та метод SaveToFile:

```
if (SavePictureDialog->Execute())

        Image->Picture->SaveToFile(SavePictureDialog->FileName);
```

Ці дії відбуваються коли користувач вибере пункт головного меню «Імпорт в *.bmp». в результаті він побачить відповідне вікно діалогу та зможе зберегти створене зображення з розширенням bmp.

2. ІНСТРУКЦІЯ КОРИСТУВАЧА

2.1.Основні відомості про програму Lotos

Lotos являє собою інструмент для рисування простих геометричних фігур. Створені рисунки можна зберігати у файлах з розширенням *.gp та відкривати ті ж файли для подальшого редагування. Створені рисунки також можна імпортувати у загальновідомі файли з розширення bmp, з якими в свою чергу можна працювати в інших редакторах.

Одними з існуючих програм, які дозволяють будувати геометричні приміти є Gran2D та FreePlane. Нова програма відрізняється від інших легкістю та зручністю використання, тому навіть простий учень, який має найменші навики роботи з комп'ютером зможе побудувати потрібне йому зображення до задачі не використовуючи лінійок та циркулів і при цьому отримати точне зображення у своєму комп'ютері.

Програма Lotos має основними компонентами як точку, відрізок, пряму та коло, так ще й трикутник, прямокутник, паралелограм, чотирикутник, еліпс та дугу, тобто основні геометричні фігури, що вивчаються у школі. Таким чином програма Lotos може бути практично використана на уроках геометрії під час вивчення всього курсу планіметрії. Оскільки вона має широкий спектр створюваних об'єктів, то вчитель може більш детально зупинитись на вивченні властивостей кожної фігури, а учень наглядно дослідити фігуру і її особливості.

2.2.Створення фігур

Створення рисунку відбувається шляхом рисування курсором на полі для рисування. Для того щоб створити фігуру слід вибрати потрібний інструмент з панелі ліворуч та встановити потрібний колір та стиль ліній і заливки на панелі внизу. Щоб змінити параметри створеної якоїсь фігури досить, виділивши її, налаштувати потрібні параметри на нижній панелі.

Для того щоб створити *точку* потрібно вибрати інструмент «Точка», натиснувши на кнопку з піктограмою 

. Далі навести курсор на потрібні координати і натиснути ліву кнопку «миші».

Щоб нарисувати *коло* слід вибрати інструмент  . Далі навести курсор на місце центру кола і клацнути лівою кнопкою «миші». Для створення кола переміщуйте курсор поки коло не буду мати потрібний радіус і знову натисніть лівою кнопкою миші.

Створити лінію в програмі можна вибравши інструмент «Лінія» 

і одним кліком «миші» вказати першу точку через яку проходитиме пряма, а потім ще одним кліком вказати другу точку через яку проходить пряма.

Відрізок створюється аналогічним чином: потрібно перш за все вибрати інструмент «Відрізок»  та двома кліками «мишки» вказати кінці відрізку

Користувач може нарисувати прямокутник наступним чином: вибрати відповідний інструмент – «Прямокутник»  та двома кліками вказати дві вершини прямокутника по діагоналі.

На панелі інструментів знаходиться інструмент «Трикутник»  , що використовується для створення трикутників. Для того щоб нарисувати трикутник потрібно почергово вказувати вершини трикутника шляхом кліками «мишки». Щоб нарисувати рівносторонній, прямокутний чи рівнобедрений трикутник слід у вікні, що з'являється поруч з кнопкою  , натиснути на кнопку з піктограмою  ,  або  відповідно.

В програмі можна легко нарисувати паралелограм. Для цього, вибравши інструмент «Паралелограм»  , слід вказати на полі три вершини паралелограма, положення четвертої вершини визначає програма. Ромб можна нарисувати натиснувши на кнопку у вікні поруч з інструментом  .

Будь-який чотирикутник можна нарисувати аналогічно трикутнику. Потрібно вибрати інструмент «Чотирикутник»  та почергово вказати чотири вершини. Також за допомогою цього інструмент можна рисувати трапеції, натиснувши на кнопки у вікні поруч з інструментом. Так, для того щоб нарисувати довільну

трапецію, прямокутну або рівносторонню потрібно натиснути на кнопки  ,  та  відповідно та нарисувати фігури

Еліпс можна нарисувати двома шляхами, та у будь-якому випадку спершу слід вибрати інструмент «Еліпс»  . Так, досить вказати прямокутну область в яку вписано еліпс, цент еліпса програма обчистить самостійно. Другий спосіб полягає у тому, що користувач повинен вказати дві півосі еліпса. Для цього потрібно натиснути на кнопку у вікні поруч з цим інструментом -  . Далі користувач повинен нарисувати спочатку вертикальну вісь радіус, а потім вже вказується горизонтальна вісь.

Дугу також можна нарисувати двома способами, вибравши інструмент «Дуга»  . У першому випадку вказуються три точки через які проходить дуга. У другому випадку, натиснувши на кнопку у вікні, що з'явилося поруч,  , вказати спочатку цент дуги а потім точку початку та кінця дуги за часовою стрілкою.

Інструмент «Текст»  призначений для створення тексту на полі. У полі що знаходиться на панелі зліва слід ввести текст а потім на полі вказати положення тексту.

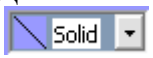
Для того щоб відмінити створення фігури потрібно викликати контекстне меню та вибрати команду «Відмінити».

2.3.Колір стиль ліній та заливки фігур

На нижній панелі знаходяться засоби для задавання таких параметрів фігури як колір, стиль та товщина ліній, спосіб заливки та

її колір. Для того щоб створити фігуру з бажаними параметрами слід перед початком рисування встановити потрібні параметри.

Для встановлення потрібного кольору слід користуватися палітрою кольорів, що містить шістнадцять основних кольорів. Для того щоб встановити колір лінії слід натиснути ліву кнопку «миші», а для встановлення кольору заливки – праву. Для розширення палітри кольорі слід натиснути на кнопки поруч:  та  для встановлення кольору ліній та заливки відповідно. У вікні стандартного діалогу Windows вибору кольорів потрібно вибрати потрібний колір та відтінок і натиснути на кнопку «ОК».

Для встановлення стилю лінії потрібно вибрати із випадного списку 

потрібний стиль:

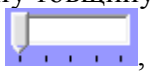
- суцільна лінія

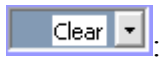
- штрихова лінія

- пунктирна лінія

- штрих-пунктирна лінія

лінія, що чергує два штрихи та два пунктири

Щоб встановити потрібну товщину лінії потрібно встановити у потрібне положення бігунок  , що також знаходиться на нижній панелі. Тут слід зазначити, що при товщині лінії більше за одиницю стиль лінії завжди суцільний.

Стиль заливки також встановлюється вибором потрібного пункту з випадного списку 

:

- суцільна заливка

- відсутність заливки

- заливка горизонтальними лініями

- заливка вертикальними лініями

- заливка діагональними лініями

- заливка діагональними лініями

- заливка сіткою

- заливка хрест на хрест

Щоб змінити колір точок фігури слід вибрати пункт головного меню «Опції»-> «Колір точки». У вікні що відкрилося вибрати з палітри потрібний колір лівою кнопкою «миші». Для того щоб змінити відтінок потрібно натиснути на кнопку  і вибрати потрібний колір. Щоб змінити колір точки на вибраний натисніть на кнопку «Змінити точку» або кнопку «Відмінити» щоб залишити попередній колір точки.

Для зміни кольору точки можна також скористатися і головною палітрою кольорів. Для цього потрібно встановити фляжок біля опції «Змінити колір точки», що знаходиться праворуч на нижній панелі, і правою натиснути правою кнопкою «миші» на потрібному кольорі палітри.

Для того щоб змінити точки фігури на крапки потрібно вибрати пункт головного меню «Опції»-> «Стиль точки» і вибрати потрібний стиль.

Назви вершин фігур можна вводити у відповідні поля на лівій панелі перед створенням фігури. Для того щоб змінити шрифт назв вершин потрібно натиснути на кнопку  , що знаходиться на лівій панелі, і у стандартному діалозі Windows вибрати потрібний шрифт.

Для того щоб змінити якісь з вищенаведених параметрів (колір ліні, стиль заливки, колір та стиль точки, назви вершин та ін.) уже створених фігур слід виділити їх та встановити потрібні параметри.

2.4.Виділення та робота з виділеними об'єктами

Виділити створену одну чи декілька фігур можна кількома способами:

- Втримувати клавішу Ctrl, навести курсор на потрібну фігуру і натиснути ліву кнопку «миші»
- На панелі інструментів натиснути на кнопку  , навести курсор на потрібну фігуру і натиснути ліву кнопку «миші»
- На панелі інструментів натиснути на кнопку  , у вікні що відкрилося поруч натиснути кнопку  , та вказати прямокутну область, так щоб хоча б одна вершина потрапляла в цю область.
- Користуючись головним меню, вибрати команду «Виділити всі об'єкти» чи використати комбінацію клавіш Ctrl + A. У цьому випадку виділяться всі об'єкти, що знаходяться на полі.
- Вибрати пункт головного меню «Виділити…» та вибрати з підменю потрібні фігури. Таким чином можна виділити однотипні фігури, наприклад, виділити тільки трикутники, які знаходяться на полі.

З виділеними фігурами можна виконувати наступні дії:

- переміщувати, шляхом переміщення курсору
- копіювати, вибравши відповідний пункт головного меню

- видалити, вибравши в головному меню команду «Видалити виділені об'єкти» або ж натиснути комбінацію гарячих клавіш Ctrl + Del чи натиснути на кнопку, що знаходиться на верхній панелі, з піктограмою  .
- Змінити наступні параметри:
  - Назву вершин
  - Шрифт назви
  - Стиль точки
  - Колір точки
  - Стиль, товщину та колір лінії
  - Стиль та колір заливки
- Вивести на передній план

Для того щоб зняти виділення потрібно вибрати пункт «Зняти виділення» з контекстного меню чи головного меню у вкладці «Редагувати».

2.5.Зміна положень вершин

У таких фігур, як відрізок, прямокутник, паралелограм та чотирикутник, передбачена можливість змінити положення вершини після того коли фігура вже була створена. Змінити положення потрібної вершини можна наступним способами:

- Встановити у натиснуте положення кнопку з піктограмою  і натиснути лівою кнопкою «миші» на тій вершині, що потребує переміщення
- Утримувати кнопку Alt під час натиску на ліву кнопку «миші» на вершині
- Викликати контекстне меню, натиснувши праву кнопку над вершиною, яку потрібно перемістити, і вибрати пункт «Змінити положення вершини»

2.6.Кнопка «Магніт» та Shift

Для того щоб вказані координати лежали на перетині ліній сітки потрібно при рисуванні тримати клавішу Shift або встановити у натиснуте положення кнопку з піктограмою  , що знаходиться на лівій панелі.

Якщо ж виникає потреба нарисувати одну точку (вершина) точно та місці іншої точки (вершини) можна скористатися функцією «Магніт». Для цього під час рисування потрібно вставити у натиснуте положення кнопку  , яка лежить на панелі зліва.

2.7.Робота з фігурами

У програмі для кожної з фігур передбачена своє контекстне меню, свої особливості роботи, наприклад, в прямокутнику є можливість проведення діагоналей, а у кола – радіусів. Таким чином кожна фігура має певний перелік індивідуальних дій.

Індивідуальне контекстне меню викликається у випадку коли виділена тільки одна фігура, або ж жодна з фігур не виділена та контекстне меню було викликане над якоюсь одною фігурою. В загальному, за допомогою індивідуального контекстного меню можна змінювати видаляти фігуру, змінювати такі дані як положення вершин чи радіус, рисувати додаткові лінії на зразок радіуса, висоти, діагоналі та ін.

Що стосується додаткових ліній то тут слід зазначити, що стиль таких лінії буде такий як встановлений на нижній панелі. Для того щоб стиль створюваної ліній був такий самий як і у фігури, що містить цю лінію, потрібно встановити фляжок біля опції «Стиль додаткових ліній батьківської фігури», що знаходиться на нижній панелі. У випадку коли потрібно встановити власний стиль такої ліній і при цьому не змінити колір виділеної фігури, достатньо встановити

фляжок на опції «Не змінювати стиль виділеної фігури», яка також знаходиться на панелі внизу, та встановити потрібний стиль колі чи товщину лінії.

2.7.1.Робота з колом

В контекстному меню кола передбачені такі пункти(Схема 2.1):

- Радіус
- Діаметр
- Хорда
- Змінити радіус
- Видалити коло
- Скопіювати (Ctrl + C)
- Вставити (Ctrl + V)

Призначенням останніх трьох пунктів є зрозумілим. Пункт меню «Змінити радіус» призначений для того щоб змінити радіус уже створеного кола. Таким чином, вибравши цей пункт меню, потрібно курсором вказати новий радіус.

Для того щоб нарисувати у кола радіус чи діаметр, потрібно вибрати пункт «Радіус» -> «Нарисувати радіус» або «Діаметр» -> «Нарисувати діаметр», переміщуючи «мишкою», нарисувати потрібний радіус (діаметр) і натиснули ліву кнопку коли буде створене потрібне зображення. Таким чином можна створити безмежну кількість радіусів (діаметрів). Видалити радіус (діаметр) можна насипним чином: вибрати пункт «Радіус» -> «Видалити радіус» («Діаметр» -> «Видалити діаметр»), але таким чином видалиться останній нарисований радіус (діаметр). Щоб видалити всі радіуси чи діаметри потрібно також звернутися контекстного меню: «Радіус» -> «Видалити всі радіус» або «Діаметр» -> «Видалити всі діаметри».

Аналогічним чином можна створити і хорди, тільки, вибравши пункт «Хорда» -> «Нарисувати хорду», потрібно вказати курсором дві точки. Це точки через які проходить пряма на якій лежить хорда. Видалити останню нарисовану хорду можна скориставшись командою «Хорда» -> «Видалити хорду», а всі ходи кола – «Хорда» -> «Видалити всі хорди».

2.7.2.Робота з прямою, відрізком та прямокутником

Контекст не меню прямої дозволяє виконувати наступні дії х прямою:

- Видаляти
- Копіювати
- Змінювати кут нахилу

За допомогою пункту «Змінити кут нахилу» можна змінити уже створену пряму. Вибравши цей пункт потрібно за допомогою «мишки» вказати новий вигляд прямої, натиснувши на ліву кнопку.

Відрізок створюється аналогічно з прямою. Його контекст не меню майже не відрізняється від контекстного меню прямої, та замість пункту «Змінити кут нахилу» тут пункт «Змінити положення вершин відрізка»(Рис 2.29). Для того щоб скористатися цим пунктом потрібно викликати контекстне меню над одним з кінців відрізку. Вибравши цей пункт потрібно вказати нове положення того кінця відрізку над яким було викликане контекстне меню

Як і пряму та відрізок, прямокутник також можна видаляти та копіювати. Його контекстне меню має два нові пункти: «Змінити положення вершин» та «Діагональ» (Схема 2.4). Пункт «Змінити положення вершин» призначений для зміни координат тієї вершини над якою було викликане контекстне меню. За допомогою команди «Діагональ» -> «Нарисувати діагональ» -> … можна нарисувати одну або обидві діагоналі прямокутника.

2.7.3.Робота з трикутником

Контекстне меню трикутника (Схема 2.2) дозволяє виконувати такі дії:

- Проводити висоти
- Проводити медіани
- Проводити бісектриси
- Проводити середні лінії
- Змінити положення вершин
- Скопіювати
- Видалити

Для того щоб провести висоту, медіану, бісектрису чи середню лінію у створеному трикутнику, потрібно, викликавши його контекстне меню, вибрати один з пунктів: «Висота»->«Провести висоти», «Медіана»->«Нарисувати медіану», «Бісектриса»->«Нарисувати бісектрису», «Середня лінія» ->«Нарисувати середню лінію», та вибрати відповідний підпункт. Таким чином можна опустити висоту чи іншу пряму з будь-якої вершини. Для того щоб одрізу провести всі три слід вибрати підпункт «Нарисувати всі висоти». Аналогічним чином можна провести всі бісектриси, медіани та середні лінії.

Щоб видалити висоту, медіану, бісектрису чи середню лінію трикутника, слід скористатися одним з пунктів: «Висота»->«Видалити висоти», «Медіана»->«Видалити медіану», «Бісектриса»->« Видалити бісектрису», «Середня лінія» ->« Видалити середню лінію», та вибрати потрібний підпункт. Видалити одразу всі, наприклад, медіани можна користуючись підпунктом «Видалити всі медіани». Аналогічним чином видаляються висоти, бісектриси та середні лінії.

2.7.4. Робота з паралелограмом та чотирикутником

Контекст меню паралелограма та чотирикутника (Схема 2.4) ідентичні, тобто дії, що можна виконувати з паралелограмом, можна виконувати і з чотирикутником:

- Нарисувати діагоналі
- Нарисувати висоти
- Нарисувати середні лінії
- Змінювати положення вершин
- Видаляти
- Копіювати

У кожному паралелограмі чи чотирикутнику існує дві діагоналі та вісім висот. Для того щоб провести діагональ потрібно вибрати пункт контекстного меню «Діагональ»-> «Нарисувати діагональ» та вибрати потрібний підпункт. Аналогічно рисуються висоти та середні лінії.

Для того щоб видалити діагональ, висоту чи середню лінію, слід викликати контекст не меню та вибрати один із пунктів: «Діагональ»->«Видалити діагональ»,«Висота»->«Видалити висоти», «Середня лінія» ->« Видалити середню лінію», та вибрати потрібний пункт.

2.7.5.Робота з еліпсом та дугою

В контекстному меню еліпса (Схема 2.4) передбачені такі дії з еліпсом:

- Змінити півосі
- Нарисувати півосі
- Видалити еліпс
- Скопіювати

Щоб змінити уже створений еліпс можна скористатися однією з команд контекстного меню: «Змінити півосі» -> «Змінити

горизонтальну півосі», «Змінити півосі» -> «Змінити вертикальну піввісь». Далі потрібно курсором вказати потрібні розміри півосей еліпса.

Для того щоб нарисувати півосі у еліпсі слід скористатися пунктом контекстного меню «Півосі» -> «Нарисувати півосі». А для того щоб видалити нарисовані півосі в контекстному меню існує пункт «Півосі» -> «Видали півосі».

Дії, що передбачені для дуги:

- Змінити радіус дуги
- Видалити
- Скопіювати

Єдиний спосіб змінити створену дугу, крім переміщення, це змінити її радіус за допомогою пункту контекстного меню «Змінити радіус дуги». Вибравши цей пункт, потрібно курсором сформувати новий вигляд дуги.

2.8.Масштабування

Програма Lotos підтримує масштабування без втрати якості рисунку, тобто збільшення або зменшення його зі збереженням пропорцій (Рис. 3.1). Можливі масштаби: 1:0,5 - 1:4 з кроком 0,5. Для збільшення чи зменшення масштабу можна використовувати ролик «миші». Також для збільшення масштабу призначена кнопка  , а для зменшення -  . Кожен натиск на ці кнопки збільшує або зменшує масштаб на 0,5.

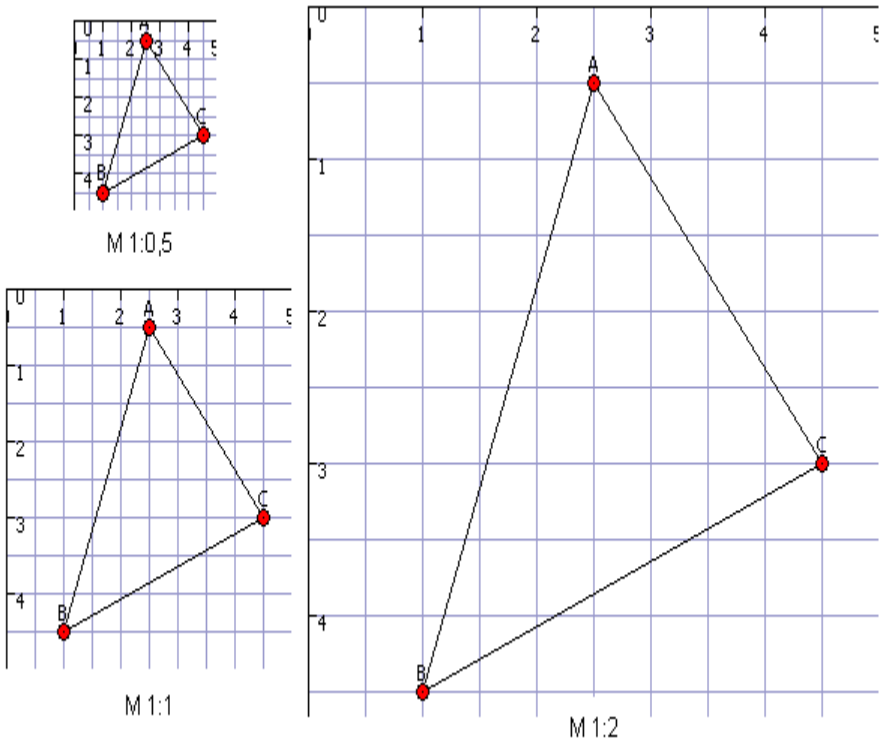


Рис. 1.44. Масштабування

Слід також зазначити, що при імпорті рисунку у формат bmp масштаб буде зберігати поточне значення.

2.9.Опції програми

За допомогою вкладки головного меню можна змінити такі параметри:

- Розміри поля
- Стиль точки
- Колір точки

- Шрифт лінійки
- Вигляд заднього фону

Для того щоб змінити розміри поля потрібно вибрати пункт головного меню «Змінити розміри поля», вказати потрібну ширину та висоту поля в сантиметрах у вікні, що відкрилося, та натиснути кнопку «Змінити розміри».

Стиль точки можна змірити вибравши потрібний підпункт пункту «Стиль точки». Можна вибрати з двох стилів: «Крапка» та «Точка». Напроти потрібного стилю слід встановити флажок.

Колір точки можна змірити у вікні «Зміна кольору точки», яке відкривається за допомогою пункту «Колір точки». У цьому вікні слід вибрати колір або з стандартної палітри кольорів, або ж з розширеної палітри, що викликається за допомогою кнопки  , що знаходиться поруч. Вибравши потрібний колір потрібний натиснути на кнопку «Змінити точку».

Для того щоб змінити шрифт лінійки на задньому фоні потрібно вибрати пункт меню «Змінити шрифт лінійки». У вікні, що відрилося встановити потрібний шрифт та натиснути кнопку «ОК».

Щоб встановити потрібний вигляд фону слід скористатися пунктом «Вигляд поля» та поставити фляжок біля потрібного підпункту. Пункт «Вигляд поля» має два підпункти: «Сітка», «Лінійка». Таким чином можливі чотири вигляди заднього фону (Рис. 2.21., Рис.2.22, Рис. 2.23):

- Тільки сітка
- Тільки лінійка
- Сітка з лінійкою
- Чистий аркуш

2.10.Введення координат вручну

Для введення координат точок та вершин фігур з клавіатури призначене вікно «Введення координат (у міліметрах)», що з'являється після вибору пункту головного меню «Введення координат», що знаходиться у вкладці «Редагувати».

Щоб ввести координати потрібної фігури потрібно:

- Вибрати потрібний інструмент
- Відкрити вікно «Введення координат (у міліметрах)»
- Ввести координати по осі OX та OY всіх вершин у міліметрах
- Натиснути кнопку «Створити»

Для введення координат першої вершини фігури потрібно вводити координати в поля позначені «Точка А», для введення другої вершини – «Точка В» і тд. Наприклад для введення координат для точки (Рис. 3.2), в поле «x» Точки А слід ввести 20, а в поле «y» - 35. Щоб нарисувати трикутник, що наведений на Рис. 3.2 потрібно вказати такі дані:

- Точка А
  - x=10, y=40
- Точка В
  - x=15, y=5
- Точка С
  - x=30, y=35

Для створення паралелограма (Рис. 3.2) також слід ввести три точки, оскільки для побудови паралелограма їх достатньо:

- Точка А
  - x=35, y=40
- Точка В
  - x=60, y=5
- Точка С
  - x=88, y=5

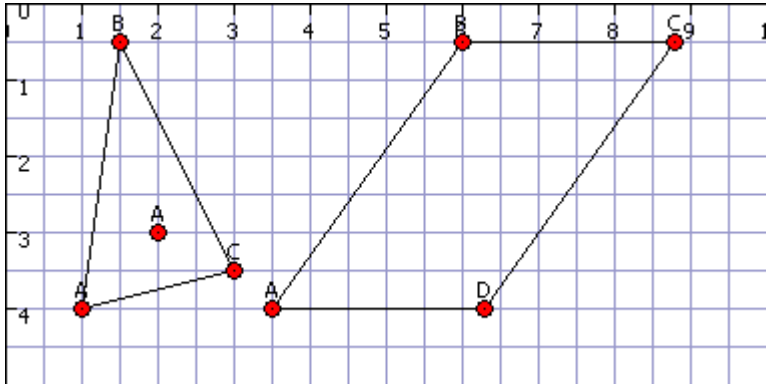


Рис. 1.45. Фігури створені введенням координат з клавіатури

Після введення координат можна переглянути який вигляд буде мати фігура з введеними координатами натиснувши і утримуючи кнопку «Переглянути». Якщо фігура має задовільний вигляд то можна натискати на кнопку «Створити», інакше потрібно скорегувати координати та повторити дії.

Слід також зазначити, що поки вікно «Введення координат (у міліметрах)» знаходиться у відкритому всі панелі є доступними але поле для рисування недоступне для рисування.

2.11.Редагування рисунку

Виділені фігури можна копіювати та вставляти в документ. Скопіювати одну чи декілька фігур можна одним з способів, попередньо виділивши їх:

- Вибрати пункт «Скопіювати» з контекстного меню
- Вибрати пункт «Скопіювати» вкладки «Редагувати» головного меню
- Натиснути на кнопку  .
- Натиснути комбінацію клавіш Ctrl +C

Вставити скопійовані фігури можна також трьома способами:

- Вибрати пункт «Вставити» з контекстного меню
- Вибрати пункт «Вставити» вкладки «Редагувати» головного меню
- Натиснути на кнопку  .
- Натиснути комбінацію клавіш Ctrl +V

Створені фігури можна видаляти, для цього потрібно виділи непотрібні фігури та виконати одну із дій:

- Вибрати пункт контекстного меню «Видалити»
- Вибрати пункт «Видалити» вкладки «Редагувати» головного меню
- Натиснути на кнопку  .
- Натиснути комбінацію клавіш Ctrl + Del

Також можна зразу очистити поле виконавши одну із дій:

- Вибрати пункт контекстного меню «Очистити поле»
- Вибрати пункт «Очистити поле» вкладки «Редагувати» головного меню
- Натиснути на кнопку  .
- Натиснути комбінацію клавіш Shift + Del.

У випадку коли одна фігура нарисова над іншою і є потреба вивести на передній план то потрібно вибрати пункт «На передні план», що знаходиться у вкладки «Редагувати» головного меню, або натиснути на кнопку  , попередньо виділивши фігури, що потрібно перерисувати.

2.12.Збереження зображення

Для того щоб зберегти створений рисунок потрібно вибрати пункти головного меню «Зберегти» або «Зберегти як», або ж натиснути на кнопку  . Таким чином можна зберегти дані у файлі з розширенням .qp. Такі файли, у подальшому, можна відкривати та редагувати тільки в даній програмі. Відкрити файл з розширенням .qp можна або вибравши пункт головного меню «Відкрити» або натиснути на кнопку  .

Створений рисунок можна імпортувати в формат BMP користуючись командою головного меню «Імпорт в *.bmp» або натиснувши на кнопку  . Відкрити і редагувати такий файл в програмі Lotos неможливо. Однак файл з розширенням .bmp може бути відкритий, відредагований або роздрукований за допомогою іншого графічного редактора, який є зручним для користувача.

Створити новий рисунок можна за допомогою пункт головного меню «Створити» або кнопки  .

3.ПРИКЛАДИ ВИКОРИСТАННЯ ПРОГРАМИ

Розглянемо кілька прикладів задач в яких можна використати програму Lotos для створення рисунків.

Приклад 1. Створення рисунків для наочного зображення властивостей.

В довільному трикутнику квадрат сторони, що лежить проти гострого кута, дорівнює сумі квадратів двох інших сторін без подвійного добутку однієї з цих сторін і проекції на неї другої сторони (Рис. 4.1) [19, с.332]:

$$c^2 = a^2 + b^2 - 2ab_a = a^2 + b^2 - 2ba_b.$$

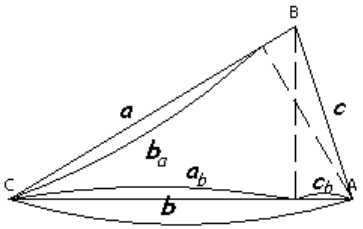


Рис. 0.1

Приклад 2. Ілюстрація до визначень.

Бісектрисою трикутника називається відрізок бісектриси будь – якого кута цього трикутника від вершини до перетину її з протилежною стороною. Всі три бісектриси (за кількістю кутів) трикутника перетинаються в одній точці, яка завжди лежить всередині трикутника і є центром вписаного кола (Рис.4.2) [19,с.337].

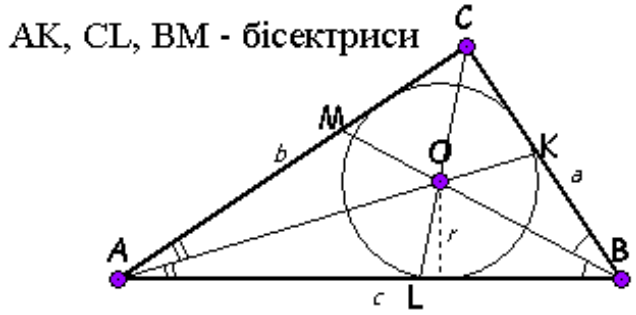


Рис. 0.2

Приклад 3. Рисунок до задачі.

Непаралельні сторони трапеції продовжено до взаємного перетину і через одержану точку проведено пряму паралельно до основ трапеції (Рис. 4.3). Знайти відрізок її, обмежений продовженими діагоналями, якщо основи трапеції дорівнюють DE=a, MN=b (a>b) [19, с.339].

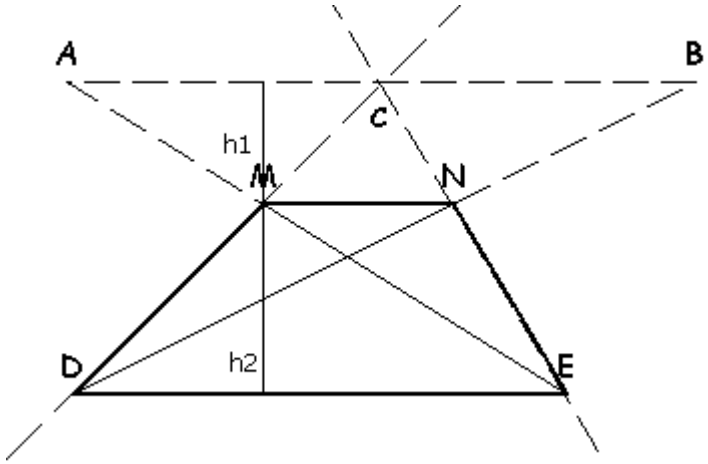


Рис. 0.3

Приклад 4. Розв’язування задачі на побудову.

Через дану точку всередині кола провести хорду, яка ділиться в даній точці на рівні частини (Рис 4.4) [20, с.60].

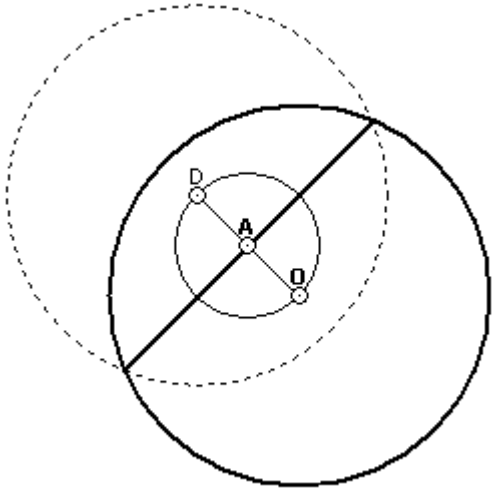


Рис. 0.4

Алгоритм розв’язку задачі у програмі Lotos:

1. Маємо коло з центром в точці О та точку А. Вибираємо інструмент «Коло» (попередньо натискаємо на кнопку «магніт» для того щоб центр кола точно співпав з точкою) і рисуємо коло з центром в точці А, так щоб на отриманому колі лежала точка О.
2. Виділяємо коло з центром А і вибравши з контекстного меню пункт «Нарисувати діаметр» і наводимо курсор на точку О.
3. Будуємо точку D, діаметрально протилежну точці О.
4. Виділяємо коло з центром у точці О і копіюємо його. Копію переміщаємо так щоб центр збігався з точкою D.
5. Виділяємо коло з центром в точці О, вибираємо пункт контекстного меню «Нарисувати хорду» і рисуємо хорду так щоб початок і кінець хорди співпадав з точками перетину кіл.

Отже, програма Lotos може бути використана як вчителем для ілюстрації задач чи теорем, демонстрації властивостей фігур їх розміри та ін, так і учнем для створення ілюстрацій до задач. Також програму можна використовувати при вивченні задач на побудову, тільки при цьому не потрібно використовувати ні лінійку ні циркуль, причому отримане зображення буде динамічним і точним.

ВИСНОВКИ

Планіметрія – це важливий розділ геометрії, що вивчає фігури на площині. Правильне засвоєння планіметрії учнями є надзвичайно важливим, оскільки під час її вивчення учень здобуває «фундамент» своїх майбутніх знань з геометрії та й математики в цілому, а також розвиває ряд корисних навиків. Велику роль в засвоєнні матеріалу планіметрії має наочне зображення фігур, їх взаємне розміщення, розміри сторін, кутів, висот, бісектрис, медіан, хорд та ін. У сучасному світі, який важко уявити без комп'ютерів, для побудов таких простих фігур як відрізок, трикутник, коло, паралелограм та інших фігур широко використовується різне програмне забезпечення.

Одними з існуючих програм, які дозволяють будувати геометричні примітиви, є Gran 2D та FreePlane. Нова програма LOTOS, що була розглянута в даній роботі, відрізняється від інших легкістю та зручністю використання, тому навіть простий учень, який має найменші навики роботи з комп'ютером зможе побудувати потрібне йому зображення до задачі не використовуючи лінійок та циркулів і при цьому отримати точне зображення у своєму комп'ютері.

Якщо програма Gran 2D та її аналоги основними компонентами мають точку, пряму (промінь, відрізок і т.і.) та коло, то програма LOTOS має основними компонентами як точку, відрізок, пряму та коло, так ще й трикутник, прямокутник, паралелограм, чотирикутник, еліпс та дугу, тобто основні геометричні фігури, що вивчаються у школі. Таким чином програма Lotos може бути практично використана на уроках геометрії під час вивчення всього курсу планіметрії. Оскільки нова програма має більш широкий спектр створюваних об'єктів, то користувач вчитель може більш детально зупинитись на вивченні властивостей кожної фігури, а учень наглядно дослідити фігуру і її особливості. Для кожного об'єкту передбачено своє контекстне меню, що дозволяє виконувати

індивідуальні операції з кожною фігурою в залежності від її особливостей.

Слід відзначити зовсім різний інтерфейс програми LOTOS, який суттєво відрізняється від інтерфейсу Gran2D. При розробці інтерфейсу багато уваги виділялося зручності, легкості використання програми, для того щоб користувач на інтуїтивному рівні міг зробити елементарні дії. Також зовнішній вигляд програми спрямований викликати інтерес користувача та забезпечити комфортну роботу під час використання програми LOTOS.

Таким чином, програма Gran2D є чудовим засобом для створення геометричний зображень, однак є досить складною для пересічного школяра та вимагає певного досвіду роботи в ній. А от нова програма LOTOS відрізняється перш за все легкістю та зручністю. Це є дуже важливим для школяра, оскільки він часто має мету швидко і легко отримати результат.

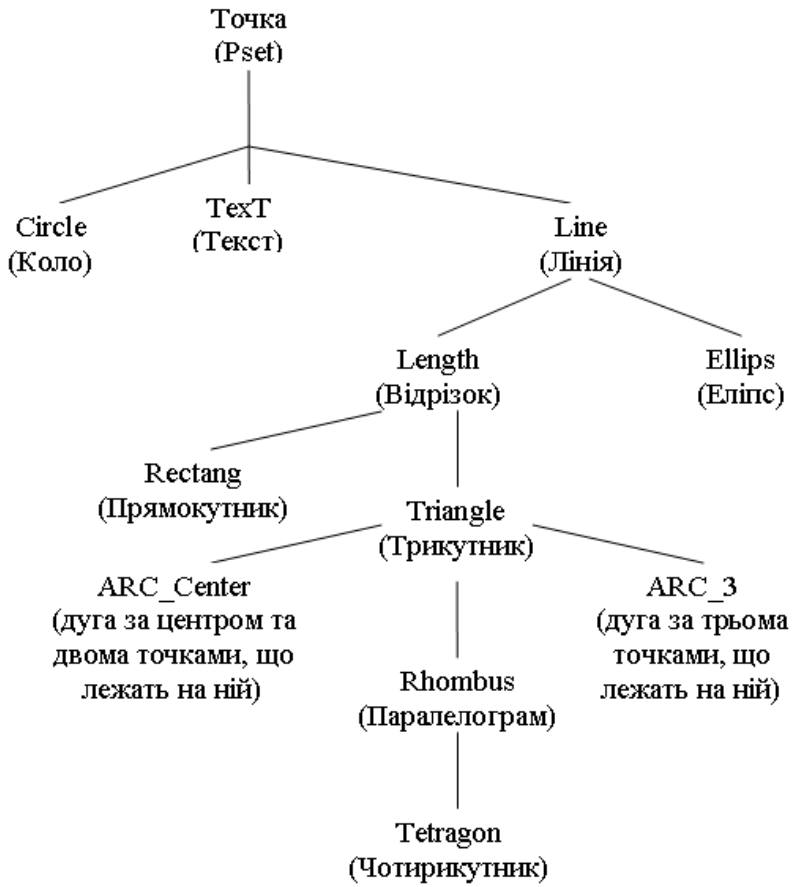
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. А.Я. Архангельський. ПРОГРАММИРОВАНИЕ В C++ Builder 6. Москва «БИНОМ» 2003. - 1150 с.
2. А.Я. Архангельський.C++ Builder 6. СПРАВОЧНОЕ ПОСОБИЕ. Книга 1Язык C++ . Москва ЗАО «БИНОМ» 2002.- 543 с.
3. А.Я. Архангельський. C++ Builder 6. СПРАВОЧНОЕ ПОСОБИЕ. Книга 2Классы и компоненты. ЗАО «БИНОМ» 2002.- 527с.
4. Об'єктно-орієнтоване програмування. – Wikipedia. –[Цит. 2011 2 квітня]. – Доступний з: <http://uk.wikipedia.org/wiki/Об'єктно-орієнтоване_програмування>
5. С.В. Глушков, С.В. Смирнов, А.В. Коваль. Практикум по C++. – Харьков:ФОЛИО, 2006.-525с.
6. . STL. – Wikipedia. –[Цит. 2011 2 квітня]. – Доступний з: <http://uk.wikipedia.org/wiki/STL>
7. Справочник по C++ STL.Итераторы. – DarkRaha. –[Цит. 2011 9 березня]. – Доступний з: < http://darkraha.com/rus/cpp/stl/stl08.php>
8. Указатели на компоненты класса. Доступ по указателю. – CYBERGURU. –[Цит. 2011 1 квітня]. – Доступний з: <http://citforum.ru/programming/cpp_march/cpp_082.shtml>
9. Я.М. Глинський, В.Є. Анохін, В.А. Ряжська. C++ і C++ Builder. Львів 2006.- 190 с.
10. . Класс TFileStream. Понятие потока VCL. –CYBERGURU. –[Цит. 2011 3 квітня]. – Доступний з: <http://www.cyberguru.ru/cbuilder-sources/classes-vcl/klass-tfilestream-ponyatie-potoka-vcl.html>
11. Запись в двоичный файл. – CYBERGURU. –[Цит. 2011 7 квітня]. – Доступний з: <http://www.cyberguru.ru/programming/cpp-builder/borland-cpp-vcl-faq-page29.html>
12. Пересечение окружности и прямой. –MAXimal. –[Цит. 2011 11 березня]. – Доступний з: <http://e-maxx.ru/algo/circle_line_intersection>
13. Підченко Ю.П., Пастушенко С.Н. Вища математика: У 2-х част.: Навч. Посіб. Для студентів вищих навч. закладів. – К: Діал, 2004. – Ч1 – 192с.

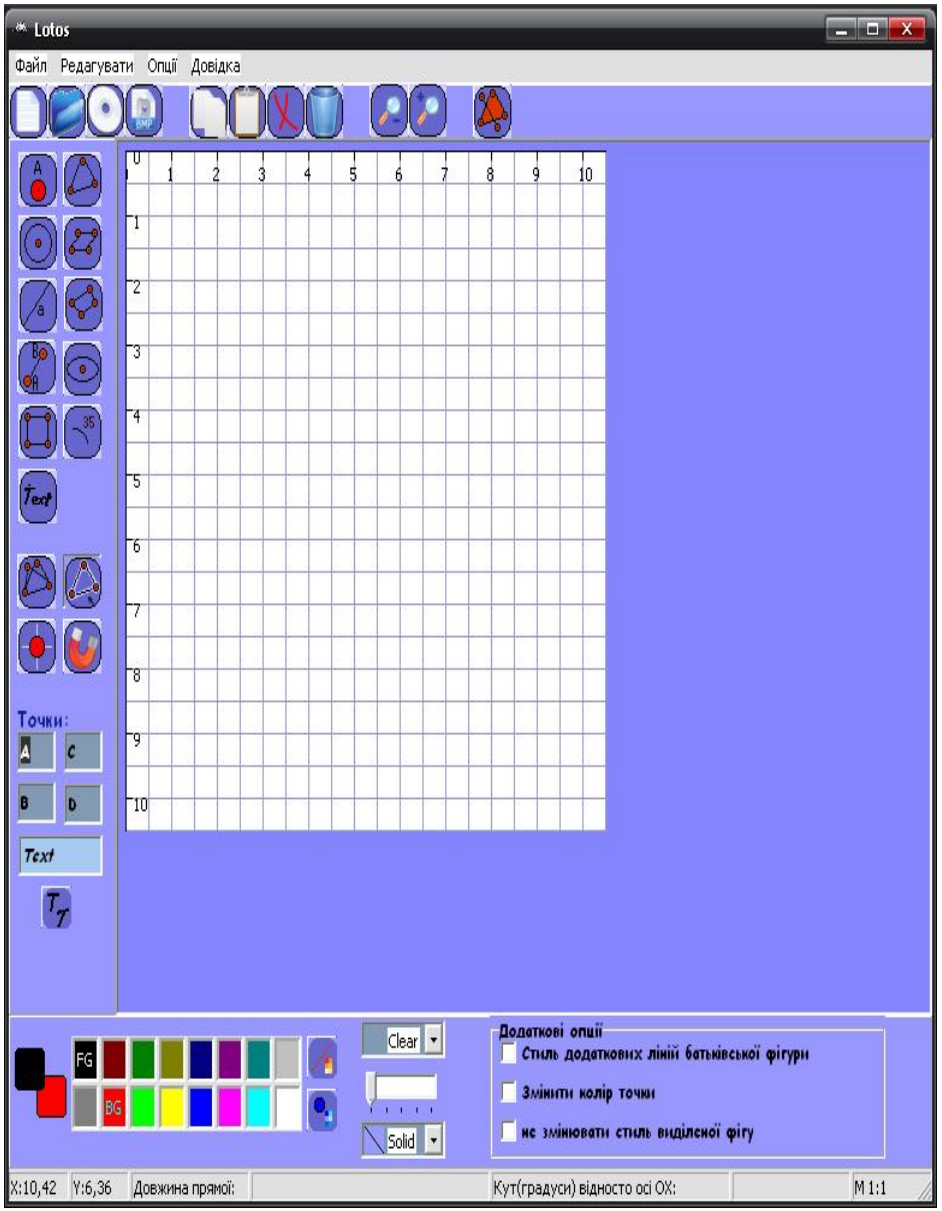
14. Нахождение уравнения прямой для отрезка. –MAXimal. –[Цит. 2011 12 березня]. – Доступний з: <http://e-maxx.ru/algo/segment_to_line>
15. Точка пересечения прямых. –MAXimal. –[Цит. 2011 14 березня]. – Доступний з: <http://e-maxx.ru/algo/lines_intersection>
16. Окружность по трем точкам 2D. –Алгоритмы методы исходники. –[Цит. 2011 13 березня]. – Доступний з: <http://algolist.manual.ru/maths/geom/equation/circle.php>
17. Графічний інтерфейс користувача. – Wikipedia. –[Цит. 2011 15 березня]. – Доступний з: <http://uk.wikipedia.org/wiki/Графічний_інтерфейс_користувача>
18. Win32 API. Справочник по диалоговому окну. – CYBERGURU. –[Цит. 2011 9 квітня]. – Доступний з: <http://www.cyberguru.ru/programming/win32/win32-dialog-window-ref-page22.html>
19. Г.П.Бевз, П.Ф.Фільчаков, К.І.Швецов, Ф.П.Яремчук. Довідник з елементарної математики. Для вступників до вузів. «Наукова думка» Київ, 1973. –735с.
20. С.И.Зетель. Геометрия линейки и геометрия циркуля. Москва, 1950. 160с.
21. Культин Н. Б. Самоучитель C++ Builder. - СПб.: БХВ-Петербург, 2004. -320 с.
22. Культин Н. Б.C/C++ в задачах и примерах. — СПб.: БХВ-Петербург, 2005. — 288 с.
23. Линьков В.М., Яременко Н.Н. Высшая математика в примерах и задачах. Компьютерный практикум: Учеб. Пособие/Под ред. А.А. Емельянова – М.: Финансы и статистика, 2006. – 320с.
24. Жалдак М.І., Комп'ютер на уроках математики: Посібник для вчителів. — К. : Техніка, 1997.
25. Слєпкань З.І. Методика навчання математики. К.: «Вища школа»,2006.

ДОДАТКИ

Додаток А.Ієрархічне дерево класів



Додаток В
Головне вікно програми



НАУКОВО - МЕТОДИЧНЕ ВИДАННЯ

Оксана Мирославівна Черкун

Розробка і дослідження графічного середовища LOTOS для створення динамічних геометричних побудов

МОНОГРАФІЯ

Під редакцією Р.М.Літнаровича, кандидата технічних наук, доцента

Комп'ютерний набір в редакторі Microsoft®Office® Word 2007
О.М. Черкун

Редагування, верстка, макетування та дизайн Р.М.Літнарович

Науковий керівник Ю.Г.Лотюк, кандидат педагогічних наук, доцент

Науковий консультант Р.М.Літнарович, доцент, кандидат технічних наук

Міжнародний Економіко-Гуманітарний Університет імені академіка Степана Дем'янчука

33027,м.Рівне,Україна
Вул.акад. С.Дем'янчука,4, корпус 1
Телефон:(+00380) 362 23-73-09
Факс:(+00380) 362 23-01-86
E-mail:mail@regi.rovno.ua
E-mail: oksana077@list.ru