МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

**Український державний університет
науки і технологій**

Кафедра «Технологія машинобудування»

В авторській редакції

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Навчально-методичні рекомендації
до виконання лабораторного практикуму

Частина ІІІ

Електронне видання

ДНІПРО
2025

УДК 004.652.5(076.5)
О 29

Упорядники:
В. С. Гришин, В. М. Карабут

Електронне видання

Схвалено Групою забезпечення якості освітньої програми
«Технологія машинобудування»
131 «Прикладна механіка» (бакалаврський рівень)
Протокол №5 від 20.03.2025 р.

О 29 Об'єктно-орієнтоване програмування : навчально-методичні рекомендації до виконання лабораторного практикуму / упоряд.: В. С. Гришин, В. М. Карабут ; Укр. держ. ун-т науки і технологій. – Електрон. вид. – Дніпро : УДУНТ, 2025. – Ч. III – 114 с.

Наведені методика і послідовність проведення лабораторних робіт, вимоги до оформлення звіту з лабораторних робіт, запитання з самоконтролю до розділів, рекомендована література.

Призначені для студентів спеціальності 131 «Прикладна механіка» (бакалаврський рівень).

ЗМІСТ

РОЗДІЛ 6. МЕТОДИ ВИВОДУ ДАНИХ

Лабораторна робота №17

Тема: Структурне програмування. Підпрограма – процедура.

Мета роботи: Засвоєння початкових знань та придбання навичок, необхідних для розробки програмних продуктів комп`ютерного забезпечення в технологічних процесах обробки металів різанням.

Порядок виконання роботи

1. Вивчити теоретичні відомості.
2. Виконати алгоритм виконання лабораторної роботи №17.
3. Скласти звіт з лабораторної роботи №17 за наведеною формою (див. Додаток С).

Хід виконання роботи

1. Теоретичні відомості

Вивчити теоретичні відомості зі списку наведеної літератури.

2. Алгоритм виконання лабораторної роботи №17

2.1. Запуск програми Delphi

У системному меню «***Пуск***», операційної системи Windows, вибрати: **Пуск > Програми > Borland Delphi 7 > Delphi 7**.

2.2. Створення проєкту

Після запуску програми Delphi створюється за замовчуванням проєкт типу – **Application** (Програма).

Delphi дозволяє розробляти різні програми, які в Delphi називаються проєктами. У лабораторних роботах створюються проєкти – **Application** (Програма), під операційну систему Windows з графічним інтерфейсом.

Для створення нового проєкту виберіть в «***Головному***» меню **File** (Файл) **New** (Нова) > **Application** (Програма) (рис. 2.1).

2.3. Збереження проєкту

Delphi для кожного проєкту створює декілька файлів. Щоб файли різних проєктів не переплуталися між собою, слід для кожного проєкту створювати свою **папку** з іменем, яке містить назву та номер лабораторної роботи (наприклад, лабораторна робота 17 – Lab_17).

Рекомендується імені проєкту надавати ім'я, яке містить номер лабораторної роботи (наприклад, проєкт для лабораторної роботи 17 – Project_17). **Файл проєкту** Project _17 потрібно зберегти в папці Lab_17.

Необхідно також зберегти **файл модуля вихідного коду**.

Рекомендується імені файлу модуля вихідного коду надавати ім'я, як у проєкті (наприклад, Unit _17). Його також потрібно зберегти у папці Lab_17.

На панелі інструментів «***Standard***» (Стандартна) натиснути кнопку **Save all** (Зберегти все) та зберегти **файли проєкту** та **файл модуля вихідного коду** згідно з вище сказаними рекомендаціями (рис. 2.2).

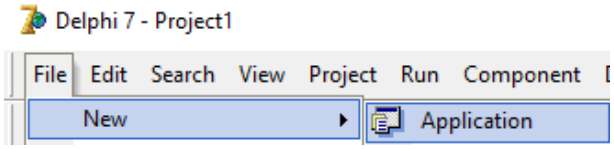


Рис. 2.1. Виберіть **New** > **Application** (Нова > Програма) в «***Головному***» меню **File** (Файл)

Рис. 2.2. Натиснути кнопку **Save all** (Зберегти все) на панелі інструментів «***Standard***» (Стандартна)

2.4. Розробка програми

Розробити програму обчислення значень змінних a та b, якщо: x = 2,1; y = 0,59 та z = - 4,8, використовуючи формули:

$$a = y \cdot tg^3(x^2) + \sqrt{\frac{z^2}{y^2 + x^2}} \qquad b = \ln(y + x^2) + sin^2\left(\frac{z}{x}\right)$$

2.4.1. Створення інтерфейсу програми

Інтерфейс програми представлений на рис. 2.3. Вивід повідомлень програми представлено на рис. 2.4 (а, б).

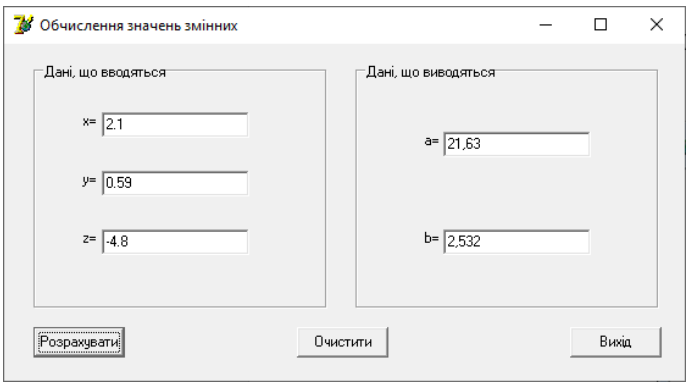
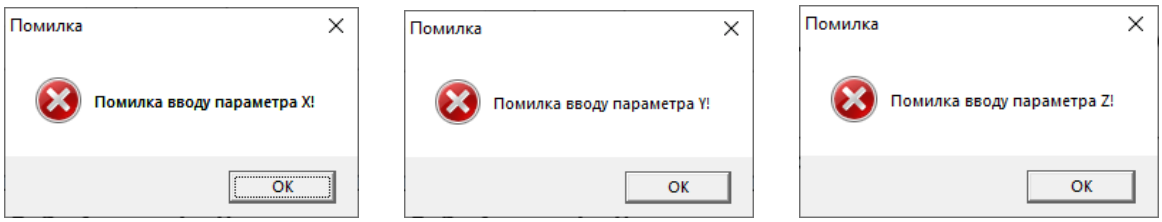


Рис. 2.3. Інтерфейс програми



а б в

Рис. 2.4. Вивід повідомлень програми: а, б, в

2.4.1.1. Вікно форми Form1 (Форма1)

При створенні проєкту типу – **Application** (Програма), Delphi автоматично створює вікно форми «***Form1***» (Форма1) (рис. 2.5). Вікно форми «***Form1***» (Форма1) являє собою шаблон вікна розробляємої програми.

2.4.1.2. Розміщення компонентів на вікні форми Form1 (Форма1)

На вікні форми «***Form1***» (Форма1) розмістити компоненти (рис. 2.6) вказані в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) на «***Панелі компонентів***» на вкладці «***Standard***» (Стандартна) (рис. 2.7) вказати курсором миші на піктограму потрібного компонента (табл. 2.1);

б) потім вказати курсором миші у довільному місці на формі «***Form1***» (Форма1) (рис. 2.6);

в) якщо необхідно, перетягніть компонент у потрібне місце на формі «***Form1***» (Форма1) (рис. 2.6).

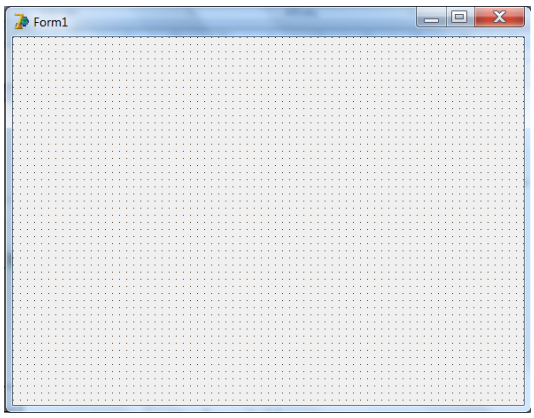


Рис. 2.5. Вікно форми «***Form1***» (Форма1)

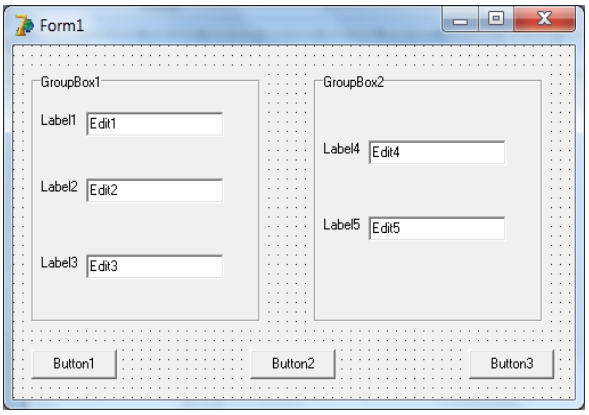


Рис. 2.6. Компоненти розміщені на вікні форми «***Form1***» (Форма1)

Рис. 2.7. «***Панель компонентів***» на вкладці «***Standard***» (Стандартна)

Таблиця 2.1 – Компоненти та їх властивості для інтерфейсу програми

Компонент	**Піктограма**	**Властивість**	**Значення**
Form1	-	Caption	Обчислення значень змінних
		BorderStyle	bsSingle
Groupbox1		Caption	Дані, що вводяться
Groupbox2		Caption	Дані, що виводяться
Edit1		Name	edtInputx
		Text	значення не вводимо, а тільки очистити поле
Edit2		Name	edtInputy
		Text	значення не вводимо, а тільки очистити поле
Edit3		Name	edtInputz
		Text	значення не вводимо, а тільки очистити поле
Edit4		Name	edtOutputa
		ReadOnly	True
		Text	значення не вводимо, а тільки очистити поле
Edit5		Name	edtOutputb
		ReadOnly	True
		Text	значення не вводимо, а тільки очистити поле
Label1		Caption	x=

Продовження таблиці 2.1

Компонент	Піктограма	Властивість	Значення
Label2	A	Caption	y=
Label3	A	Caption	z=
Label4	A	Caption	a=
Label5	A	Caption	b=
Button1	OK	Name	btnCalculate
		Caption	Розрахувати
Button2	OK	Name	btnClear
		Caption	Очистити
Button3	OK	Name	btnExit
		Caption	Вихід

2.4.1.3. Зміна властивостей вікна форми Form1 (Форма1) та компонентів

Змінимо властивості вікна форми «***Form1***» (Форма1) та компонентів (рис. 2.8) згідно з вказаними властивостями в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.9) вибрати ім'я об'єкта (форми або компонентів) (табл. 2.1);

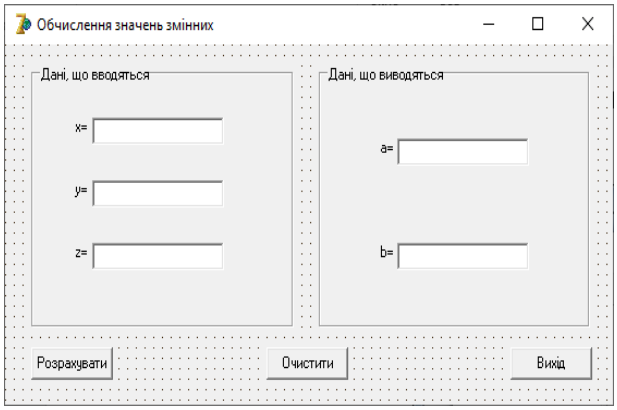


Рис. 2.8. Компоненти та вікно форми «***Form1***» (Форма1) із зміненими властивостями

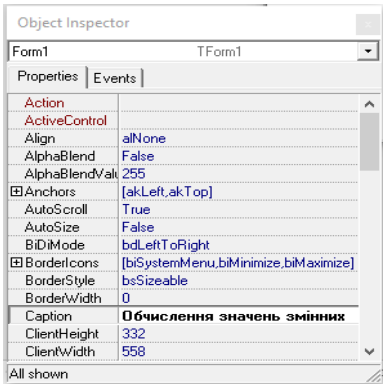


Рис. 2.9. Зміна значення властивості **Caption** (Заголовок) вікна форми «***Form1***» (Форма1) у вікні «***Object Inspector***» (Інспектор об'єктів) на **Обчислення значень змінних**

б) у лівій колонці вкладки «***Properties***» (Властивості) (рис. 2.9), де перелічені властивості об'єкта (форми чи компонентів) виділити властивість (табл. 2.1);

в) у правій колонці вкладки «***Properties***» (Властивості) (рис. 2.9), де вказано значення об'єкта (форми або компонентів) вибрати або ввести значення властивості (табл. 2.1).

2.4.2. Створення обробників подій

Створимо обробники подій для компонентів, вказаних у таблиці 2.2.

Таблиця 2.2 – Обробники подій компонентів для програми

Компонент	Подія	Ім'я обробника
btnCalculate	OnClick	btnCalculateClick
btnClear	OnClick	btnClearClick
btnExit	OnClick	btnExitClick

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.10) вибрати ім'я об'єкта (форми або компонента) (табл. 2.2);

б) у цьому ж вікні потрібно вибрати вкладку «***Events***» (Події) (рис. 2.10);

в) у лівій колонці вкладки «***Events***» (Події) (рис. 2.10), де перелічені імена подій, які може використовувати вибраний об’єкт (компонент), вибрати ім'я події (табл. 2.2);

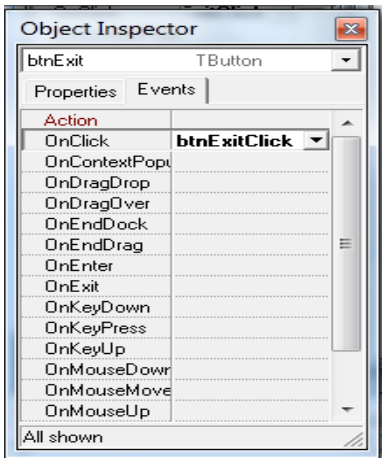


Рис. 2.10. Створення обробника **btnExitClick** події **OnClick** для компонента btnExit, на вкладці «***Events***» (Події) у вікні «***Object Inspector***» (Інспектор об'єктів)

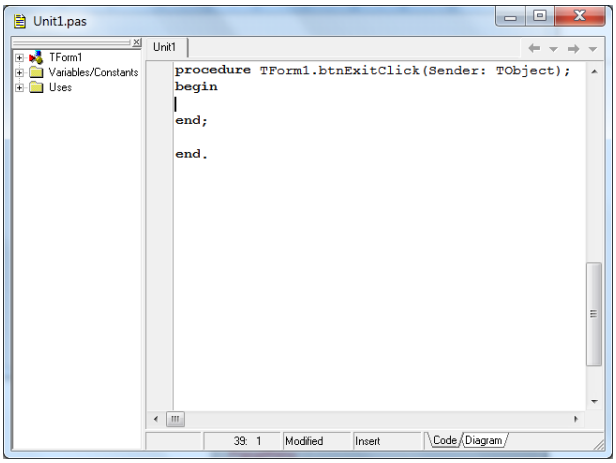


Рис. 2.11. Шаблон процедури обробника **btnExitClick** події **OnClick**, згенерований програмою Delphi

г) для створення обробника події потрібно у правій колонці вкладки «***Events***» (Події) навпроти обраної раніше події зробити подвійне клацання мишею у полі імені обробника (рис. 2.10);

д) відкриється вікно «***Unit_17.pas***» (Редактор коду), до якого буде додано шаблон процедури обробника **btnExitClick** події **OnClick** (рис. 2.11), а у вікні «***Object Inspector***» (Інспектор об’єктів) поряд з ім'ям події з'явиться ім'я обробника (рис. 2.10);

е) у вікні «***Unit_17.pas***» (Редактор коду) між словами **begin** і **end** необхідно ввести інструкції, що реалізують обробку події (рис. 2.11). У лістингу програми, див. п. п. 2.4.3, наведені тексти інструкцій обробки події **OnClick** для кнопок: **Розрахувати** (btnCalculate), **Очистити** (btnClear) та **Вихід** (btnExit).

2.4.3. Створення вихідного коду

Вихідний код програми представлений у вигляді лістингу програми.

Так як у вихідному коді використовуються математичні функції, то необхідно в розділі **uses** додати модуль **Math**.

Лістинг програми

```
unit Unit_17;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Math;
type
  TForm1 = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    edtInputx: TEdit;
    edtInputy: TEdit;
    edtInputz: TEdit;
    edtOutputa: TEdit;
    edtOutputb: TEdit;
    Label1: TLabel;
```

```
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    btnCalculate: TButton;
    btnClear: TButton;
    btnExit: TButton;
    procedure btnCalculateClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
    procedure btnExitClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
// Оголошення процедури
procedure calc(fx,fy,fz:real; var fa,fb:real);
begin
  // Обробка даних
  fa := (fy*power(tan(sqr(fx)),3))+(sqrt(sqr(fz)/(sqr(fy)+sqr(fx))));
  fb := ln(fy+sqr(fx))+sqr(sin(fz/fy));
end;
procedure TForm1. btnCalculateClick(Sender: TObject);
var
  // Оголошення змінних
  x,y,z,a,b : Real;
  code : Integer;
begin
  // Ввід даних із захистом поля вводу
  // X
  val(edtInputx.Text,x,code);
```

```
  if (code<>0) or (x=0) then
  begin
    Application.MessageBox('Помилка вводу параметра X!',
    'Помилка', mb_IconError + mb_OK);
    edtInputx.SetFocus;
    exit;
  end;
  // Y
  val(edtInputy.Text,y,code);
  if (code<>0) or (y=0) then
  begin
    Application.MessageBox('Помилка вводу параметра Y!',
    'Помилка', mb_IconError + mb_OK);
    edtInputy.SetFocus;
    exit;
  end;
  // Z
  val(edtInputz.Text,z,code);
  if (code<>0) or (z=0) then
  begin
    Application.MessageBox('Помилка вводу параметра Z!',
    'Помилка', mb_IconError + mb_OK);
    edtInputz.SetFocus;
    exit;
  end;
   // Звернення до процедури
  calc(x,y,z,a,b);
   // Вивід даних із форматуванням
  edtOutputa.Text := FloatToStrF(a,ffGeneral,4,2);
  edtOutputb.Text := FloatToStrF(b,ffGeneral,4,2);
end;
procedure TForm1.btnClearClick(Sender: TObject);
begin
  // Очищення полів вводу
  edtInputx.Text:= ' ';
```

```
  edtInputy.Text:= ' ';
  edtInputz.Text:= ' ';
  // Очищення полів виводу
  edtOutputa.Clear;
  edtOutputb.Clear;
end;
procedure TForm1.btnExitClick(Sender: TObject);
begin
   Close;    // Закриття програми
end;
end.
```

2.4.4. Компіляція програми

Процес компіляції складається із двох етапів. На першому етапі виконується перевірка тексту програми на відсутність синтаксичних помилок, а на другому – генерується виконувана програма (exe-файл).

Компіляція програми виконується з «***Головного***» меню **Project** (Проєкт) обравши команду **Compile Project** (Компілювати проєкт) (рис. 2.12) або натиснути клавіші **Ctrl+F9**. Повідомлення про помилки, попередження та підказки відображаються в нижній частині вікна редактора коду (рис. 2.13).

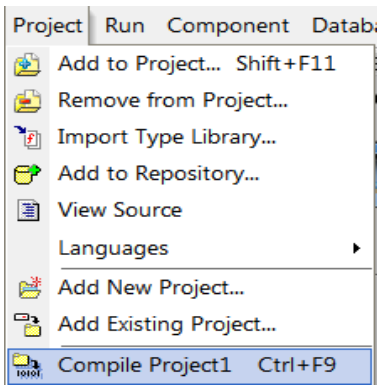


Рис. 2.12. Вибір команди **Compile Project** (Компілювати проєкт) з «***Головного***» меню **Project** (Проєкт)

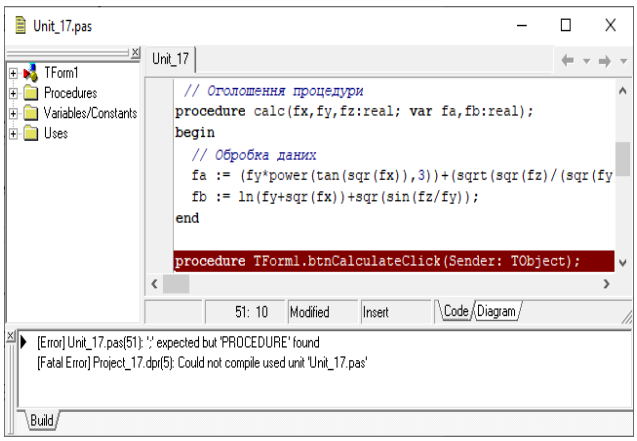


Рис. 2.13. Повідомлення компілятора про виявлені помилки

Процес усунення помилок має ітераційний характер. Зазвичай, спочатку усуваються найбільш очевидні помилки, наприклад, декларуються неоголошені змінні. Після чергового внесення змін до тексту програми виконується

повторна компіляція. Слід зважати на те, що компілятор не завжди може точно локалізувати помилку. Тому аналізуючи фрагмент програми який на думку компілятора містить помилку, потрібно звертати увагу не тільки на той фрагмент коду, на який компілятор встановив курсор, але й на той, який знаходиться у попередньому рядку.

Якщо у програмі немає синтаксичних помилок, компілятор створює виконуваний файл програми. Ім'я виконуваного файлу таке ж, як і у файлу проєкту, а розширення – exe. Delphi поміщає виконуваний файл у той самий каталог, де знаходиться файл проєкту.

2.4.5. Запуск програми

Запуск програми може виконуватися з «***Головного***» меню **Run** (Запуск) обравши команду **Run** (Запуск) або натиснути на кнопці  **Run** (Запуск) на панелі інструментів «***Debug***» (Налагодження) (рис. 2.14) або натиснути клавішу **F9.**

Рис. 2.14. Панель інструментів «***Debug***» (Налагодження)

2.4.6. Помилки часу виконання

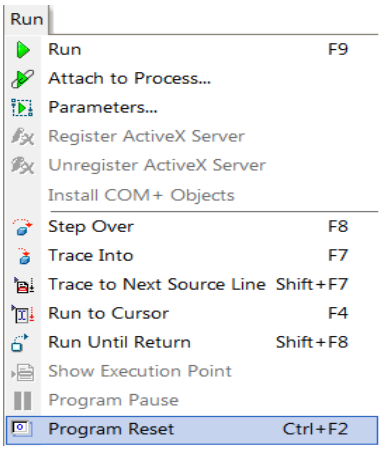


Рис. 2.15. Вибір команди **Program Reset** (Скидання програми) із «***Головного***» меню **Run** (Запуск)

Під час роботи програми можуть виникати помилки, які називаються **помилками часу виконання** (run-time errors) або **винятками** (exceptions).

У більшості випадків причинами винятків є невірні вихідні дані. Після виникнення таких помилок програма перестане робити. Необхідно перервати неробочу програму, вибравши із «***Головного***» меню **Run** (Запуск) команду **Program Reset** (Скидання програми) (рис. 2.15).

2.4.7. Захист полів вводу

Для виключення помилок, при вводі даних у поля вводу необхідно передбачити у вихідному коді програми захист полів вводу.

У лабораторній роботі №17 використовуємо рядкову функцію Val, яка дозволяє використовувати такі види захисту поля вводу (див. п. п. 2.4.3):

- захист поля вводу від вводу порожнього рядка;
- захист поля вводу від вводу цифри «0» нуль;
- захист поля вводу від вводу літер та деяких символів.

3. Зміст звіту з лабораторної роботи №17

Звіт з лабораторної роботи №17 складається з:

- титульного аркуша (див. Додаток С);
- аркуша (див. Додаток С) на якому розташовано: **тема** лабораторної роботи, **мета** лабораторної роботи, **розробка програми** (наведіть **умову** та **формули**, які розглядаються в лабораторній роботі №17, див. п. п. 2.4);
- аркуша (див. Додаток С) на якому розташовано: **створення інтерфейсу програми** (навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №17, див. п. п. 2.4.1);
- аркушів (див. Додаток С) на яких розташовані: **лістинг програми** (навести **лістинг** програми, який розглядається в лабораторній роботі №17, див. п. п. 2.4.3) та **висновки**.

Лабораторна робота №18

Тема: Структурне програмування. Підпрограма - функція

Мета роботи: Засвоєння початкових знань та придбання навичок, необхідних для розробки програмних продуктів комп`ютерного забезпечення в технологічних процесах обробки металів різанням.

Порядок виконання роботи

1. Вивчити теоретичні відомості.
2. Виконати алгоритм виконання лабораторної роботи №18.

3. Скласти звіт з лабораторної роботи №18 за наведеною формою (див. Додаток Т).

Хід виконання роботи

1. Теоретичні відомості

Вивчити теоретичні відомості зі списку наведеної літератури.

2. Алгоритм виконання лабораторної роботи №18

2.1. Запуск програми Delphi

У системному меню «***Пуск***», операційної системи Windows, вибрати: **Пуск > Програми > Borland Delphi 7 > Delphi 7**.

2.2. Створення проєкту

Після запуску програми Delphi створюється за замовчуванням проєкт типу – **Application** (Програма).

Delphi дозволяє розробляти різні програми, які в Delphi називаються проєктами. У лабораторних роботах створюються проєкти – **Application** (Програма), під операційну систему Windows з графічним інтерфейсом.

Для створення нового проєкту в «***Головному***» меню **File** (Файл) вибрати: **New** (Нова) > **Application** (Програма) (рис. 2.1).

2.3. Збереження проєкту

Delphi для кожного проєкту створює декілька файлів. Щоб файли різних проєктів не переплуталися між собою, слід для кожного проєкту створювати свою **папку** з іменем, яке містить назву та номер лабораторної роботи (наприклад, лабораторна робота 18 – Lab_18).

Рекомендується імені проєкту надавати ім’я, яке містить номер лабораторної роботи (наприклад, проєкт для лабораторної роботи 18 – Project_18). **Файл проєкту** Project _18 потрібно зберегти в папці Lab_18.

Необхідно також зберегти **файл модуля вихідного коду**.

Рекомендується імені файлу модуля вихідного коду надавати ім’я, як у проєкті (наприклад, Unit _18). Його також потрібно зберегти у папці Lab_18.

На панелі інструментів «***Standard***» (Стандартна) натиснути кнопку **Save all** (Зберегти все) та зберегти **файли проєкту** та **файл модуля вихідного коду** згідно з вище сказаними рекомендаціями (рис. 2.2).

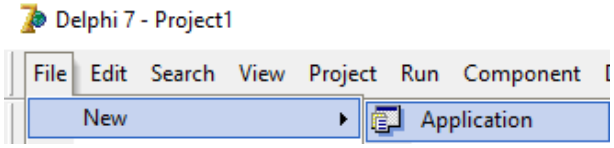


Рис. 2.1. Виберіть **New** > **Application** (Нова > Програма) в «***Головному***» меню **File** (Файл)

Рис. 2.2. Натиснути кнопку **Save all** (Зберегти все) на панелі інструментів «***Standard***» (Стандартна)

2.4. Розробка програми

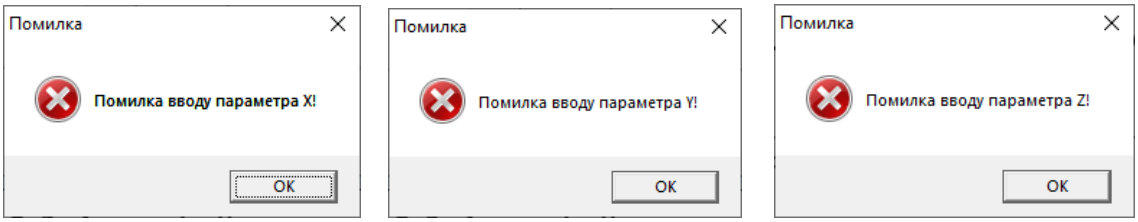
Розробити програму обчислення значень змінних a та b, якщо: x = 2,1; y = 0,59 та z = - 4,8, використовуючи формули:

$$a = y \cdot tg^3(x^2) + \sqrt{\frac{z^2}{y^2 + x^2}} \qquad b = \ln(y + x^2) + sin^2\left(\frac{z}{x}\right)$$

2.4.1. Створення інтерфейсу програми

Інтерфейс програми представлений на рис. 2.3. Вивід повідомлень програми представлено на рис. 2.4 (а, б).

Рис. 2.3. Інтерфейс програми



а б в

Рис. 2.4. Вивід повідомлень програми: а, б, в

2.4.1.1. Вікно форми Form1 (Форма1)

При створенні проєкту типу – **Application** (Програма), Delphi автоматично створює вікно форми «***Form1***» (Форма1) (рис. 2.5). Вікно форми «***Form1***» (Форма1) являє собою шаблон вікна розробляємої програми.

2.4.1.2. Розміщення компонентів на вікні форми Form1 (Форма1)

На вікні форми «***Form1***» (Форма1) розмістити компоненти (рис. 2.6) вказані в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) на «***Панелі компонентів***» на вкладці «***Standard***» (Стандартна) (рис. 2.7) вказати курсором миші на піктограму потрібного компонента (табл. 2.1);

б) потім вказати курсором миші у довільному місці на формі «***Form1***» (Форма1) (рис. 2.6);

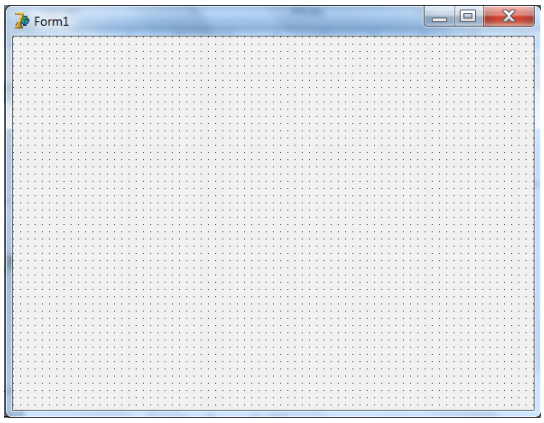


Рис. 2.5. Вікно форми «***Form1***» (Форма1)

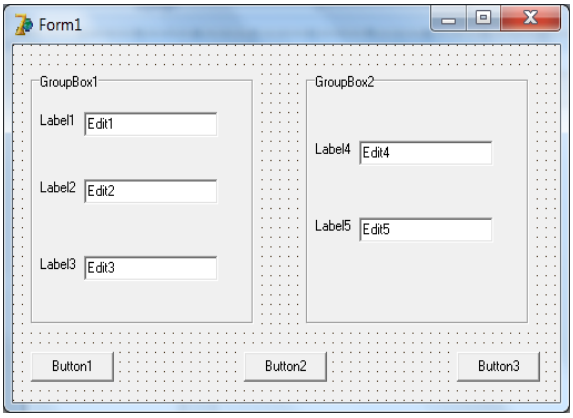


Рис. 2.6. Компоненти розміщені на вікні форми «***Form1***» (Форма1)

в) якщо необхідно, перетягніть компонент у потрібне місце на формі «***Form1***» (Форма1) (рис. 2.6).

Рис. 2.7. «***Панель компонентів***» на вкладці «***Standard***» (Стандартна)

Таблиця 2.1 – Компоненти та їх властивості для інтерфейсу програми

Компонент	Піктограма	Властивість	Значення
Form1	-	Caption	Обчислення значень змінних
		BorderStyle	bsSingle
Groupbox1		Caption	Дані, що вводяться
Groupbox2		Caption	Дані, що виводяться
Edit1		Name	edtInputx
		Text	значення не вводимо, а тільки очистити поле
Edit2		Name	edtInputy
		Text	значення не вводимо, а тільки очистити поле
Edit3		Name	edtInputz
		Text	значення не вводимо, а тільки очистити поле
Edit4		Name	edtOutputa
		Text	значення не вводимо, а тільки очистити поле
		ReadOnly	True
Edit5		Name	edtOutputb
		Text	значення не вводимо, а тільки очистити поле
		ReadOnly	True
Label1	A	Caption	x=
Label2	A	Caption	y=
Label3	A	Caption	z=
Label4	A	Caption	a=
Label5	A	Caption	b=
Button1		Name	btnCalculate
		Caption	Розрахувати
Button2		Name	btnClear
		Caption	Очистити
Button3		Name	btnExit
		Caption	Вихід

2.4.1.3. Зміна властивостей вікна форми Form1 (Форма1) та компонентів

Змінимо властивості вікна форми «***Form1***» (Форма1) та компонентів (рис. 2.8) згідно з вказаними властивостями в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.9)

вибрати ім'я об'єкта (форми або компонентів) (табл. 2.1);

б) у лівій колонці вкладки «***Properties***» (Властивості) (рис. 2.9), де перелічені властивості об'єкта (форми чи компонентів) виділити властивість (табл. 2.1);

в) у правій колонці вкладки «***Properties***» (Властивості) (рис. 2.9), де вказано значення об'єкта (форми або компонентів) вибрати або ввести значення властивості (табл. 2.1).

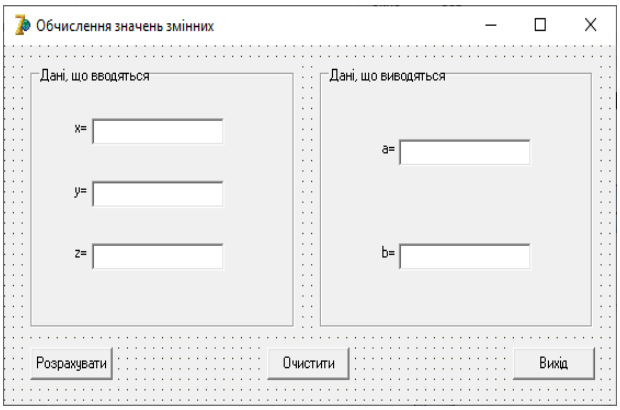


Рис. 2.8. Компоненти та вікно форми «***Form1***» (Форма1) із зміненими властивостями

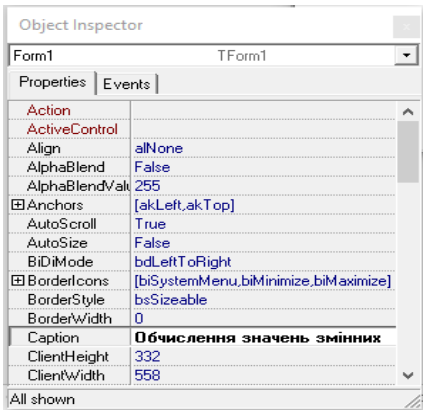


Рис. 2.9. Зміна значення властивості **Caption** (Заголовок) вікна форми «***Form1***» (Форма1) у вікні «***Object Inspector***» (Інспектор об'єктів) на **Обчислення значень змінних**

2.4.2. Створення обробників подій

Створимо обробники подій для компонентів, вказаних у таблиці 2.2.

Таблиця 2.2 – Обробники подій компонентів для програми

Компонент	Подія	Ім'я обробника
btnCalculate	OnClick	btnCalculateClick
btnClear	OnClick	btnClearClick
btnExit	OnClick	btnExitClick

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.10) вибрати ім'я об'єкта (форми або компонента) (табл. 2.2);

б) у цьому ж вікні потрібно вибрати вкладку «***Events***» (Події) (рис. 2.10);

в) у лівій колонці вкладки «***Events***» (Події) (рис. 2.10), де перелічені імена подій, які може використовувати вибраний об’єкт (компонент), вибрати ім'я події (табл. 2.2);

г) для створення обробника події потрібно у правій колонці вкладки «***Events***» (Події) навпроти обраної раніше події зробити подвійне клацання мишею у полі імені обробника (рис. 2.10);

д) відкриється вікно «***Unit_18.pas***» (Редактор коду), до якого буде додано шаблон процедури обробника **btnExitClick** події **OnClick** (рис. 2.11), а у вікні «***Object Inspector***» (Інспектор об’єктів) поряд з ім'ям події з'явиться ім'я обробника (рис. 2.10);

е) у вікні «***Unit_18.pas***» (Редактор коду) між словами **begin** і **end** необхідно ввести інструкції, що реалізують обробку події (рис. 2.11). У лістингу програми, див. п. п. 2.4.3, наведені тексти інструкцій обробки події **OnClick** для кнопок: **Розрахувати** (btnCalculate), **Очистити** (btnClear) та **Вихід** (btnExit).

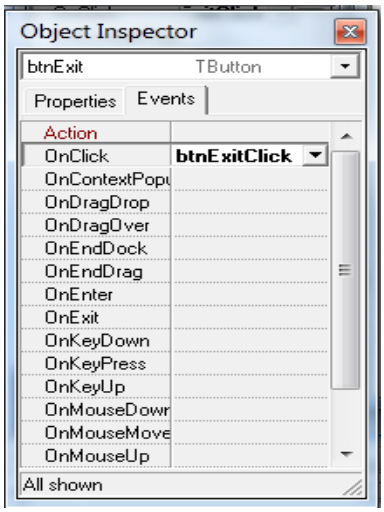


Рис. 2.10. Створення обробника **btnExitClick** події **OnClick** для компонента btnExit, на вкладці «***Events***» (Події) у вікні «***Object Inspector***» (Інспектор об'єктів)

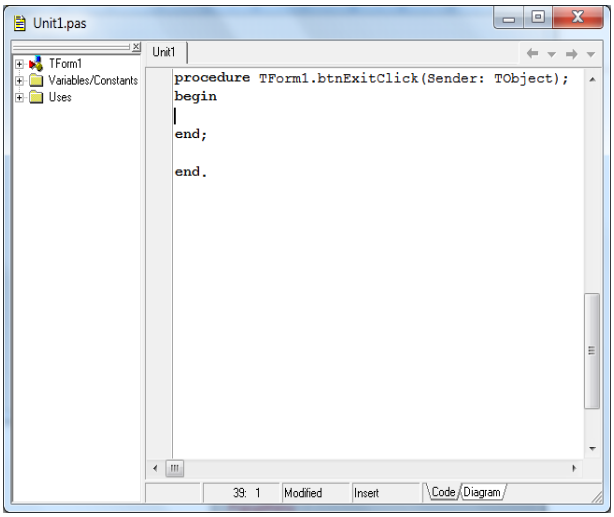


Рис. 2.11. Шаблон процедури обробника **btnExitClick** події **OnClick**, згенерований програмою Delphi

2.4.3. Створення вихідного коду

Вихідний код програми представлений у вигляді лістингу програми. Так

як у вихідному коді використовуються математичні функції, то необхідно в розділі **uses** додати модуль **Math**.

Лістинг програми

```
unit Unit_18;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Math;
type
  TForm1 = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    edtInputx: TEdit;
    edtInputy: TEdit;
    edtInputz: TEdit;
    edtOutputa: TEdit;
    edtOutputb: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    btnCalculate: TButton;
    btnClear: TButton;
    btnExit: TButton;
    procedure btnCalculateClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
    procedure btnExitClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
end;
```

```
var
  Form1: TForm1;
implementation
{$R *.dfm}
// Оголошення функції calca
function calca(fx,fy,fz:real):real;
begin
  // Обробка даних
  calca:= (fy*power(tan(sqr(fx)),3))+(sqrt(sqr(fz)/(sqr(fy)+sqr(fx))))    ;
end;
// Оголошення функції calcb
function calcb(fx,fy,fz:real):real;
begin
  // Обробка даних
  calcb := ln(fy+sqr(fx))+sqr(sin(fz/fy));
end;
procedure TForm1.btnCalculateClick(Sender: TObject);
var
  // Оголошення змінних
  x, y, z, resulta, resultb : Real;
  code : Integer;
begin
  // Ввід даних із захистом поля вводу
  // X
  val(edtInputx.Text,x,code);
  if (code<>0) or (x=0) then
  begin
    Application.MessageBox('Помилка вводу параметра X!',
    'Помилка', mb_IconError + mb_OK);
    edtInputx.SetFocus;
    exit;
  end;
  // Y
  val(edtInputy.Text,y,code);
  if (code<>0) or (y=0) then
```

```
  begin
    Application.MessageBox('Помилка вводу параметра Y!',
    'Помилка', mb_IconError + mb_OK);
    edtInputy.SetFocus;
    exit;
  end;
  // Z
  val(edtInputz.Text,z,code);
  if (code<>0) or (z=0) then
  begin
    Application.MessageBox('Помилка вводу параметра Z!',
    'Помилка', mb_IconError + mb_OK);
    edtInputz.SetFocus;
    exit;
  end;
  // Звернення до функцій
  resulta := calca(x,y,z);
  resultb := calcb(x,y,z);
   // Виведення даних із форматуванням
  edtOutputa.Text := FloatToStrF(resulta,ffGeneral,4,2);
  edtOutputb.Text := FloatToStrF(resultb,ffGeneral,4,2);
end;
procedure TForm1.btnClearClick(Sender: TObject);
begin
  // Очистка полів вводу
  edtInputx.Text:= ' ';
  edtInputy.Text:= ' ';
  edtInputz.Text:= ' ';
  // Очистка полів виводу
  edtOutputa.Clear;
  edtOutputb.Clear;
end;
procedure TForm1.btnExitClick(Sender: TObject);
begin
   Close; // Закриття програми
```

end;

end.

2.4.4. Компіляція програми

Процес компіляції складається із двох етапів. На першому етапі виконується перевірка тексту програми на відсутність синтаксичних помилок, а на другому – генерується виконувана програма (exe-файл).

Компіляція програми виконується з «***Головного***» меню **Project** (Проєкт) обравши команду **Compile Project** (Компілювати проєкт) (рис. 2.12) або натиснути клавіші **Ctrl+F9**. Повідомлення про помилки, попередження та підказки відображаються в нижній частині вікна редактора коду (рис. 2.13).

Процес усунення помилок має ітераційний характер. Зазвичай, спочатку усуваються найбільш очевидні помилки, наприклад, декларуються неоголошені змінні. Після чергового внесення змін до тексту програми виконується повторна компіляція.

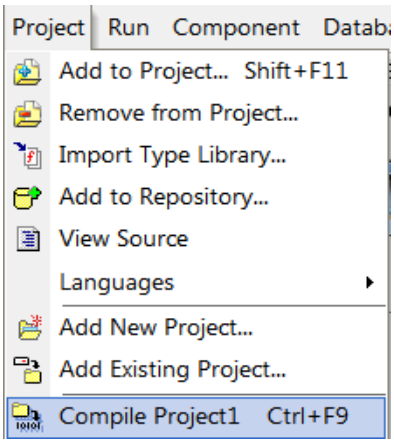


Рис. 2.12. Вибір команди **Compile Project** (Компілювати проєкт) з «***Головного***» меню **Project** (Проєкт)

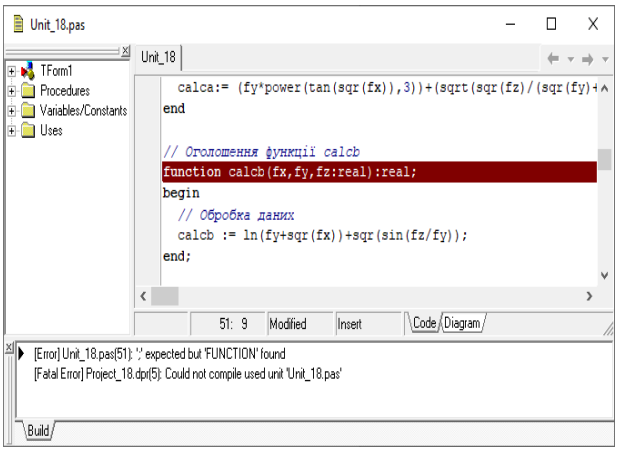


Рис. 2.13. Повідомлення компілятора про виявлені помилки

Слід зважати на те, що компілятор не завжди може точно локалізувати помилку. Тому аналізуючи фрагмент програми який на думку компілятора містить помилку, потрібно звертати увагу не тільки на той фрагмент коду, на який компілятор встановив курсор, але й на той, який знаходиться у попередньому рядку.

Якщо у програмі немає синтаксичних помилок, компілятор створює виконуваний файл програми. Ім'я виконуваного файлу таке ж, як і у файлу

проєкту, а розширення – exe. Delphi поміщає виконуваний файл у той самий каталог, де знаходиться файл проєкту.

2.4.5. Запуск програми

Запуск програми може виконуватися з «***Головного***» меню **Run** (Запуск) обравши команду **Run** (Запуск) або натиснути на кнопці  **Run** (Запуск) панелі інструментів «***Debug***» (Налагодження) (рис. 2.14) або натиснути клавішу **F9.**

Рис. 2.14. Панель інструментів «***Debug***» (Налагодження)

2.4.6. Помилки часу виконання

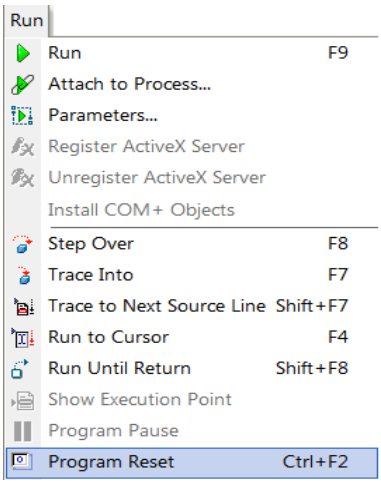


Рис. 2.15. Вибір команди **Program Reset** (Скидання програми) із «***Головного***» меню **Run** (Запуск)

Під час роботи програми можуть виникати помилки, які називаються **помилками часу виконання** (run-time errors) або **винятками** (exceptions).

У більшості випадків причинами винятків є невірні вихідні дані. Після виникнення таких помилок програма перестане робити. Необхідно перервати неробочу програму, вибравши із «***Головного***» меню **Run** (Запуск) команду **Program Reset** (Скидання програми) (рис. 2.15).

2.4.7. Захист полів вводу

Для виключення помилок, при вводі даних у поля вводу необхідно передбачити у вихідному коді програми захист полів вводу.

У лабораторній роботі №18 використовуємо рядкову функцію Val, яка дозволяє використовувати такі види захисту поля вводу (див. п. п. 2.4.3):

- захист поля вводу від вводу порожнього рядка;

- захист поля вводу від вводу цифри «0» нуль;

- захист поля вводу від вводу літер та деяких символів.

3. Зміст звіту з лабораторної роботи №18

Звіт з лабораторної роботи №18 складається з:

- титульного аркуша (див. Додаток Т);

- аркуша (див. Додаток Т) на якому розташовано: **тема** лабораторної роботи, **мета** лабораторної роботи, **розробка програми** (наведіть **умову** та **формулу**, які розглядаються в лабораторній роботі №18, див. п. п. 2.4);

- аркуша (див. Додаток Т) на якому розташовано: **створення інтерфейсу програми** (навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №18, див. п. п. 2.4.1);

- аркушів (див. Додаток Т) на яких розташовані: **лістинг програми** (навести **лістинг** програми, який розглядається в лабораторній роботі №18, див. п. п. 2.4.3), **висновки**.

Лабораторна робота №19

Тема: Текстові файли. Тип файл.

Мета роботи: Засвоєння початкових знань та придбання навичок, необхідних для розробки програмних продуктів комп`ютерного забезпечення в технологічних процесах обробки металів різанням.

Порядок виконання роботи

1. Вивчити теоретичні відомості.

2. Виконати алгоритм виконання лабораторної роботи №19.

3. Скласти звіт з лабораторної роботи №19 за наведеною формою (див. Додаток У).

Хід виконання роботи

1. Теоретичні відомості

Вивчити теоретичні відомості зі списку наведеної літератури.

2. Алгоритм виконання лабораторної роботи №19

2.1. Запуск програми Delphi

У системному меню «***Пуск***», операційної системи Windows, вибрати: **Пуск > Програми > Borland Delphi 7 > Delphi 7**.

2.2. Створення проєкту

Після запуску програми Delphi створюється за замовчуванням проєкт типу – **Application** (Програма).

Delphi дозволяє розробляти різні програми, які в Delphi називаються проєктами. У лабораторних роботах створюються проєкти – **Application** (Програма), під операційну систему Windows з графічним інтерфейсом.

Для створення нового проєкту в «***Головному***» меню **File** (Файл) вибрати: **New** (Нова) > **Application** (Програма) (рис. 2.1).

2.3. Збереження проєкту

Delphi для кожного проєкту створює декілька файлів. Щоб файли різних проєктів не переплуталися між собою, слід для кожного проєкту створювати свою **папку** з іменем, яке містить назву та номер лабораторної роботи (наприклад, лабораторна робота 19 – Lab_19).

Рекомендується імені проєкту надавати ім’я, яке містить номер лабораторної роботи (наприклад, проєкт для лабораторної роботи 19 – Project_19). **Файл проєкту** Project _19 потрібно зберегти в папці Lab_19.

Необхідно також зберегти **файл модуля вихідного коду**.

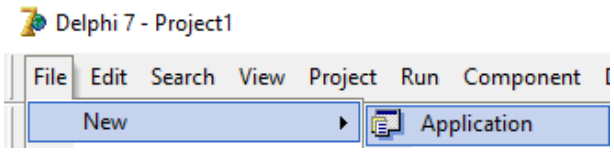


Рис. 2.1. Виберіть **New** > **Application** (Нова > Програма) в «***Головному***» меню **File** (Файл)

Рис. 2.2. Натиснути кнопку **Save all** (Зберегти все) на панелі інструментів **Standard** (Стандартна)

Рекомендується імені файлу модуля вихідного коду надавати ім’я, як у

проєкті (наприклад, Unit _19). Його також потрібно зберегти у папці Lab_19.

На панелі інструментів **Standard** (Стандартна) натиснути кнопку  **Save all** (Зберегти все) та зберегти **файли проєкту** та **файл модуля вихідного коду** згідно з вище сказаними рекомендаціями (рис. 2.2).

2.4. Розробка програми

Розробити програму обчислення значень змінних a та b, використовуючи формули:

$$a = y \cdot tg^3(x^2) + \sqrt{\frac{z^2}{y^2 + x^2}} \qquad b = \ln(y + x^2) + sin^2\left(\frac{z}{x}\right)$$

Реалізувати: читання з файлу вихідних даних: x = 2,1, y = 0,59, z = - 4,8; запис у файл результатів розрахунку значень змінних a и b.

2.4.1. Створення файлу, з якого будуть зчитуватимуться вихідні дані

Створіть у папці лабораторної роботи (Lab_19) текстовий файл у текстовому редакторі, наприклад, у Блокноті, та привласніть йому ім'я – Inputxyz (рис. 2.3). У створений текстовий файл введіть: у перший рядок – x=2.1, у другий рядок – y=0.59, у третій рядок – z=-4.8 (рис. 2.3). Увага! Для відокремлення цілої частини від дробової частини числа використовувати точку. Збережіть введені дані.

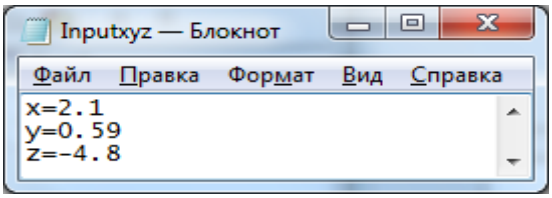


Рис. 2.3. Текстовий файл Inputxyz з введеними вихідними даними: x = 2.1, y = 0.59, z = -4.8

2.4.2. Створення інтерфейсу програми

Інтерфейс програми представлений на рис. 2.4. Файл, в який будуть записані результати розрахунку значень змінних a та b представлений на рис. 2.5. Він створюється у процесі виконання програми.

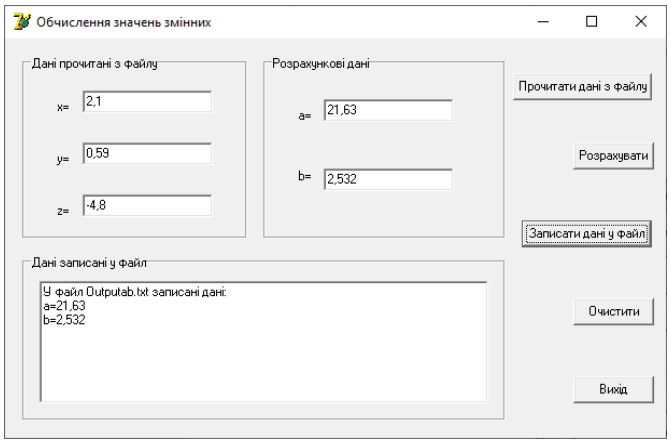


Рис. 2.4. Інтерфейс програми

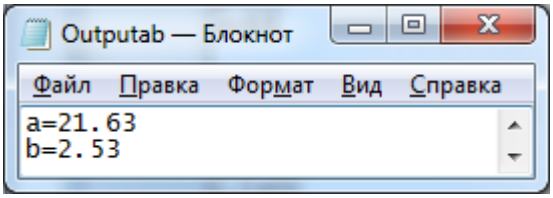


Рис. 2.5. Текстовий файл Outputab з записаними результатами розрахунку значень змінних a і b

2.4.2.1. Вікно форми Form1 (Форма1)

При створенні проєкту типу – **Application** (Програма), Delphi автоматично створює вікно форми «***Form1***» (Форма1) (рис. 2.6). Вікно форми «***Form1***» (Форма1) являє собою шаблон вікна розробляємої програми.

2.4.2.2. Розміщення компонентів на вікні форми Form1 (Форма1)

На вікні форми «***Form1***» (Форма1) розмістити компоненти (рис. 2.7) вказані в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) на «***Панелі компонентів***» на вкладці «***Standard***» (Стандартна) (рис. 2.8), вказати курсором миші на піктограму потрібного компонента (табл. 2.1);

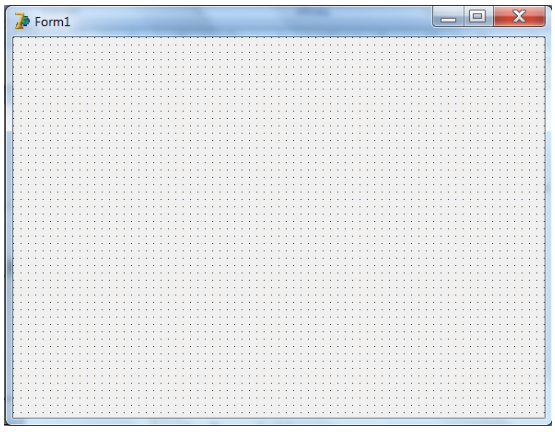


Рис. 2.6. Вікно форми «***Form1***» (Форма1)

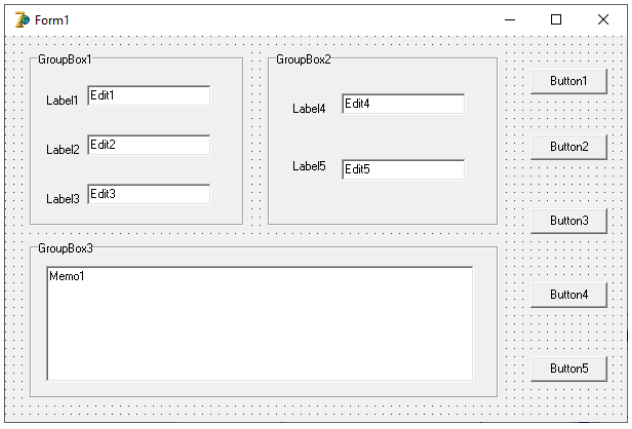


Рис. 2.7. Компоненти розміщені на вікні форми «***Form1***» (Форма1)

б) потім вказати курсором миші у довільному місці на формі «***Form1***» (Форма1) (рис. 2.7);

в) якщо необхідно, перетягніть компонент у потрібне місце на формі «***Form1***» (Форма1) (рис. 2.7).

Рис. 2.8. «***Панель компонентів***» на вкладці **Standard** (Стандартна)

Таблиця 2.1 – Компоненти та їх властивості для інтерфейсу програми

Компонент	**Піктограма**	**Властивість**	**Значення**
Form1	-	Caption	Обчислення значень змінних
		BorderStyle	bsSingle
Groupbox1		Caption	Дані прочитані з файлу
Groupbox2		Caption	Розрахункові дані
Groupbox3		Caption	Дані записані у файл
Edit1		Name	edtInputx
		Text	значення не вводимо, а тільки очистити поле
		ReadOnly	True
Edit2		Name	edtInputy
		Text	значення не вводимо, а тільки очистити поле
		ReadOnly	True
Edit3		Name	edtInputz
		Text	значення не вводимо, а тільки очистити поле
		ReadOnly	True
Edit4		Name	edtOutputa
		Text	значення не вводимо, а тільки очистити поле
		ReadOnly	True
Edit5		Name	edtOutputb
		Text	значення не вводимо, а тільки очистити поле
		ReadOnly	True
Label1		Caption	x=
Label2		Caption	y=
Label3		Caption	z=
Label4		Caption	a=

Продовження таблиці 2.1

Компонент	Піктограма	Властивість	Значення
Label5	A	Caption	b=
Memo1		Name	memOutput
		Lines	натиснути кнопку ... У з'явившемуся вікні «***String List Editor***» (Редактор рядків) очистити поле, потім натиснути кнопку **ОК**
		ReadOnly	True
Button1	OK	Name	btnRead
		Caption	Прочитати дані з файлу
Button2	OK	Name	btnCalculate
		Caption	Розрахувати
Button3	OK	Name	btnWrite
		Caption	Записати дані у файл
Button4	OK	Name	btnClear
		Caption	Очистити
Button5	OK	Name	btnExit
		Caption	Вихід

2.4.2.3. Зміна властивостей вікна форми Form1 (Форма1) та компонентів

Змінимо властивості вікна форми «***Form1***» (Форма1) та компонентів (рис. 2.9) згідно з вказаними властивостями в таблиці 2.1.

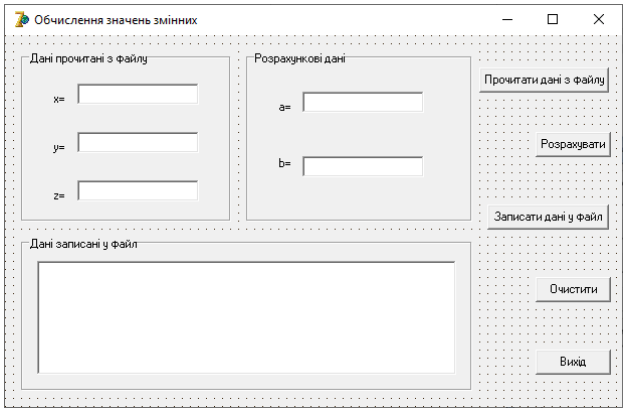


Рис. 2.9. Компоненти та вікно форми «***Form1***» (Форма1) із зміненими властивостями

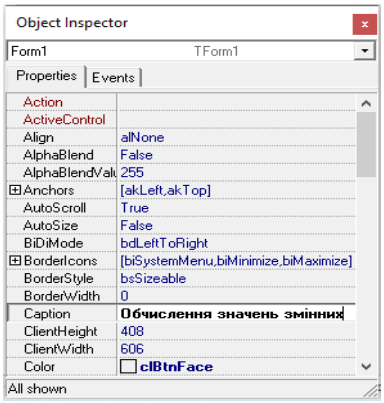


Рис. 2.10. Зміна значення властивості **Caption** (Заголовок) вікна форми «***Form1***» (Форма1) у вікні «***Object Inspector***» (Інспектор об'єктів) на **Обчислення значень змінних**

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.10) вибрати ім'я об'єкта (форми або компонентів) (табл. 2.1);

б) у лівій колонці вкладки «***Properties***» (Властивості) (рис. 2.10), де перелічені властивості об'єкта (форми чи компонентів) виділити властивість (табл. 2.1);

в) у правій колонці вкладки «***Properties***» (Властивості) (рис. 2.10), де вказано значення об'єкта (форми або компонентів) вибрати або ввести значення властивості (табл. 2.1).

2.4.2. Створення обробників подій

Створимо обробники подій для компонентів, вказаних у таблиці 2.2.

Таблиця 2.2 – Обробники подій компонентів для програми

Компонент	Подія	Ім'я обробника
btnRead	OnClick	btnReadClick
btnCalculate	OnClick	btnCalculateClick
btnWrite	OnClick	btnWriteClick
btnClear	OnClick	btnClearClick
btnExit	OnClick	btnExitClick

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.11) вибрати ім'я об'єкта (форми або компонента) (табл. 2.2);

б) у цьому ж вікні потрібно вибрати вкладку «***Events***» (Події) (рис. 2.11);

в) у лівій колонці вкладки «***Events***» (Події) (рис. 2.11), де перелічені імена подій, які може використовувати вибраний об’єкт (компонент), вибрати ім'я події (табл. 2.2);

г) для створення обробника події потрібно у правій колонці вкладки «***Events***» (Події) навпроти обраної раніше події зробити подвійне клацання мишею у полі імені обробника (рис. 2.11);

д) відкриється вікно «***Unit_19.pas***» (Редактор коду), до якого буде додано шаблон процедури обробника **btnExitClick** події **OnClick** (рис. 2.12), а у вікні «***Object Inspector***» (Інспектор об’єктів) поряд з ім'ям події з'явиться ім'я обробника (рис. 2.11);

е) у вікні «***Unit_19.pas***» (Редактор коду) між словами **begin** і **end** необхідно ввести інструкції, що реалізують обробку події (рис. 2.12). У лістингу програми, див. п. п 2.4.3, наведені тексти інструкцій: обробки події **OnClick** для кнопок: **Прочитати дані з файлу** (btnRead), **Розрахувати** (btnCalculate), **Записати дані у файл** (btnWrite), **Очистити** (btnClear) та **Вихід** (btnExit).

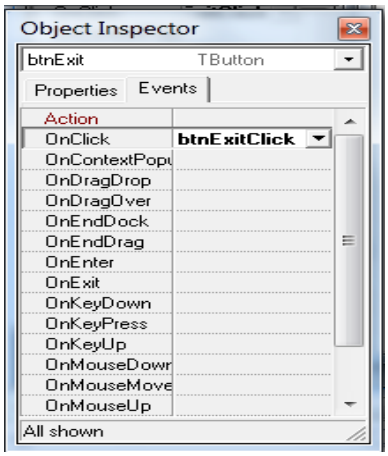


Рис. 2.11. Створення обробника **btnExitClick** події **OnClick** для компонента btnExit, на вкладці «***Events***» (Події) у вікні «***Object Inspector***» (Інспектор об'єктів)

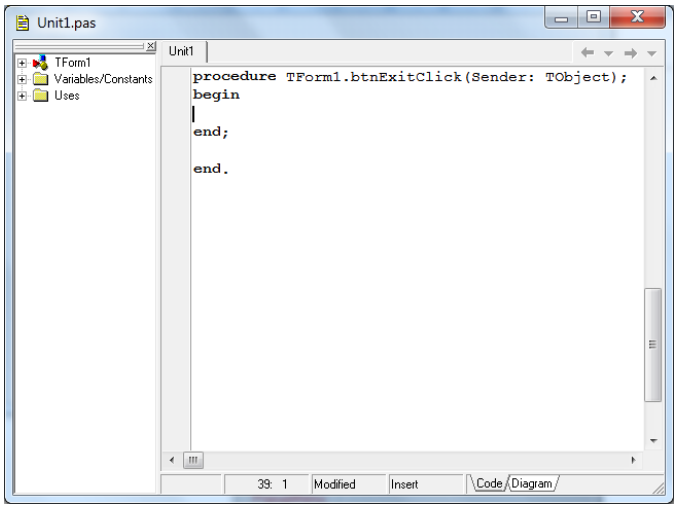


Рис. 2.12. Шаблон процедури обробника **btnExitClick** події **OnClick**, згенерований програмою Delphi

2.4.4. Створення вихідного коду

Вихідний код програми представлений у вигляді лістингу програми. Так як у вихідному коді використовуються математичні функції, то необхідно в розділі **uses** додати модуль **Math**.

Лістинг програми

```
unit Unit _19;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Math;
```

```
type
  TForm1 = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    GroupBox3: TGroupBox;
    edtInputx: TEdit;
    edtInputy: TEdit;
    edtInputz: TEdit;
    edtOutputa: TEdit;
    edtOutputb: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    memOutput: TMemo;
    btnRead: TButton;
    btnWrite: TButton;
    btnCalculate: TButton;
    btnClear: TButton;
    btnExit: TButton;
    procedure btnReadClick(Sender: TObject);
    procedure btnCalculateClick(Sender: TObject);
    procedure btnWriteClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
    procedure btnExitClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
```

```
procedure TForm1.btnReadClick(Sender: TObject);
var
  // Оголошення змінних
  Inputxyz : TextFile;
  v:string[2];
  x,y,z,n : Real;
begin
  // Зв'язування змінної з файлом
  AssignFile(Inputxyz,'Inputxyz.txt');
// Відкриття файлу
  Reset(Inputxyz);
  // Читання даних із файлу
  // x
  Readln(Inputxyz,v,n);
  x:=n;
  edtInputx.Text:=FloatToStr(n);
  // y
  Readln(Inputxyz,v,n);
  y:=n;
  edtInputy.Text:=FloatToStr(n);
  // z
  Readln(Inputxyz,v,n);
  z:=n;
  edtInputz.Text:=FloatToStr(n);
  // закриття файлу
  CloseFile(Inputxyz);
end;
procedure TForm1. btnCalculateClick(Sender: TObject);
var
  x, y, z, a, b : Real;   // Оголошення змінних
begin
    // Захист полів виводу
   // x, y, z
   if (edtInputx.Text='') or (edtInputy.Text='') or (edtInputz.Text='') then
   begin
```

```
  Application.MessageBox('Немає даних.'+#13#10+'Прочитайте дані з файлу.',
  'Помилка',mb_IconError+mb_OK);
  exit;
  end;
  // Захист полів вводу
  if (edtInputx.Text='0') or (edtInputy.Text='0') then
  begin
  Application.MessageBox(' Дані X або Y не можуть бути рівними нулю!' +
#13#10+'Прочитайте дані з файлу.','Помилка', mb_IconError+mb_OK);
  exit;
  end;
 // Дані прочитані з файлу
   x:= StrToFloat(edtInputx.Text);
   y:= StrToFloat(edtInputy.Text);
   z:= StrToFloat(edtInputz.Text);
  // Обробка даних
 a:=(y*power(Tan(Sqr(x)),3))+Sqrt(Sqr(z)/(Sqr(y)+Sqr(x)));
 b:=ln(y+sqr(x))+sqr(sin(z/y));
 // Вивід розрахункових даних з форматуванням
 edtOutputa.Text:=FloatToStrF(a,ffGeneral,4,2);
 edtOutputb.Text:=FloatToStrF(b,ffGeneral,4,2);
end;
procedure TForm1.btnWriteClick(Sender: TObject);
var
 // Оголошення змінних
 Outputab : TextFile;
 a, b : Real;
begin
  // Захист полів для виводу a, b
  if (edtOutputa.Text='') or (edtOutputb.Text='') then
  begin
  Application.MessageBox('Немає даних'+#13#10+'Прочитайте дані з файлу',
  'Помилка',mb_IconError+mb_OK);
  exit;
  end;
```

```
  // Ввід розрахункових даних
  a:=StrToFloat(edtOutputa.Text);
  b:=StrToFloat(edtOutputb.Text);
  // Відкриття файлу для запису даних
  AssignFile(Outputab,'Outputab.txt');
  // Створення файлу
  Rewrite(Outputab);
  // Запис даних у файл запису даних з форматуванням
  writeln(Outputab,'a=', a:4:2);
  writeln(Outputab,'b=', b:4:2);
  // Очищення рядка
  memOutput.Lines.Clear;
  memOutput.Lines.Add('У файл Outputab.txt записані дані:');
  // Дані, що виводяться з форматуванням
  memOutput.Lines.Add('a='+FloatToStrF(a,ffGeneral,4,2));
  memOutput.Lines.Add('b='+FloatToStrF(b,ffGeneral,4,2));
  // Закриття файлу запису даних
  CloseFile(Outputab);
end;
procedure TForm1.btnClearClick(Sender: TObject);
begin
// Очищення полів для читання даних
  edtInputx.Text:=' ';
  edtInputy.Clear;
  edtInputz.Clear;
  // Очищення полів розрахунку даних
  edtOutputa.Text:=' ';
  edtOutputb.Clear;
    memOutput.Clear;  // Очищення полів запису даних
end;
procedure TForm1.btnExitClick(Sender: TObject);
begin
   Close;    // Закриття програми
end;
end.
```

2.4.5. Компіляція програми

Процес компіляції складається із двох етапів. На першому етапі виконується перевірка тексту програми на відсутність синтаксичних помилок, а на другому – генерується виконувана програма (exe-файл).

Компіляція програми виконується з «***Головного***» меню **Project** (Проєкт) обравши команду **Compile Project** (Компілювати проєкт) (рис. 2.13) або натиснути клавіші **Ctrl+F9**. Повідомлення про помилки, попередження та підказки відображаються в нижній частині вікна редактора коду (рис. 2.14).

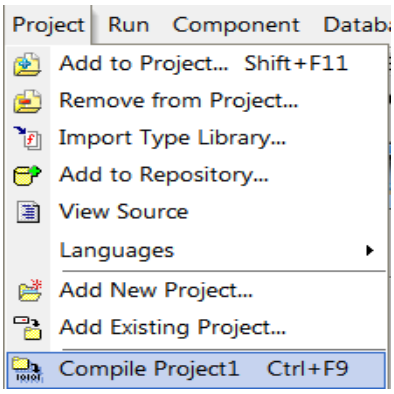


Рис. 2.13. Вибір команди **Compile Project** (Компілювати проєкт) з «***Головного***» меню **Project** (Проєкт)

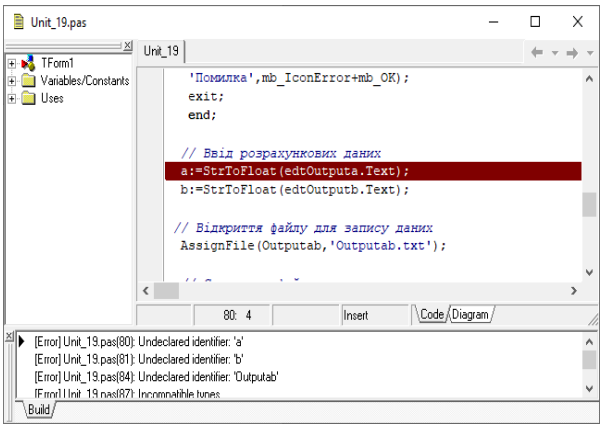


Рис. 2.14. Повідомлення компілятора про виявлені помилки

Процес усунення помилок має ітераційний характер. Зазвичай, спочатку усуваються найбільш очевидні помилки, наприклад, декларуються неоголошені змінні. Після чергового внесення змін до тексту програми виконується повторна компіляція. Слід зважати на те, що компілятор не завжди може точно локалізувати помилку. Тому аналізуючи фрагмент програми який на думку компілятора містить помилку, потрібно звертати увагу не тільки на той фрагмент коду, на який компілятор встановив курсор, але й на той, який знаходиться у попередньому рядку.

Якщо у програмі немає синтаксичних помилок, компілятор створює виконуваний файл програми. Ім'я виконуваного файлу таке ж, як і у файлу проєкту, а розширення – exe. Delphi поміщає виконуваний файл у той самий каталог, де знаходиться файл проєкту.

2.4.6. Запуск програми

Запуск програми може виконуватися з «***Головного***» меню **Run** (Запуск) обравши команду **Run** (Запуск) або натиснути на кнопці  **Run** (Запуск) на панелі інструментів «***Debug***» (Налагодження) (рис. 2.15) або натиснути клавішу **F9.**

Рис. 2.15. Панель інструментів «***Debug***» (Налагодження)

2.4.7. Помилки часу виконання

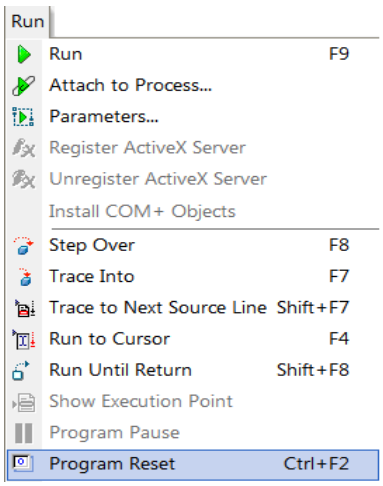


Рис. 2.16. Вибір команди **Program Reset** (Скидання програми) із «***Головного***» меню **Run** (Запуск)

Під час роботи програми можуть виникати помилки, які називаються **помилками часу виконання** (run-time errors) або **винятками** (exceptions).

У більшості випадків причинами винятків є невірні вихідні дані. Після виникнення таких помилок програма перестане робити. Необхідно перервати неробочу програму, вибравши із «***Головного***» меню **Run** (Запуск) команду **Program Reset** (Скидання програми) (рис. 2.16).

2.4.8. Захист полів вводу

Для виключення помилок, при вводі даних у поля вводу необхідно передбачити у вихідному коді програми захист полів вводу.

У лабораторній роботі №19 передбачимо такі види захисту полів вводу (див. п. п. 2.4.3):

- захисту полів вводу від вводу порожнього рядка;
- захисту полів вводу від вводу цифри «0» нуль;
- захисту полів вводу від вводу букв та деяких символів.

3. Зміст звіту з лабораторної роботи №19

Звіт з лабораторної роботи №19 складається з:

- титульного аркуша (див. Додаток У);

- аркуша (див. Додаток У) на якому розташовано: **тема** лабораторної роботи, **мета** лабораторної роботи, **розробка програми** (наведіть **умову** та **формулу**, які розглядаються в лабораторній роботі №19, див. п. п. 2.4);

- аркуша (див. Додаток У) на якому розташовано: **створення інтерфейсу програми** (навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №19, див. п. п. 2.4.1);

- аркушів (див. Додаток У) на яких розташовані: **лістинг програми** (навести **лістинг** програми, який розглядається в лабораторній роботі №19, див. п. п. 2.4.3) та **висновки**.

Запитання з самоконтролю до розділу 6

1. Що називається процедурою у програмуванні?
2. Що називається функцією у програмуванні?
3. Що таке фактичний параметр?
4. Що таке формальний параметр?
5. Що таке тип-файл?
6. Що загального в процедурах Reset і Rewrite?
7. Чим відрізняються процедури Reset і Rewrite?

РОЗДІЛ 7. СТВОРЕННЯ ГРАФІЧНИХ МОДЕЛЕЙ

Лабораторна робота №20

Тема: Графічні компоненти. Компонент Image.

Мета роботи: Засвоєння початкових знань та придбання навичок, необхідних для розробки програмних продуктів комп`ютерного забезпечення в технологічних процесах обробки металів різанням.

Порядок виконання роботи

1. Вивчити теоретичні відомості.
2. Виконати алгоритм виконання лабораторної роботи №20.
3. Скласти звіт з лабораторної роботи №20 за наведеною формою (див. Додаток Ф).

Хід виконання роботи:

1. Теоретичні відомості

Вивчити теоретичні відомості зі списку наведеної літератури.

2. Алгоритм виконання лабораторної роботи №20

2.1. Запуск програми Delphi

У системному меню «***Пуск***», операційної системи Windows, вибрати: **Пуск > Програми > Borland Delphi 7 > Delphi 7**.

2.2. Створення проєкту

Після запуску програми Delphi створюється за замовчуванням проєкт типу – **Application** (Програма). Delphi дозволяє розробляти різні програми, які в Delphi називаються проєктами. У лабораторних роботах створюються проєкти – **Application** (Програма), під операційну систему Windows з графічним інтерфейсом. Для створення нового проєкту, виберіть в «***Головному»*** меню **File** (Файл) команду **New** > **Application** (Нова > Програма) (рис. 2.1).

2.3. Збереження проєкту

Delphi для кожного проєкту створює декілька файлів. Щоб файли різних проєктів не переплуталися між собою, слід для кожного проєкту створювати свою **папку** з іменем, яке містить назву та номер лабораторної роботи (наприклад, лабораторна робота 20 – Lab_20).

Рекомендується імені проєкту надавати ім'я, яке містить номер лабораторної роботи (наприклад, проєкт для лабораторної роботи 20 – Project_20). **Файл проєкту** Project _20 потрібно зберегти в папці Lab_20.

Необхідно також зберегти **файл модуля вихідного коду**.

Рекомендується імені файлу модуля вихідного коду надавати ім'я, як у проєкті (наприклад, Unit _20). Його також потрібно зберегти у папці Lab_20.

На панелі інструментів «***Standard***» (Стандартна) натиснути кнопку  **Save all** (Зберегти все) та зберегти **файли проєкту** та **файл модуля вихідного коду** згідно з вище сказаними рекомендаціями (рис. 2.2).

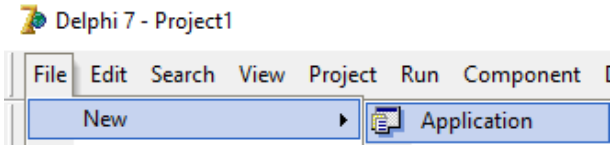


Рис. 2.1. Виберіть **New** > **Application** (Нова > Програма) в «***Головному***» меню **File** (Файл)

Рис. 2.2. Натиснути кнопку  **Save all** (Зберегти все) на панелі інструментів «***Standard***» (Стандартна)

2.4. Розробка програми

Розробити програму обчислення вартості поїздки на автомобілі на дачу (туди і назад). Вихідними даними є: відстань до дачі (туди і назад) у км; кількість бензину, яке споживає автомобіль на 100 км пробігу в літрах; ціна одного літра бензину в гривнях.

Приймаємо позначення: rast – відстань на дачу (туди і назад) у км;
potr - кількість бензину, яке споживає автомобіль на 100 км пробігу в літрах;
cena - ціна одного літра бензину в гривнях.

Складаємо формулу:

$$sum=2\cdot(potr/100)\cdot rast\cdot cena$$

Для наочного представлення призначення програми додати до неї

графічний компонент (малюнок, картинку або схему).

Схема поїздки на дачу (туди і назад) для використання у програмі представлена на рис. 2.3.

Рис. 2.3. Схема поїздки на дачу (туди і назад)

2.4.1. Створення інтерфейсу програми

Інтерфейс програми представлений на рис. 2.4. Вивід повідомлень програми представлено на рис. 2.5 (а, б, в).

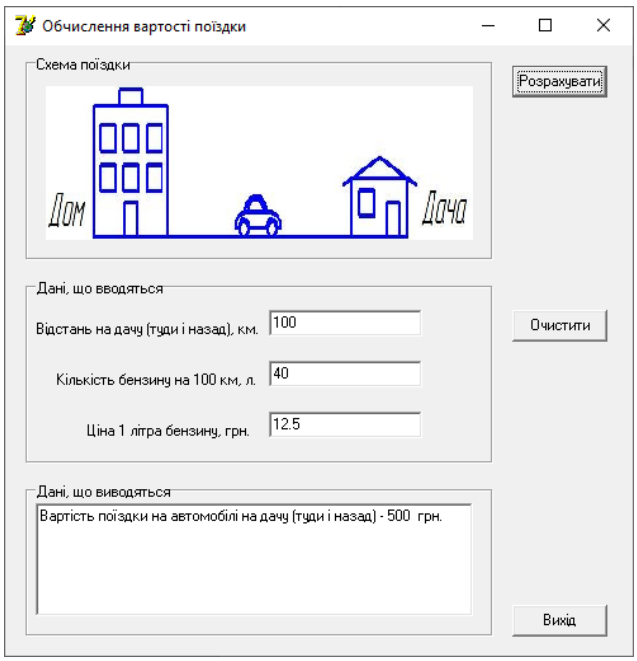
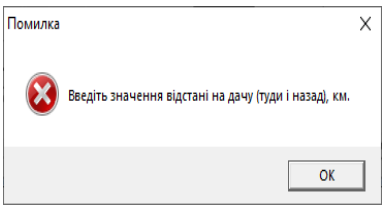
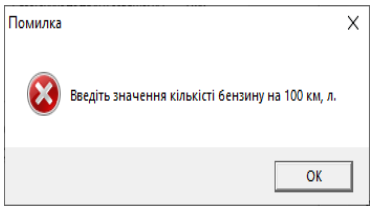


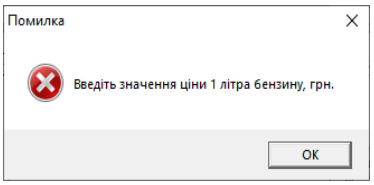
Рис. 2.4. Інтерфейс програми



а



б



в

Рис. 2.5. Вивід повідомлень програми: а, б, в.

2.4.1.1. Вікно форми Form1 (Форма1)

При створенні проєкту типу – **Application** (Програма), Delphi автоматично

створює вікно форми «***Form1***» (Форма1) (рис. 2.6). Вікно форми «***Form1***» (Форма1) являє собою шаблон вікна розробляємої програми.

2.4.1.2. Розміщення компонентів на вікні форми Form1 (Форма1)

На вікні форми «***Form1***» (Форма1) розмістити компоненти (рис. 2.7) вказані в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) на «***Панелі компонентів***» на вкладці «***Standard***» (Стандартна) (рис. 2.8, а) та на вкладці «***Additional***» (Додатково) (рис. 2.8, б), вказати курсором миші на піктограму потрібного компонента (табл. 2.1);

б) потім вказати курсором миші у довільному місці на формі «***Form1***» (Форма1) (рис. 2.7);

в) якщо необхідно, перетягніть компонент у потрібне місце на формі «***Form1***» (Форма1) (рис. 2.7).

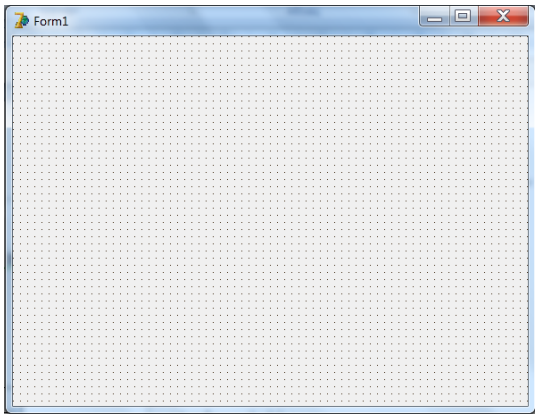


Рис. 2.6. Вікно форми «***Form1***» (Форма1)

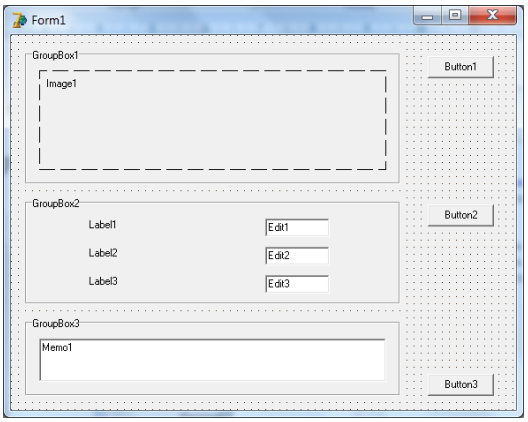


Рис. 2.7. Компоненти розміщені на вікні форми «***Form1***» (Форма1)

а

б

Рис. 2.8. «***Панель компонентів***»: а) вкладка «***Standard***» (Стандартна), б) вкладка «***Additional***» (Додатково)

Таблиця 2.1 – Компоненти та їх властивості для інтерфейсу програми

Компонент	Піктограма	Властивість	Значення
Form1	-	Caption	Обчислення вартості поїздки
		BorderStyle	bsSingle
Groupbox1		Caption	Схема поїздки
Groupbox2		Caption	Дані, що вводяться
Groupbox3		Caption	Дані, що виводяться
Edit1		Name	edtInputRast
		Text	значення не вводимо, а тільки очистити поле
Edit2		Name	edtInputPotr
		Text	значення не вводимо, а тільки очистити поле
Edit3		Name	edtInputCena
		Text	значення не вводимо, а тільки очистити поле
Label1		Caption	Відстань до дачі (туди і назад), км.
Label2		Caption	Кількість бензину на 100 км, л.
Label3		Caption	Ціна 1 літра бензину, грн.
Memo1		Name	memOutput
		ReadOnly	True
		Lines	натиснути кнопку  . У з'явившемуся вікні очистити поле, потім натиснути кнопку **ОК**
Button1		Name	btnCalculate
		Caption	Розрахувати
Button2		Name	btnClear
		Caption	Очистити
Button3		Name	btnExit
		Caption	Вихід
Image1		Picture	натиснути кнопку  . У з'явившемуся вікні «***Picture Editor***» (Редактор малюнку) натиснути кнопку **Load** (Завантажити) (рис. 2.9). У з'явившемуся вікні «***Load Picture***» (Завантажити малюнок) вибрати файл малюнка та натиснути кнопку **Відкрити** (рис. 2.10).

Продовження таблиці 2.1

Компонент	Піктограма	Властивість	Значення
Image1		Picture	Продовження. У вікні «***Picture Editor***» (Редактор малюнку) з'явиться зображення завантажуваного малюнку (рис. 2.11). Потім натиснути кнопку **ОК** (рис. 2.11).
		Stretch	True

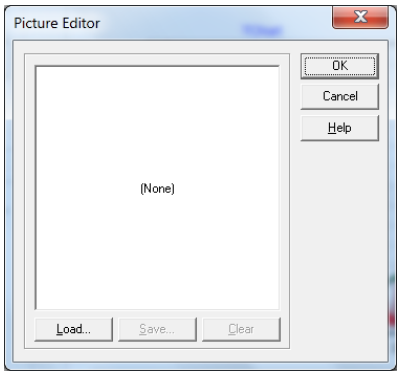


Рис. 2.9. Вікно «***Picture Editor***» (Редактор малюнка)

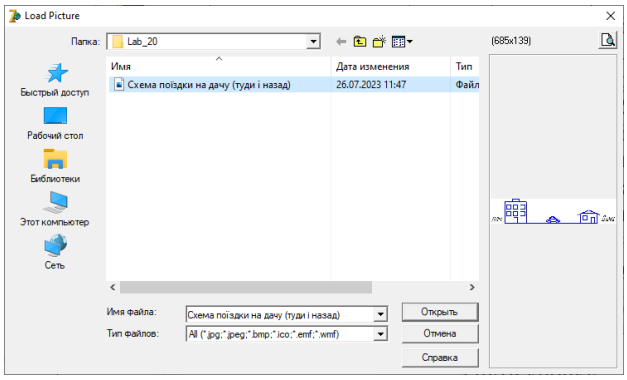


Рис. 2.10. Вікно «***Load Picture***» (Завантажити малюнок)

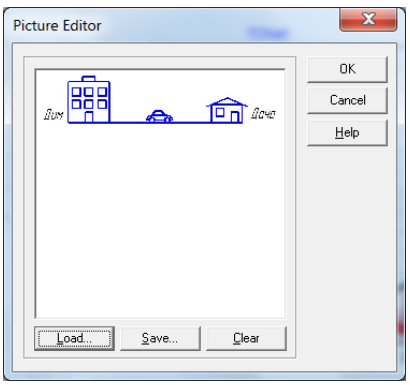


Рис. 2.11. У вікні «***Picture Editor***» (Редактор малюнка) з'явиться зображення завантажуваного малюнку

2.4.1.3. Зміна властивостей вікна форми Form1 (Форма1) та компонентів

Змінимо властивості вікна форми «***Form1***» (Форма1) та компонентів (рис. 2.12) згідно з вказаними властивостями в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.13) вибрати ім'я об'єкта (форми або компонентів) (табл. 2.1);

б) у лівій колонці вкладки «***Properties***» (Властивості) (рис. 2.13), де перелічені властивості об'єкта (форми чи компонентів) виділити властивість (табл. 2.1);

в) у правій колонці вкладки «***Properties***» (Властивості) (рис. 2.13), де вказано значення об'єкта (форми або компонентів) вибрати або ввести значення властивості (табл. 2.1).

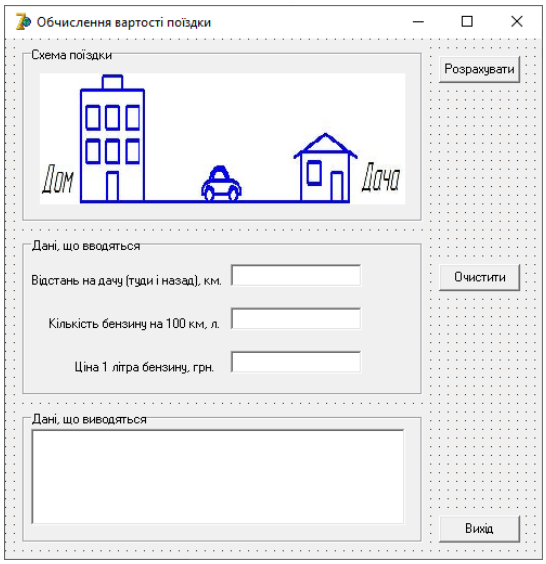


Рис. 2.12. Компоненти та вікно форми «***Form1***» (Форма1) із зміненими властивостями

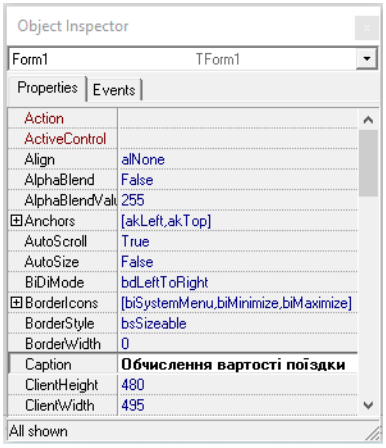


Рис. 2.13. Зміна значення властивості **Caption** (Заголовок) вікна форми «***Form1***» (Форма1) у вікні «***Object Inspector***» (Інспектор об'єктів) на **Обчислення вартості поїздки**

2.4.2. Створення обробників подій

Створимо обробники подій для компонентів, вказаних у таблиці 2.2.

Таблиця 2.2 – Обробники подій компонентів для програми

Компонент	Подія	Ім'я обробника
btnCalculate	OnClick	btnCalculateClick
btnClear	OnClick	btnClearClick
btnExit	OnClick	btnExitClick

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.14) вибрати ім'я об'єкта (форми або компонента) (табл. 2.2);

б) у цьому ж вікні потрібно вибрати вкладку «***Events***» (Події) (рис. 2.14);

в) у лівій колонці вкладки «***Events***» (Події) (рис. 2.14), де перелічені імена подій, які може використовувати вибраний об’єкт (компонент), вибрати ім'я події (табл. 2.2);

г) для створення обробника події потрібно у правій колонці вкладки «***Events***» (Події) навпроти обраної раніше події зробити подвійне клацання мишею у полі імені обробника (рис. 2.14);

д) відкриється вікно «***Unit_20.pas***» (Редактор коду), до якого буде додано шаблон процедури обробника **btnExitClick** події **OnClick** (рис. 2.15), а у вікні «***Object Inspector***» (Інспектор об’єктів) поряд з ім'ям події з'явиться ім'я обробника (рис. 2.14);

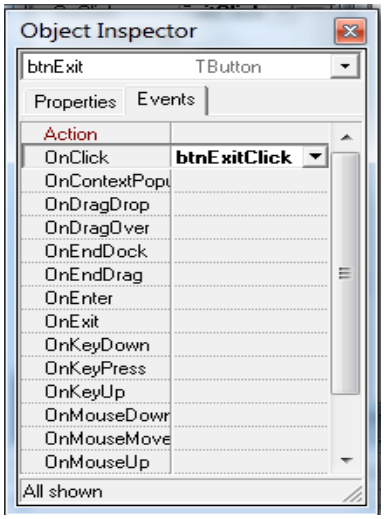


Рис. 2.14. Створення обробника **btnExitClick** події **OnClick** для компонента btnExit, на вкладці «***Events***» (Події) у вікні «***Object Inspector***» (Інспектор об'єктів)

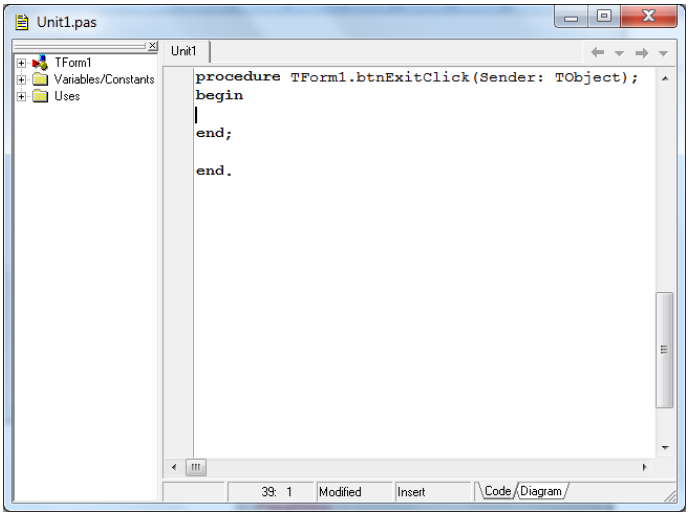


Рис. 2.15. Шаблон процедури обробника **btnExitClick** події **OnClick**, згенерований програмою Delphi

е) у вікні «***Unit_20.pas***» (Редактор коду) між словами **begin** і **end** необхідно ввести інструкції, що реалізують обробку події (рис. 2.15). У лістингу програми, див. п. п 2.4.3, наведені тексти інструкцій обробки події **OnClick** для кнопок: **Розрахувати** (btnCalculate), **Очистити** (btnClear) та **Вихід** (btnExit).

2.4.3. Створення вихідного коду

Вихідний код програми представлений у вигляді лістингу програми.

Лістинг програми

```
unit Unit_20;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls;
type
  TForm1 = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    Groupbox3: TGroupbox;
    Image1: TImage1;
    edtInputRast: TEdit;
    edtInputPotr: TEdit;
    edtInputCena: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    memOutput: TMemo;
    btnCalculate: TButton;
    btnClear: TButton;
    btnExit: TButton;
    procedure btnCalculateClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
    procedure btnExitClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
```

```
procedure TForm1. btnCalculateClick(Sender: TObject);
const
  z : Integer=100; // Оголошення констант
var
  // Оголошення змінних
  Sum, Potr, Rast, Cena : Real;
  code : integer;
begin
  // Ввід даних з захистом полів вводу
  // Відстань (Rast)
  val(edtInputRast.Text, Rast,code);
  if code<>0 then
  begin
  Application.MessageBox('Введіть значення відстані на дачу (туди і назад), км.',
  'Помилка',mb_IconError+mb_OK);
  edtInputRast.SetFocus;
  exit;
  end;
  // Кількість бензину (Potr)
  val(edtInputPotr.Text, Potr,code);
  if code<>0 then
  begin
  Application.MessageBox('Введіть значення кількісті бензину на 100 км, л.',
  'Помилка', mb_IconError+mb_OK);
  edtInputPotr.SetFocus;
  exit;
  end;
  // Ціна (Cena)
  val(edtInputCena.Text, Cena,code);
  if code<>0 then
  begin
  Application.MessageBox('Введіть значення ціни 1 літра бензину, грн.',
  'Помилка', mb_IconError+mb_OK);
  edtInputCena.SetFocus;
  exit;
```

```
  end;
  // Обробка даних
  Sum:= Rast*(Potr/z)*Cena;
  // Очищення рядків
  memOutput.Lines.Clear;
  // Вивід даних з форматуванням
  memOutput.Lines.Add('Вартість поїздки на автомобілі на дачу ' +
'(туди і назад) - ' + FloatToStrF(Sum,ffGeneral,6,2)+'  грн.');
end;
procedure TForm1.btnClearClick(Sender: TObject);
begin
  // Очищення полів вводу даних
  edtInputRast.Text:= ' ';
  edtInputPotr.Text:= ' ';
  edtInputCena.Text:= ' ';
  // Очищення поля виводу даних
  memOutput.Lines.Clear;
end;
procedure TForm1.btnExitClick(Sender: TObject);
begin
   Application.Terminate;  // Закриття програми
end;
end.
```

2.4.4. Компіляція програми

Процес компіляції складається із двох етапів. На першому етапі виконується перевірка тексту програми на відсутність синтаксичних помилок, а на другому – генерується виконувана програма (exe-файл).

Компіляція програми виконується з «***Головного***» меню **Project** (Проєкт) обравши команду **Compile Project** (Компіляція проєкту) (рис. 2.16) або натиснути клавіші **Ctrl+F9**. Повідомлення про помилки, попередження та підказки відображаються в нижній частині вікна редактора коду (рис. 2.17).

Процес усунення помилок має ітераційний характер. Зазвичай, спочатку усуваються найбільш очевидні помилки, наприклад, декларуються неоголошені змінні. Після чергового внесення змін до тексту програми виконується

повторна компіляція. Слід зважати на те, що компілятор не завжди може точно локалізувати помилку. Тому аналізуючи фрагмент програми який на думку компілятора містить помилку, потрібно звертати увагу не тільки на той фрагмент коду, на який компілятор встановив курсор, але й на той, який знаходиться у попередньому рядку.

Якщо у програмі немає синтаксичних помилок, компілятор створює виконуваний файл програми. Ім'я виконуваного файлу таке ж, як і у файлу проєкту, а розширення – exe. Delphi поміщає виконуваний файл у той самий каталог, де знаходиться файл проєкту.

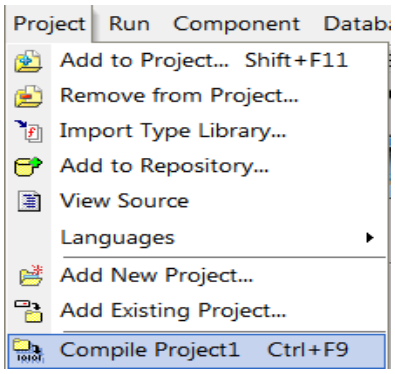


Рис. 2.16. Вибір команди **Compile Project** (Компіляція проєкту) з «***Головного***» меню **Project** (Проєкт)

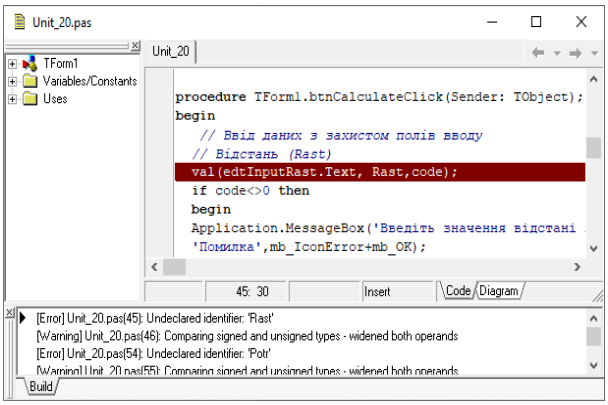


Рис. 2.17. Повідомлення компілятора про виявлені помилки

2.4.5. Запуск програми

Запуск програми може виконуватися з «***Головного***» меню **Run** (Запуск) обравши команду **Run** (Запуск) або натиснути на кнопці  **Run** (Запуск) на панелі інструментів «***Debug***» (Налагодження) (рис. 2.18) або натиснути клавішу **F9.**

Рис. 2.18. Панель інструментів «***Debug***» (Налагодження)

2.4.6. Помилки часу виконання

Під час роботи програми можуть виникати помилки, які називаються **помилками часу виконання** (run-time errors) або **винятками** (exceptions).

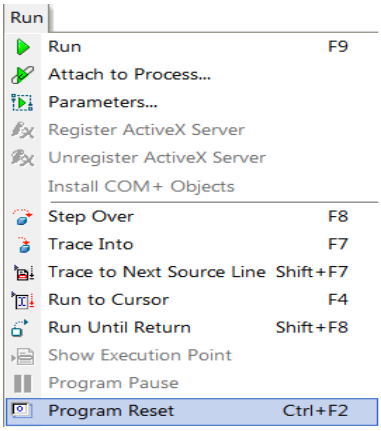


Рис. 2.19. Вибір команди **Program Reset** (Скидання програми) із «***Головного***» меню **Run** (Запуск)

У більшості випадків причинами винятків є невірні вихідні дані. Після виникнення таких помилок програма перестане робити.

Необхідно перервати неробочу програму, вибравши із «***Головного***» меню **Run** (Запуск) команду **Program Reset** (Скидання програми) (рис. 2.19).

2.4.7. Захист полів вводу

Для виключення помилок, при вводі даних у поля вводу необхідно передбачити у вихідному коді програми захист полів вводу.

У лабораторній роботі №20 використовуємо рядкову функцію Val, яка дозволяє використовувати такі види захисту поля вводу (див. п. п. 2.4.3):

- захист поля вводу від вводу порожнього рядка;
- захист поля вводу від вводу цифри «0» нуль;
- захист поля вводу від вводу літер та деяких символів.

3. Зміст звіту з лабораторної роботи №20

Звіт з лабораторної роботи №20 складається з:

- титульного аркуша (див. Додаток Ф);
- аркуша (див. Додаток Ф) на якому розташовано: **тема** лабораторної роботи, **мета** лабораторної роботи, **розробка програми** (наведіть **умову** та **формулу**, які розглядаються в лабораторній роботі №20, див. п. п. 2.4);
- аркуша (див. Додаток Ф) на якому розташовано: **створення інтерфейсу програми** (навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №20, див. п. п. 2.4.1);
- аркушів (див. Додаток Ф) на яких розташовані: **лістинг програми** (навести **лістинг** програми, який розглядається в лабораторній роботі №20, див. п. п. 2.4.3) та **висновки.**

Лабораторна робота №21

Тема: Графічні компоненти. Компонент Chart

Мета роботи: Засвоєння початкових знань та придбання навичок, необхідних для розробки програмних продуктів комп`ютерного забезпечення в технологічних процесах обробки металів різанням.

Порядок виконання роботи

1. Вивчити теоретичні відомості.
2. Виконати алгоритм виконання лабораторної роботи №21.
3. Скласти звіт з лабораторної роботи №21 за наведеною формою (див. Додаток Х).

Хід виконання роботи:

1. Теоретичні відомості

Вивчити теоретичні відомості зі списку наведеної літератури.

2. Алгоритм виконання лабораторної роботи №21

2.1. Запуск програми Delphi

У системному меню «***Пуск***», операційної системи Windows, вибрати: **Пуск > Програми > Borland Delphi 7 > Delphi 7**.

2.2. Створення проєкту

Після запуску програми Delphi створюється за замовчуванням проєкт типу – **Application** (Програма). Delphi дозволяє розробляти різні програми, які в Delphi називаються проєктами. У лабораторних роботах створюються проєкти – **Application** (Програма), під операційну систему Windows з графічним інтерфейсом.

Для створення нового проєкту в «***Головному»*** меню **File** (Файл) вибрати: **New** (Нова) > **Application** (Програма) (рис. 2.1).

2.3. Збереження проєкту

Delphi для кожного проєкту створює декілька файлів. Щоб файли різних проєктів не переплуталися між собою, слід для кожного проєкту створювати свою **папку** з іменем, яке містить назву та номер лабораторної роботи (наприклад, лабораторна робота 21 – Lab_21).

Рекомендується імені проєкту надавати ім'я, яке містить номер лабораторної роботи (наприклад, проєкт для лабораторної роботи 21 – Project_21). **Файл проєкту** Project _21 потрібно зберегти в папці Lab_21.

Необхідно також зберегти **файл модуля вихідного коду**.

Рекомендується імені файлу модуля вихідного коду надавати ім'я, як у проєкті (наприклад, Unit _21). Його також потрібно зберегти у папці Lab_21.

На панелі інструментів «***Standard***» (Стандартна) натиснути кнопку  **Save all** (Зберегти все) та зберегти **файли проєкту** та **файл модуля вихідного коду** згідно з вище сказаними рекомендаціями (рис. 2.2).

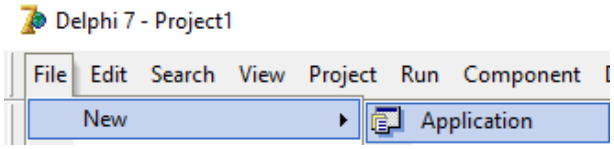


Рис. 2.1. Виберіть **New** > **Application** (Нова > Програма) в «***Головному***» меню **File** (Файл)

Рис. 2.2. Натиснути кнопку  **Save all** (Зберегти все) на панелі інструментів «***Standard***» (Стандартна)

2.4. Розробка програми

Розробити програму для обчислення значення шорсткості Ra поверхні, обробленою шліфуванням, за формулою

$$Ra = c \cdot t^z,$$

при зміні глибини різання t від 0,01 до 0,04 мм з кроком 0,01 мм, відповідно змінюється показник ступеня z від 0,4 до 0,6 з кроком 0,05. Значення коефіцієнта c = 2. Для наочного представлення залежності функції від аргументу додати до програми графічний компонент Chart і побудувати графік.

2.4.1. Створення інтерфейсу програми

Інтерфейс програми представлений на рис. 2.3. Вивід повідомлень програми представлено на рис. 2.4 (а, б, в).

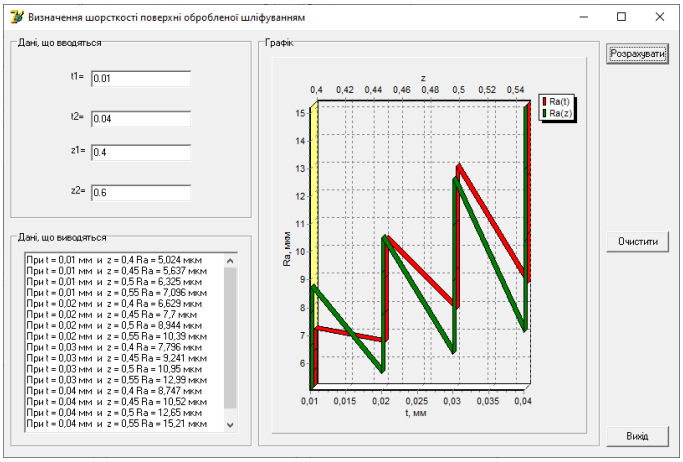
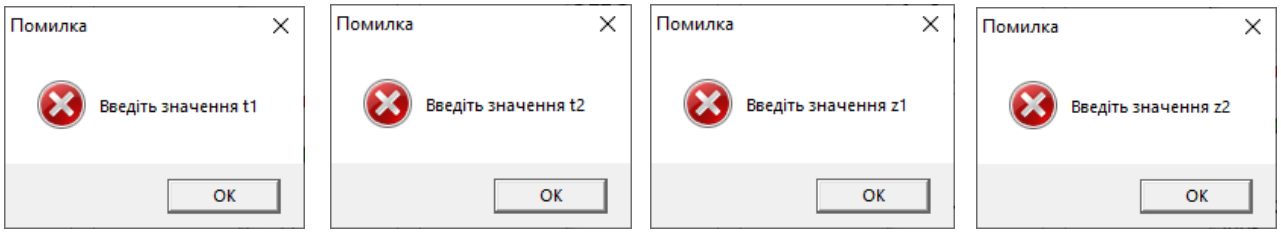


Рис. 2.3. Інтерфейс програми



а б в г

Рис.2.4. Вивід повідомлень програми: а, б, в, г

2.4.1.1. Вікно форми Form1 (Форма1)

При створенні проєкту типу – **Application** (Програма), Delphi автоматично створює вікно форми «***Form1***» (Форма1) (рис. 2.5). Вікно форми «***Form1***» (Форма1) являє собою шаблон вікна розробляємої програми.

2.4.1.2. Розміщення компонентів на вікні форми Form1 (Форма1)

На вікні форми «***Form1***» (Форма1) розмістити компоненти (рис. 2.6) вказані в таблиці 2.1.

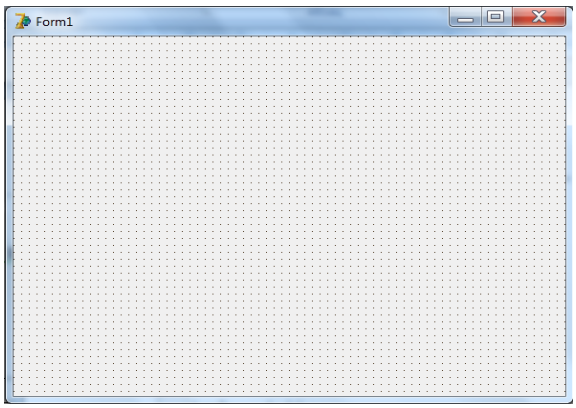


Рис. 2.5. Вікно форми «***Form1***» (Форма1)

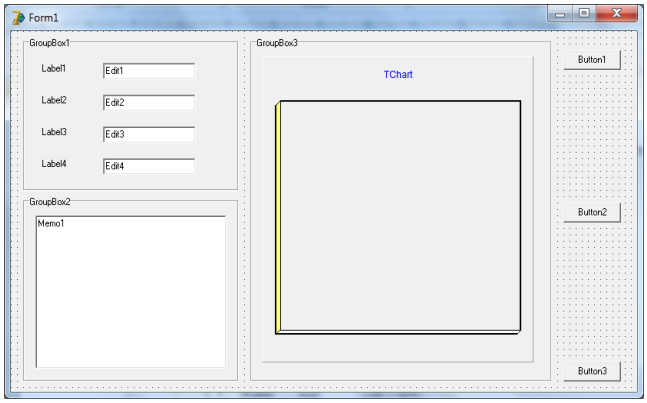


Рис. 2.6. Компоненти розміщені на вікні форми «***Form1***» (Форма1)

Для цього необхідно виконати наступні дії:

а) на «***Панелі компонентів***» на вкладці «***Standard***» (Стандартна) (рис. 2.7, а) та на вкладці «***Additional***» (Додатково) (рис. 2.7, б), вказати курсором миші на піктограму потрібного компонента (табл. 2.1);

б) потім вказати курсором миші у довільному місці на формі «***Form1***» (Форма1) (рис. 2.6);

в) якщо необхідно, перетягніть компонент у потрібне місце на формі «***Form1***» (Форма1) (рис. 2.6).

а

б

Рис. 2.7. «***Панель компонентів***»: а) вкладка «***Standard***» (Стандартна), б) вкладка «***Additional***» (Додатково)

Таблиця 2.1 – Компоненти та їх властивості для інтерфейсу програми

Компонент	**Піктограма**	**Властивість**	**Значення**
Form1	-	Caption	Визначення шорсткості поверхні обробленої шліфуванням
		BorderStyle	bsSingle
Groupbox1		Caption	Дані, що вводяться
Groupbox2		Caption	Дані, що виводяться

Продовження таблиці 2.1

Компонент	Піктограма	Властивість	Значення
Groupbox3		Caption	Графік
Edit1		Name	edtInputt1
		Text	значення не вводимо, а тільки очистити поле
Edit2		Name	edtInputt2
		Text	значення не вводимо, а тільки очистити поле
Edit3		Name	edtInputz1
		Text	значення не вводимо, а тільки очистити поле
Edit4		Name	edtInputz2
		Text	значення не вводимо, а тільки очистити поле
Label1		Caption	t1=
Label2		Caption	t2=
Label3		Caption	z1=
Label4		Caption	z2=
Memo1		Name	memOutput
		ScrollBars	ssVertical
		Lines	натиснути кнопку …. У з'явившемуся вікні очистити поле, потім натиснути кнопку кнопку **ОК**
		ReadOnly	True
Button1		Name	btnCalculate
		Caption	Розрахувати
Button2		Name	btnClear
		Caption	Очистити
Button3		Name	btnExit
		Caption	Вихід
Chart1		SeriesList	натиснути кнопку …. У з'явившемуся вікні «***Editing Chart1***» (Редактор) на вкладці Series (Серія) натиснути кнопку **Add** (рис. 2.8). У з'явившемуся вікні «***TeeChart Gallery***» (Галерея) на вкладці Standard (Стандартна) вибрати вид графіка **Line** та натиснути кнопку **ОК** (рис. 2.9).

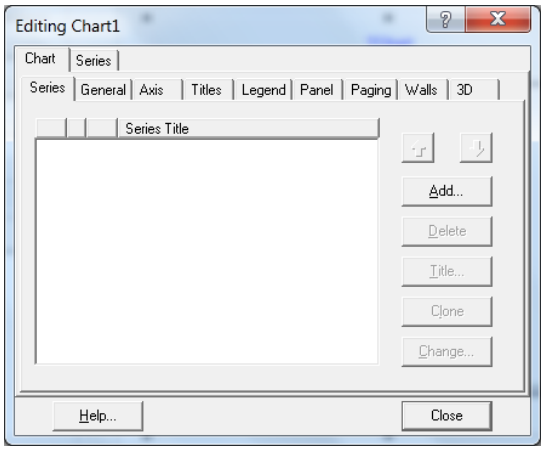


Рис. 2.8. Вікно «***Editing Chart1***» (Редактор)

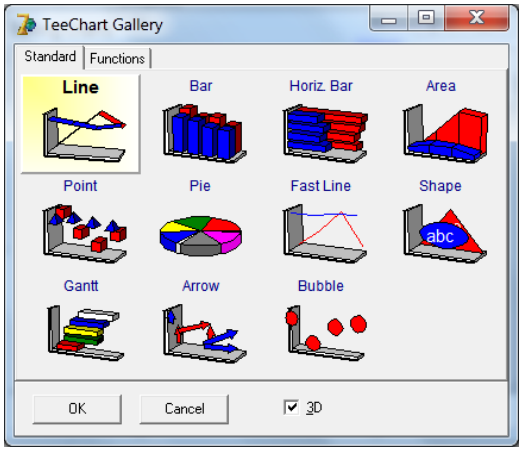


Рис. 2.9. Вікно «***TeeChart Gallery***» (Галерея)

Продовження таблиці 2.1

Компонент	Піктограма	Властивість	Значення
Chart1		SeriesList	Продовження. У вікні «***Editing Chart1***» (Редактор) на вкладці «***Series***» (Серія) з'явиться перший вибраний графік з написом Series1 (Серія1) (рис. 2.10). Для вибору другого графіка з написом Series2 (Серія2) необхідно повторити раніше розглянуті дії (рис. 2.11). Для зміни напису графіка **Series1** (Серія1) необхідно у вікні «***Editing Chart1***» (Редактор) на вкладці «***Series1***» (Серія1) вибрати перший графік і натиснути кнопку **Title** (Напис) (рис. 2.10). У з'явившемуся вікні «***Change Series Title***» (Змінити напис) ввести новий напис - **Ra(t)** (рис. 2.12). Для зміни напису графіка **Series2** (Серія2) необхідно повторити раніше розглянуті дії (рис. 2.11 і рис. 2.13). У вікні «***Editing Chart1***» (Редактор) з'явились нові написи графіків (рис. 2.14).

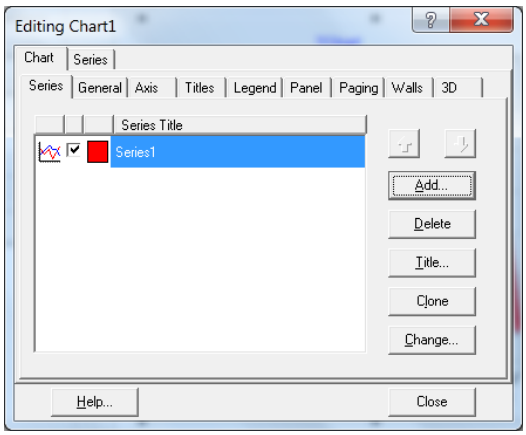


Рис. 2.10. Вікно «***Editing Chart1***» з першим вибраним графіком

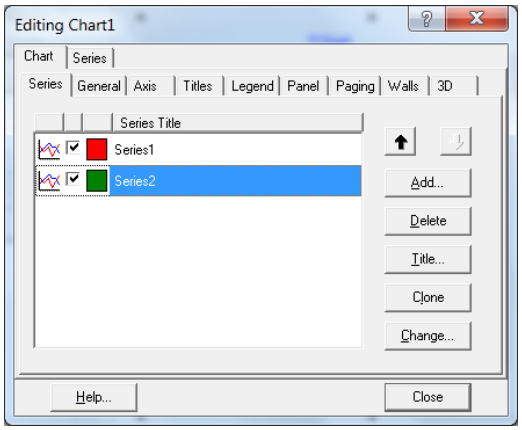


Рис. 2.11. Вікно «***Editing Chart1***» з другим вибраним графіком

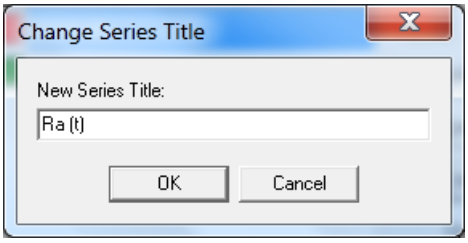


Рис. 2.12. У вікні «***Change Series Title***» (Змінити напис) ввести напис - Ra(t)

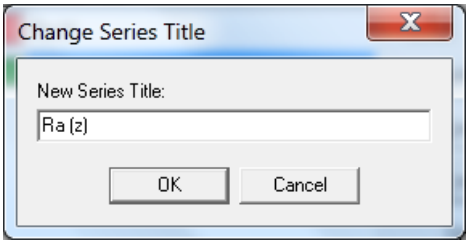


Рис. 2.13. У вікні «***Change Series Title***» (Змінити напис) ввести напис - Ra(z)

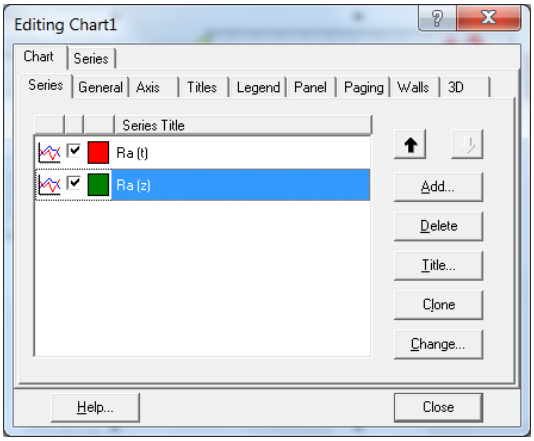


Рис. 2.14. Вікно «***Editing Chart1***» з новими написами графіків

Продовження таблиці 2.1

Компонент	**Піктограма**	**Властивість**	**Значення**
Chart1		SeriesList	Продовження. Для присвоєння напису осям графіків необхідно у вікні «***Editing Chart1***» (Редактор) на вкладці «***Axis***» (Вісь) увімкнути кнопку **Left** (Зліва), потім вибрати ще одну вкладку «***Title***» (Надпис) і у поле «***Title***» (Напис) ввести напис **Ra, мкм** (рис. 2.15)

Продовження таблиці 2.1

Компонент	Піктограма	Властивість	Значення
Chart1		SeriesList	Продовження. У полі «***Axis***» (Вісь) увімкнути кнопку **Bottom** (Внизу), а у поле «***Title***» (Напис) ввести напис **t, мм** (рис. 2.16). У полі «***Axis***» (Вісь) увімкнути кнопку **Top** (Зверху), а у полі «***Title***» (Напис) ввести напис **z** (рис. 2.17). Для вимкнення напису **TChart** (Графік) необхідно на вкладці «***Titles***» (Напис) прибрати «прапорець» біля напису «***Visible***» (Видимий) (рис. 2.18). Для розміщення додаткової осі зверху на графіку, якій присвоєно напис **z** необхідно перейти на вкладку «***Series***» (Серія) у вікні «***Editing Chart1***» (Редактор) (рис. 2.19). З випадаючого списку вибрати графік з написом **Ra(z)** (рис. 2.19). Потім перейти на вкладку «***General***» (Загальна) і в полі «***Horizontal Axis***» (Горизонтальна вісь) увімкнути кнопку **Top** (Зверху) (рис. 2.19).

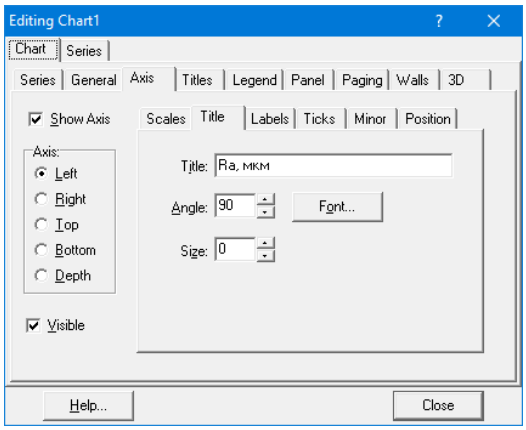


Рис. 2.15. У поле «***Title***» (Напис) ввести напис **Ra, мкм** на вкладці «***Title***» (Напис) при увімкненій кнопці **Left** (Зліва) на вкладці «***Axis***» (Вісь)

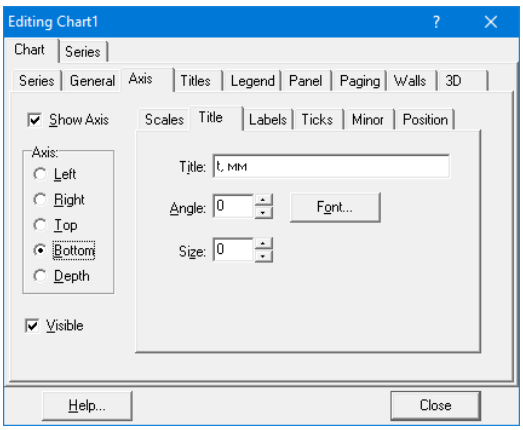


Рис. 2.16. У поле «***Title***» (Напис) ввести напис **t, мм** на вкладці «***Title***» (Напис) при увімкненій кнопці **Bottom** (Внизу) на вкладці «***Axis***» (Вісь)

Продовження таблиці 2.1

Компонент	Піктограма	Властивість	Значення
Chart1		SeriesList	Продовження. Після зміни потрібних параметрів натиснути кнопку **Close** (Закрити) (рис. 219). У компоненті «***Chart1***» (Графік1) будуть відображаються приблизні види представлення графіків (рис. 2.20).

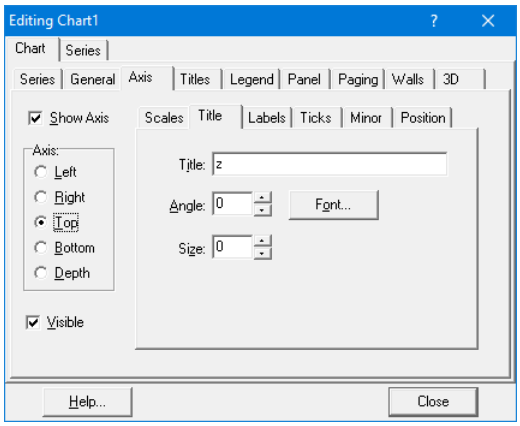


Рис. 2.17. У поле «***Title***» (Напис) ввести напис **z** на вкладці «***Title***» (Напис) при увімкненій кнопці **Top** (Зверху) на вкладці «***Axis***» (Вісь)

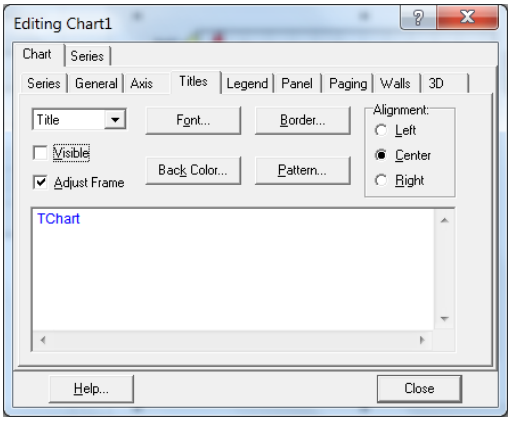


Рис. 2.18. Вимкнути напис **TChart** (Графік)

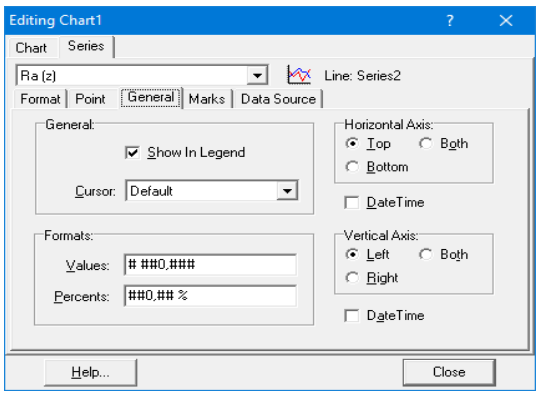


Рис. 2.19. У полі «***Horizontal Axis***» (Горизонтальна вісь) увімкнути кнопку Top (Зверху) на вкладці «***General***» (Загальна) при вибраному графіку з написом Ra(z)

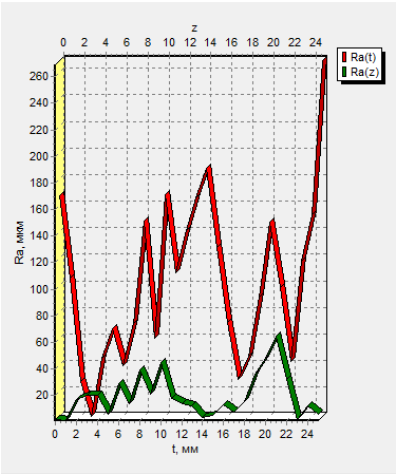


Рис. 2.20. У компоненті Chart1 (Графік) відображаються приблизні види представлення графіків

2.4.1.3. Зміна властивостей вікна форми Form1 (Форма1) та компонентів

Змінимо властивості вікна форми «***Form1***» (Форма1) та компонентів (рис. 2.21) згідно з вказаними властивостями в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.22) вибрати ім'я об'єкта (форми або компонентів) (табл. 2.1);

б) у лівій колонці вкладки «***Properties***» (Властивості) (рис. 2.22), де перелічені властивості об'єкта (форми чи компонентів) виділити властивість (табл. 2.1);

в) у правій колонці вкладки «***Properties***» (Властивості) (рис. 2.22), де вказано значення об'єкта (форми або компонентів) вибрати або ввести значення властивості (табл. 2.1).

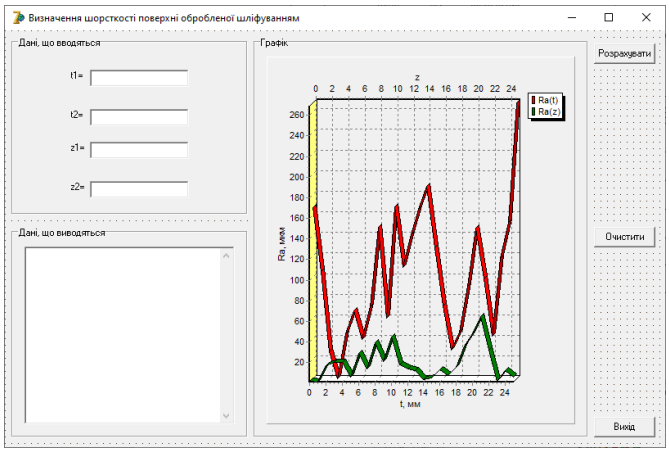


Рис. 2.21. Компоненти та вікно форми «***Form1***» (Форма1) із зміненими властивостями

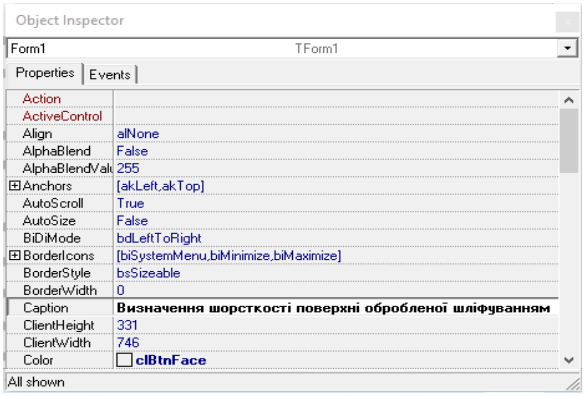


Рис. 2.22. Зміна значення властивості **Caption** (Заголовок) вікна форми «***Form1***» (Форма1) у вікні «***Object Inspector***» (Інспектор об'єктів) на **Визначення шорсткості поверхні обробленої шліфуванням**

2.4.2. Створення обробників подій

Створимо обробники подій для компонентів, вказаних у таблиці 2.2.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.23 вибрати ім'я об'єкта (форми або компонента) (табл. 2.2);

б) у цьому ж вікні потрібно вибрати вкладку «***Events***» (Події) (рис. 2.23);

в) у лівій колонці вкладки «***Events***» (Події) (рис. 2.23), де перелічені імена подій, які може використовувати вибраний об’єкт (компонент), вибрати ім'я події (табл. 2.2);

г) для створення обробника події потрібно у правій колонці вкладки «***Events***» (Події) навпроти обраної раніше події зробити подвійне клацання мишею у полі імені обробника (рис. 2.23);

д) відкриється вікно «***Unit_21.pas***» (Редактор коду), до якого буде додано шаблон процедури обробника **btnExitClick** події **OnClick** (рис. 2.24), а у вікні «***Object Inspector***» (Інспектор об’єктів) поряд з ім'ям події з'явиться ім'я обробника (рис. 2.23);

е) у вікні «***Unit_21.pas***» (Редактор коду) між словами **begin** і **end** необхідно ввести інструкції, що реалізують обробку події (рис. 2.24). У лістингу програми, див. п. п. 2.4.3, наведені тексти інструкцій обробки події **OnClick** для кнопок: **Розрахувати** (btnCalculate), **Очистити** (btnClear) та **Вихід** (btnExit).

Таблиця 2.2 – Обробники подій компонентів для програми

Компонент	Подія	Ім'я обробника
btnCalculate	OnClick	btnCalculateClick
btnClear	OnClick	btnClearClick
btnExit	OnClick	btnExitClick

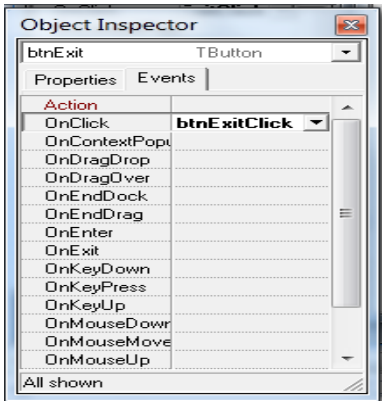


Рис. 2.23. Створення обробника **btnExitClick** події **OnClick** для компонента btnExit, на вкладці «***Events***» (Події) у вікні «***Object Inspector***» (Інспектор об'єктів)

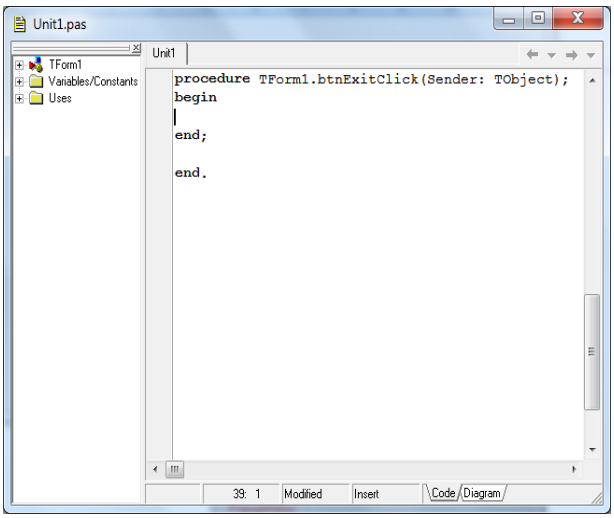


Рис. 2.24. Шаблон процедури обробника **btnExitClick** події **OnClick**, згенерований програмою Delphi

2.4.3. Створення вихідного коду

Вихідний код програми представлений у вигляді лістингу програми. Так як у вихідному коді використовуються математичні функції, то необхідно в розділі **uses** додати модуль **Math**.

Лістинг програми

```
unit Unit_21;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Math;
type
  TfrmRezult = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    GroupBox3: TGroupBox;
    edtinputt1: TEdit;
    edtinputt2: TEdit;
    edtinputz1: TEdit;
    edtinputz2: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    memOutput: TMemo;
    Chart1: TChart;
    btnCalculate: TButton;
    btnClear: TButton;
    btnExit: TButton;
    procedure btnCalculateClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
    procedure btnExitClick(Sender: TObject);
  private
```

```
    { Private declarations }
  public
    { Public declarations }
end;
var
  Rezult: TRezult;
implementation
{$R *.dfm}
procedure TfrmRezult.btnCalculateClick(Sender: TObject);
const
   c : Integer = 2; // Оголошення констант
var
// Оголошення змінних
  Ra, t, t1, t2, z, z1, z2, tmp : Real;
  code : integer;
begin
   // Ввід даних із захистом полів вводу
   // t1
  val(edtinputt1.text,t1,code);
   if code<>0 then
   begin
   Application.MessageBox('Введіть значення t1','Помилка',mb_IconError +
mb_OK);
   edtinputt1.SetFocus;
   exit;
   end;
   // t2
   val(edtinputt2.text,t2,code);
   if code<>0 then
   begin
   Application.MessageBox('Введіть значення t2','Помилка',mb_IconError +
mb_OK);
   edtinputt2.SetFocus;
   exit;
   end;
```

```
  // z1
  val(edtinputz1.text,z1,code);
  if code<>0 then
  begin
  Application.MessageBox('Введіть значення z1','Помилка', mb_IconError +
mb_OK);
  edtinputz1.SetFocus;
  exit;
  end;
  // z2
  val(edtinputz2.text,z2,code);
  if code<>0 then
  begin
  Application.MessageBox('Введіть значення z2','Помилка', mb_IconError +
mb_OK);
  edtinputz2.SetFocus;
  exit;
  end;
  // Очищення поля виводу
  memOutput.Lines.Clear;
  // Очистка графіків
  Chart1.Series[0].Clear;
  Chart1.Series[1].Clear;
  // Обмін місцями початкового t1 та кінцевого t2 значення при невірному вводі
даних
   if t1>t2 then
   begin
     tmp:=t1;
     t1:=t2;
     t2:=tmp;
   end;
  // Обмін місцями початкового z1 та кінцевого z2 значення при невірному
вводі даних
   if z1>z2 then
   begin
```

```
      tmp:=z1;
      z1:=z2;
      z2:=tmp;
    end;
  // Цикл зовнішній
    t:=t1;
    while t<=t2 do
    begin
       // Цикл внутрішній
       z:=z1;
       while z<=z2 do
       begin
         Ra := c*Power(t*1000,z); // Обробка даних
         // Вивід даних із форматуванням
         memOutput.Lines.Add('При t = ' + FloatToStr(t)+ ' мм ' + ' та ' +
         ' z = ' + FloatToStr(z) + ' Ra = ' + FloatToStrF(Ra,ffGeneral,4,2) + ' мкм');
         // Побудова графіків
         Chart1.Series[0].AddXY(t,Ra,'');
         Chart1.Series[1].AddXY(z,Ra,'');
         z:=z+0.05; // Крок внутрішнього циклу
       end;
   t:=t+0.01; // Крок зовнішнього циклу
   end;
end;
procedure TForm1.btnExitClick(Sender: TObject);
begin
   Close; // Закриття програми
end;
procedure TForm1.btnClearClick(Sender: TObject);
begin
  // Очищення полів вводу
  edtinputt1.Text:=' ';
  edtinputt2.Clear;
  edtinputz1.Text:=' ';
  edtinputz2.Clear;
```

```
  // Очищення полів виводу
  memOutput.lines.Clear;
  // Очищення графіків
  Chart1.Series[0].Clear;
  Chart1.Series[1].Clear;
end;
end.
```

2.4.4. Компіляція програми

Процес компіляції складається із двох етапів. На першому етапі виконується перевірка тексту програми на відсутність синтаксичних помилок, а на другому – генерується виконувана програма (exe-файл).

Компіляція програми виконується з «***Головного***» меню **Project** (Проєкт) обравши команду **Compile Project** (Компілювати проєкт) (рис. 2.12) або натиснути клавіші **Ctrl+F9**. Повідомлення про помилки, попередження та підказки відображаються в нижній частині вікна редактора коду (рис. 2.13).

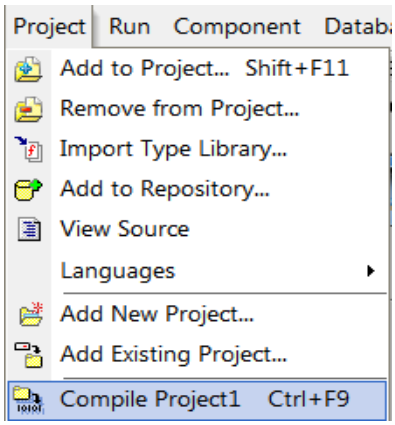


Рис. 2.12. Вибір команди **Compile Project** (Компілювати проєкт) з «***Головного***» меню **Project** (Проєкт)

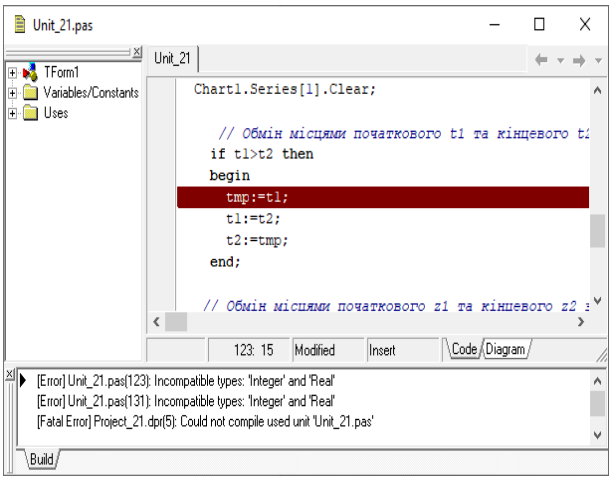


Рис. 2.13. Повідомлення компілятора про виявлені помилки

Процес усунення помилок має ітераційний характер. Зазвичай, спочатку усуваються найбільш очевидні помилки, наприклад, декларуються неоголошені змінні. Після чергового внесення змін до тексту програми виконується повторна компіляція. Слід зважати на те, що компілятор не завжди може точно локалізувати помилку. Тому аналізуючи фрагмент програми який на думку компілятора містить помилку, потрібно звертати увагу не тільки на той

фрагмент коду, на який компілятор встановив курсор, але й на той, який знаходиться у попередньому рядку.

Якщо у програмі немає синтаксичних помилок, компілятор створює виконуваний файл програми. Ім'я виконуваного файлу таке ж, як і у файлу проєкту, а розширення – exe. Delphi поміщає виконуваний файл у той самий каталог, де знаходиться файл проєкту.

2.4.5. Запуск програми

Запуск програми може виконуватися з «***Головного***» меню **Run** (Запуск) обравши команду **Run** (Запуск) або натиснути на кнопці  **Run** (Запуск) на панелі інструментів «***Debug***» (Налагодження) (рис. 2.14) або натиснути клавішу **F9.**

Рис. 2.14. Панель інструментів «***Debug***» (Налагодження)

2.4.6. Помилки часу виконання

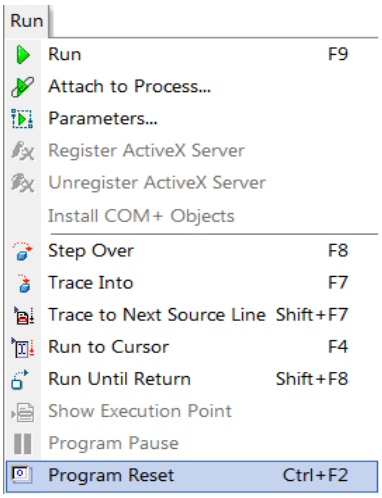


Рис. 2.15. Вибір команди **Program Reset** (Скидання програми) із «***Головного***» меню **Run** (Запуск)

Під час роботи програми можуть виникати помилки, які називаються **помилками часу виконання** (run-time errors) або **винятками** (exceptions).

У більшості випадків причинами винятків є невірні вихідні дані. Після виникнення таких помилок програма перестане робити. Необхідно перервати неробочу програму, вибравши із «***Головного***» меню **Run** (Запуск) команду **Program Reset** (Скидання програми) (рис. 2.15).

2.4.7. Захист полів вводу

Для виключення помилок, при вводі даних у поля вводу необхідно

передбачити у вихідному коді програми захист полів вводу.

У лабораторній роботі №21 використовуємо рядкову функцію Val, яка дозволяє використовувати такі види захисту поля вводу (див. п.п. 2.4.3):

- захист поля вводу від вводу порожнього рядка;
- захист поля вводу від вводу цифри «0» нуль;
- захист поля вводу від вводу літер та деяких символів.

3. Зміст звіту з лабораторної роботи №21

Звіт з лабораторної роботи №21 складається з:

- титульного аркуша (див. Додаток Х);
- аркуша (див. Додаток Х) на якому розташовано: **тема** лабораторної роботи, **мета** лабораторної роботи, **розробка програми** (наведіть **умову** та **формули**, які розглядаються в лабораторній роботі №21, див. п. п. 2.4);
- аркуша (див. Додаток Х) на якому розташовано: **створення інтерфейсу програми** (навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №21, див. п. п. 2.4.1);
- аркушів (див. Додаток Х) на яких розташовані: **лістинг програми** (навести **лістинг** програми, який розглядається в лабораторній роботі №21, див. п. п. 2.4.3) та **висновки**.

Лабораторна робота №22

Тема: Графічні моделі. Властивість Canvas. Властивість Pixels

Мета роботи: Засвоєння початкових знань та придбання навичок, необхідних для розробки програмних продуктів комп`ютерного забезпечення в технологічних процесах обробки металів різанням.

Порядок виконання роботи

1. Вивчити теоретичні відомості.
2. Виконати алгоритм виконання лабораторної роботи №22.
3. Скласти звіт з лабораторної роботи №22 за наведеною формою (див. Додаток Ц).

Хід виконання роботи

1. Теоретичні відомості

Вивчити теоретичні відомості зі списку наведеної літератури.

2. Алгоритм виконання лабораторної роботи №22

2.1. Запуск програми Delphi

У системному меню «***Пуск***», операційної системи Windows, вибрати: **Пуск > Програми > Borland Delphi 7 > Delphi 7**.

2.2. Створення проекту

Після запуску програми Delphi створюється за замовчуванням проект типу – **Application** (Програма).

Delphi дозволяє розробляти різні програми, які в Delphi називаються проектами. У лабораторних роботах створюються проекти – **Application** (Програма), під операційну систему Windows з графічним інтерфейсом.

Для створення іншого проекту, виберіть в «***Головному»*** меню **File** (Файл) команду **New** > **Application** (Нова > Програма) (рис. 2.1).

2.3. Збереження проекту

Delphi для кожного проекту створює декілька файлів. Щоб файли різних проектів не переплуталися між собою, слід для кожного проекту створювати свою **папку** з іменем, яке містить назву та номер лабораторної роботи (наприклад, лабораторна робота 22 – Lab_22).

Рекомендується імені проекту надавати ім’я, яке містить номер лабораторної роботи (наприклад, проект для лабораторної роботи 22 – Project_22). **Файл проекту** Project _22 потрібно зберегти в папці Lab_22.

Необхідно також зберегти **файл модуля вихідного коду**.

Рекомендується імені файлу модуля вихідного коду надавати ім’я, як у проекті (наприклад, Unit _22). Його також потрібно зберегти у папці Lab_22.

На панелі інструментів «***Standard***» (Стандартна) натиснути кнопку 

Save all (Зберегти все) та зберегти **файли проекту** та **файл модуля вихідного коду** згідно з вище сказаними рекомендаціями (рис. 2.2).

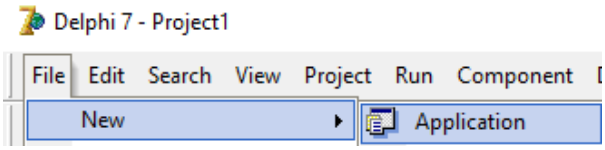


Рис. 2.1. Виберіть **New** > **Application** (Нова > Програма) в «***Головному***» меню **File** (Файл)

Рис. 2.2. Натиснути кнопку  **Save all** (Зберегти все) на панелі інструментів «***Standard***» (Стандартна)

2.4. Розробка програми

Розробити програму для обчислення сили струму в електричному ланцюгу верстата, якщо напруга U змінюється від 220 до 225 В з кроком 1 В, а опір R = 800 Ом. Відповідно до закону Ома, використовуємо формулу

$$I = U / R.$$

Для наочного представлення залежності функції від аргументу додати до програми графік використовуючи властивість Canvas і властивість Pixels.

2.4.1. Створення інтерфейсу програми

Інтерфейс програми представлений на рис. 2.3. Вивід повідомлень програми представлено на рис. 2.4 (а, б).

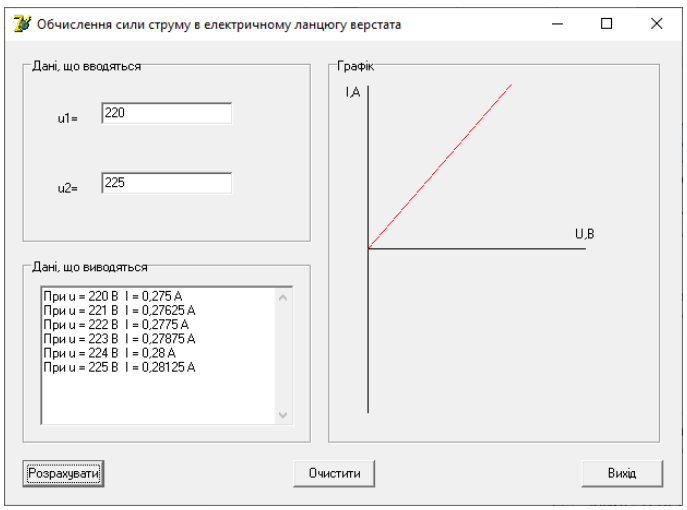
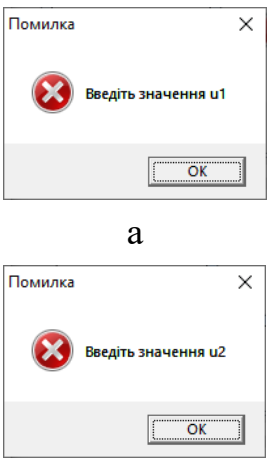


Рис. 2.3. Інтерфейс програми



а

б

Рис. 2.4. Вивід повідомлень програми: а, б

2.4.1.1. Вікно форми Form1 (Форма1)

При створенні проекту типу – **Application** (Програма), Delphi автоматично створює вікно форми «***Form1***» (Форма1) (рис. 2.5). Вікно форми «***Form1***» (Форма1) являє собою шаблон вікна розробляємої програми.

2.4.1.2. Розміщення компонентів на вікні форми Form1 (Форма1)

На вікні форми «***Form1***» (Форма1) розмістити компоненти (рис. 2.6) вказані в таблиці 2.1.

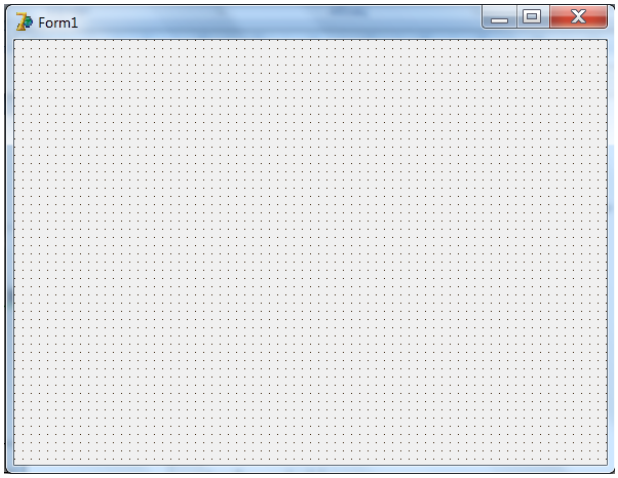


Рис. 2.5. Вікно форми «***Form1***» (Форма1)

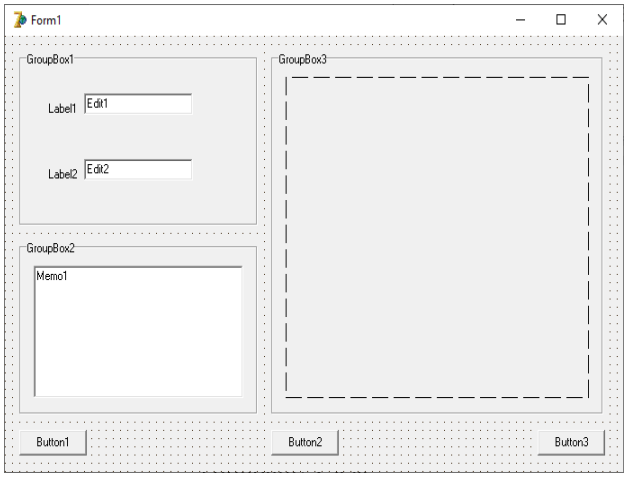


Рис. 2.6. Компоненти розміщені на вікні форми «***Form1***» (Форма1)

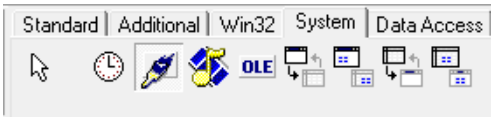
Для цього необхідно виконати наступні дії:

а) на «***Панелі компонентів***» на вкладці «***Standard***» (Стандартна) (рис. 2.7, а) та на вкладці «***System***» (Система) (рис. 2.7, б), вказати курсором миші на піктограму потрібного компонента (табл. 2.1);

б) потім вказати курсором миші у довільному місці на формі «***Form1***» (Форма1) (рис. 2.6);

в) якщо необхідно, перетягніть компонент у потрібне місце на формі «***Form1***» (Форма1) (рис. 2.6).

а



б

Рис. 2.7. «***Панель компонентів***»: а) вкладка «***Standard***» (Стандартна), б) вкладка «***System***» (Система)

Таблиця 2.1 – Компоненти та їх властивості для інтерфейсу програми

Компонент	**Піктограма**	**Властивість**	**Значення**
Form1	-	Caption	Обчислення сили струму в електричному ланцюгу верстата
		BorderStyle	bsSingle
Groupbox1		Caption	Дані, що вводяться
Groupbox2		Caption	Дані, що виводяться
Groupbox3		Caption	Графік
Edit1		Name	edtInputu1
		Text	значення не вводимо, а тільки очистити поле
Edit2		Name	edtInputu2
		Text	значення не вводимо, а тільки очистити поле
Label1		Caption	u1=
Label2		Caption	u2=
Memo1		Name	memOutput
		Lines	натиснути кнопку ···. У з'явившемуся вікні очистити поле, потім натиснути кнопку **ОК**
		ReadOnly	True
PaintBox1		Name	pntGraphic
Button1		Name	btnCalculate
		Caption	Розрахувати
Button2		Name	btnClear
		Caption	Очистити
Button3		Name	btnExit
		Caption	Вихід

2.4.1.3. Зміна властивостей вікна форми Form1 (Форма1) та компонентів

Змінимо властивості вікна форми «***Form1***» (Форма1) та компонентів (рис. 2.8) згідно з вказаними властивостями в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.9) вибрати ім'я об'єкта (форми або компонентів) (табл. 2.1);

б) у лівій колонці вкладки «***Properties***» (Властивості) (рис. 2.9), де перелічені властивості об'єкта (форми чи компонентів) виділити властивість (табл. 2.1);

в) у правій колонці вкладки «***Properties***» (Властивості) (рис. 2.9), де вказано значення об'єкта (форми або компонентів) вибрати або ввести значення властивості (табл. 2.1).

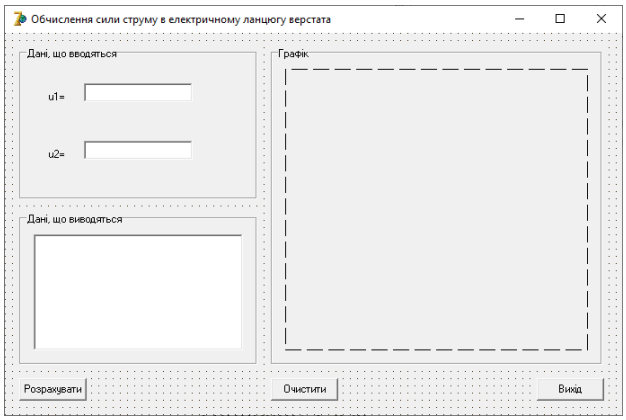


Рис. 2.8. Компоненти та вікно форми «***Form1***» (Форма1) із зміненими властивостями

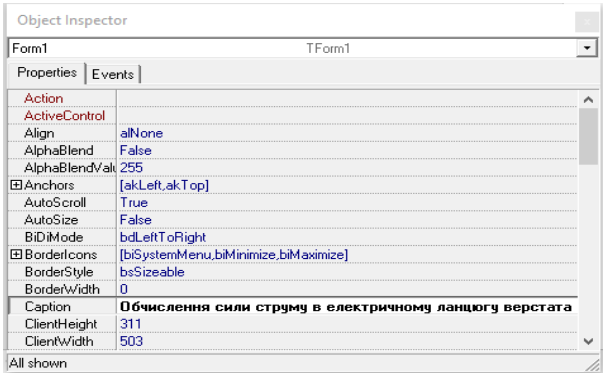


Рис. 2.9. Зміна значення властивості **Caption** (Заголовок) вікна форми «***Form1***» (Форма1) у вікні «***Object Inspector***» (Інспектор об'єктів) на **Обчислення сили струму в електричному ланцюгу верстата**

2.4.2. Створення обробників подій

Створимо обробників подій для компонентів, вказаних у таблиці 2.2.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.10) вибрати ім'я об'єкта (форми або компонента) (табл. 2.2);

б) у цьому ж вікні потрібно вибрати вкладку «***Events***» (Події) (рис. 2.10);

в) у лівій колонці вкладки «***Events***» (Події) (рис. 2.10), де перелічені імена подій, які може використовувати вибраний об’єкт (компонент), вибрати ім'я події (табл. 2.2);

г) для створення функції обробки події потрібно у правій колонці вкладки «***Events***» (Події) навпроти обраної раніше події зробити подвійне клацання мишею у полі імені обробника (рис. 2.10);

д) відкриється вікно «***Unit_22.pas***» (Редактор коду), до якого буде додано шаблон процедури обробки **btnExitClick** події **OnClick** (рис. 2.11), а у вікні **Object Inspector** (Інспектор об’єктів) поряд з ім'ям події з'явиться ім'я функції його обробника (рис. 2.10);

е) у вікні «***Unit_22.pas***» (Редактор коду) між словами **begin** і **end** необхідно ввести інструкції, що реалізують обробку події (рис. 2.11). У лістингу програми, див. п.п. 2.4.3, наведені тексти інструкцій обробки події **OnClick** для кнопок: **Розрахувати** (btnCalculate), **Очистити** (btnClear) та **Вихід** (btnExit).

Таблиця 2.2 – Обробники подій компонентів для програми

Компонент	**Подія**	**Ім'я обробника**
btnCalculate	OnClick	btnCalculateClick
btnClear	OnClick	btnClearClick
btnExit	OnClick	btnExitClick

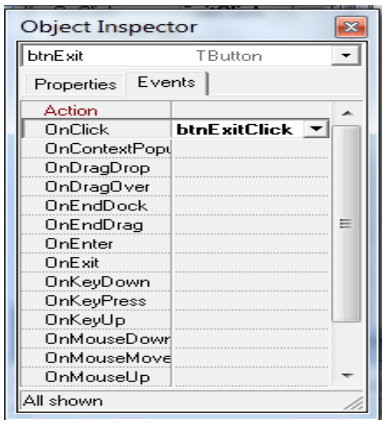


Рис. 2.10. Створення обробника **btnExitClick** події **OnClick** для компонента btnExit, на вкладці «***Events***» (Події) у вікні «***Object Inspector***» (Інспектор об'єктів)

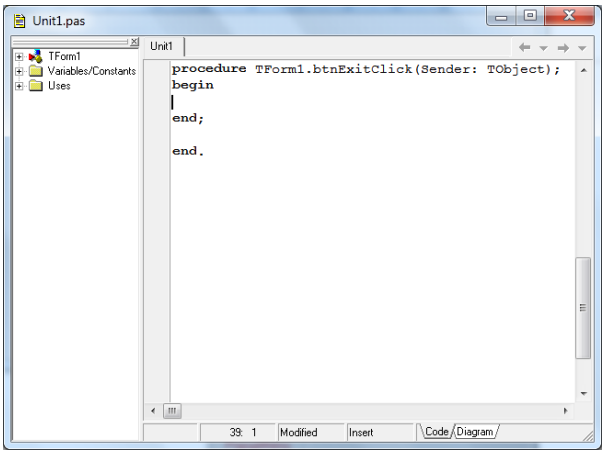


Рис. 2.11. Шаблон процедури обробки **btnExitClick** події **OnClick**, згенерований програмою Delphi

2.4.3. Створення вихідного коду

Вихідний код програми представлений у вигляді лістингу програми.

Лістинг програми

```
Unit Unit_22;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls;
type
  TForm1 = class(TForm)
    btnCalculate: TButton;
    btnClear: TButton;
    btnExit: TButton;
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    GroupBox3: TGroupBox;
    memOutput: TMemo;
    edtinputu1: TEdit;
    edtinputu2: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    pntGraphic: TPaintBox;
    procedure btnExitClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
    procedure btnCalculateClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
```

```
implementation
{$R *.dfm}
procedure TForm1.btnExitClick(Sender: TObject);
begin
   Application.Terminate;  // Закриття програми
end;
procedure TForm1.btnClearClick(Sender: TObject);
begin
   // Очищення полів вводу
  edtinputu1.Text:='';
  edtinputu2.Clear;
  // Очищення графіка
  pntGraphic.Repaint;
  // Очищення полів виводу
  memOutput.lines.Clear;
end;
procedure TForm1.btnCalculateClick(Sender: TObject);
const
  R:integer=800;  // Оголошення констант
var
  // Оголошення змінних
  x01, y01, code : integer;
  U, u1, u2, tmp, I, i1, i2, kmx, kmy, du : real;
begin
   // Ввід даних із захистом полів вводу
   // u1
   val(edtinputu1.text,u1,code);
   if code<>0 then
   begin
   Application.MessageBox('Введіть значення u1','Помилка', mb_IconError +
mb_OK);
   edtinputu1.SetFocus;
   exit;
   end;
   // u2
```

```
  val(edtinputu2.text,u2,code);
  if code<>0 then
  begin
  Application.MessageBox('Введіть значення u2','Помилка',mb_IconError +
mb_OK);
  edtinputu2.SetFocus;
  exit;
  end;
  // Обмін місцями початкового та кінцевого значення при невірному вводу
даних
  if u1>u2 then
  begin
  tmp:=u1;
  u1:=u2;
  u2:=tmp;
  end;
  // Очищення поля виводу
  memOutput.Lines.Clear;
  // Цикл
  u:=u1;
  while u<=u2 do
  begin
  I:=U/R;  // Обробка даних
  // Вивід даних із форматуванням
  memOutput.Lines.Add('При u = ' + FloatToStr(u) +' B' + '  I = ' +
  FloatToStrF(I,ffGeneral,4,2)+' A');
  u:=u+1;   // Крок циклу
  end;
  pntGraphic.Repaint;  // Очищення графіка
  // Початок координат
  x01:=20;
  y01:=150;
 // Найменування осі - y
       pntGraphic.Canvas.TextOut(0,0,'I,A');
 // Найменування осі - x
```

```
        pntGraphic.Canvas.TextOut(210,130,'U,B');
  // Побудова осі X
        pntGraphic.Canvas.MoveTo(x01,y01);
        pntGraphic.Canvas.LineTo(x01+200,y01);
  // Побудова осі Y
        pntGraphic.Canvas.MoveTo(x01,y01+150);
        pntGraphic.Canvas.LineTo(x01,y01-150);
   // Коефіцієнт масштабування по осі X
   kmx:=pntGraphic.Width/(u2-u1);
   // Крок побудови графіка
   du:=1/kmx;
   // Визначення мінімального та максимального значення i
   i1:=u1/r;
   i2:= u2/r;
   // Цикл визначення мінімального та максимального значення i
   u:=u1;
   repeat
   i:=u/r;
   if i<i1 then i1:=i;
   if i>i2 then i2:=i;
   u:=u+du;  // Крок циклу
   until u>=u2;
   // Коефіцієнт масштабування по осі Y
   kmy:=pntGraphic.Height/(i2-i1);
    // Цикл побудови графіка
   u:=u1;
   repeat
   i:=u/r;
   pntGraphic.Canvas.Pixels[x01+round((u-u1)*kmx),y01-round((i-i1)*kmy)]:=clred;
   // Крок побудови графіка
   u:=u+du;
   until u>=u2;
end;
end.
```

2.4.4. Компіляція програми

Процес компіляції складається із двох етапів. На першому етапі виконується перевірка тексту програми на відсутність синтаксичних помилок, а на другому – генерується виконувана програма (exe-файл).

Компіляція програми виконується з «***Головного***» меню **Project** (Проект) обравши команду **Compile Project** (Компілювати проект) (рис. 2.12) або натиснути клавіші **Ctrl+F9**. Повідомлення про помилки, попередження та підказки відображаються в нижній частині вікна редактора коду (рис. 2.13).

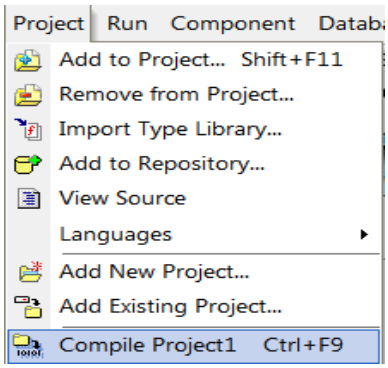


Рис. 2.12. Вибір команди **Compile Project** (Компілювати проект) з «***Головного***» меню **Project** (Проект)

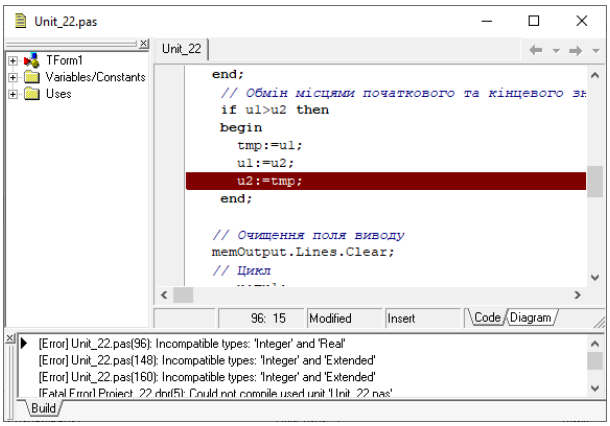


Рис. 2.13. Повідомлення компілятора про виявлені помилки

Процес усунення помилок має ітераційний характер. Зазвичай, спочатку усуваються найбільш очевидні помилки, наприклад, декларуються неоголошені змінні. Після чергового внесення змін до тексту програми виконується повторна компіляція. Слід зважати на те, що компілятор не завжди може точно локалізувати помилку. Тому аналізуючи фрагмент програми який на думку компілятора містить помилку, потрібно звертати увагу не тільки на той фрагмент коду, на який компілятор встановив курсор, але й на той, який знаходиться у попередньому рядку.

Якщо у програмі немає синтаксичних помилок, компілятор створює виконуваний файл програми. Ім'я виконуваного файлу таке ж, як і у файлу проекту, а розширення – exe. Delphi поміщає виконуваний файл у той самий каталог, де знаходиться файл проекту.

2.4.5. Запуск програми

Запуск програми може виконуватися з «***Головного***» меню **Run** (Запуск) обравши команду **Run** (Запуск) або натиснути на кнопці  **Run** (Запуск) на панелі інструментів «***Debug***» (Налагодження) (рис. 2.14) або натиснути клавішу **F9.**

Рис. 2.14. Панель інструментів «***Debug***» (Налагодження)

2.4.6. Помилки часу виконання

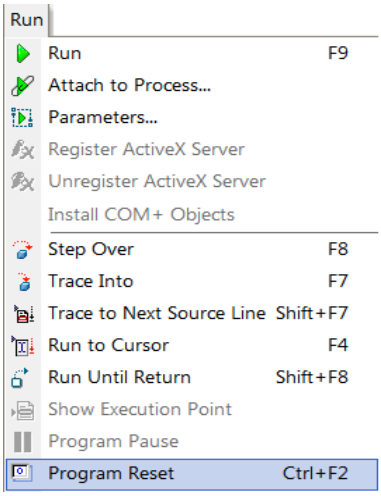


Рис. 2.15. Вибір команди **Program Reset** (Скидання програми) із «***Головного***» меню **Run** (Запуск)

Під час роботи програми можуть виникати помилки, які називаються **помилками часу виконання** (run-time errors) або **винятками** (exceptions).

У більшості випадків причинами винятків є невірні вихідні дані. Після виникнення таких помилок програма перестане робити. Необхідно перервати неробочу програму, вибравши із «***Головного***» меню **Run** (Запуск) команду **Program Reset** (Скидання програми) (рис. 2.15).

2.4.7. Захист полів вводу

Для виключення помилок, при вводі даних у поля вводу необхідно передбачити у вихідному коді програми захист полів вводу.

У лабораторній роботі №22 використовуємо рядкову функцію Val, яка дозволяє використовувати такі види захисту поля вводу (див. п.п. 2.4.3):

- захист поля вводу від вводу порожнього рядка;
- захист поля вводу від вводу цифри «0» нуль;
- захист поля вводу від вводу літер та деяких символів.

3. Зміст звіту з лабораторної роботи №22

Звіт з лабораторної роботи №22 складається з:

- титульного аркуша (див. далі);

- аркуша (див. далі) на якому розташовано: **тема** лабораторної роботи, **мета** лабораторної роботи, **розробка програми** (наведіть **умову** та **формули**, які розглядаються в лабораторній роботі №22, див. п.п. 2.4);

- аркуша (див. далі) на якому розташовано: **створення інтерфейсу програми** (навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №22, див. п. п. 2.4.1);

- аркушів (див. далі) на яких розташовані: **лістинг програми** (навести **лістинг** програми, який розглядається в лабораторній роботі №22, див. п. п. 2.4.3) та **висновки**.

Лабораторна робота №23

Тема: Графічні моделі. Властивість Canvas. Метод LineTo

Мета роботи: Засвоєння початкових знань та придбання навичок, необхідних для розробки програмних продуктів комп`ютерного забезпечення в технологічних процесах обробки металів різанням.

Порядок виконання роботи

1. Вивчити теоретичні відомості.

2. Виконати алгоритм виконання лабораторної роботи №23.

3. Скласти звіт з лабораторної роботи №23 за наведеною формою (див. Додаток Ч).

Хід виконання роботи

1. Теоретичні відомості

Вивчити теоретичні відомості зі списку наведеної літератури.

2. Алгоритм виконання лабораторної роботи №23

2.1. Запуск програми Delphi

У системному меню «***Пуск***», операційної системи Windows, вибрати: **Пуск > Програми > Borland Delphi 7 > Delphi 7**.

2.2. Створення проекту

Після запуску програми Delphi створюється за замовчуванням проект типу – **Application** (Програма).

Delphi дозволяє розробляти різні програми, які в Delphi називаються проектами. У лабораторних роботах створюються проекти – **Application** (Програма), під операційну систему Windows з графічним інтерфейсом.

Для створення іншого проекту, виберіть в «***Головному»*** меню **File** (Файл) команду **New** > **Application** (Нова > Програма) (рис. 2.1).

2.3. Збереження проекту

Delphi для кожного проекту створює декілька файлів. Щоб файли різних проектів не переплуталися між собою, слід для кожного проекту створювати свою **папку** з іменем, яке містить назву та номер лабораторної роботи (наприклад, лабораторна робота 23 – Lab_23).

Рекомендується імені проекту надавати ім’я, яке містить номер лабораторної роботи (наприклад, проект для лабораторної роботи 23 – Project_23). **Файл проекту** Project _23 потрібно зберегти в папці Lab_23.

Необхідно також зберегти **файл модуля вихідного коду**.

Рекомендується імені файлу модуля вихідного коду надавати ім’я, як у проекті (наприклад, Unit _23). Його також потрібно зберегти у папці Lab_23.

На панелі інструментів «***Standard***» (Стандартна) натиснути кнопку  **Save all** (Зберегти все) та зберегти **файли проекту** та **файл модуля вихідного коду** згідно з вище сказаними рекомендаціями (рис. 2.2).

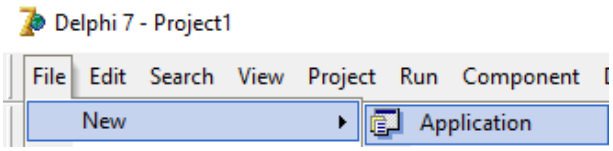


Рис. 2.1. Виберіть **New** > **Application** (Нова > Програма) в «***Головному***» меню **File** (Файл)

Рис. 2.2. Натиснути кнопку **Save all** (Зберегти все) на панелі інструментів «***Standard***» (Стандартна)

2.4. Розробка програми

Розробити програму для обчислення значення шорсткості Ra поверхні, обробленою шліфуванням, за формулою

$$Ra = c \cdot t^z,$$

при зміні глибини різання t від 0,01 до 0,04 мм з кроком 0,01 мм, де показник ступеня z = 0,4, а значення коефіцієнта c = 2. Для наочного представлення залежності функції від аргументу додати до програми графік використовуючи властивість Canvas і метод LineTo.

2.4.1. Створення інтерфейсу програми

Інтерфейс програми представлений на рис. 2.3. Вивід повідомлень програми представлено на рис. 2.4 (а, б).

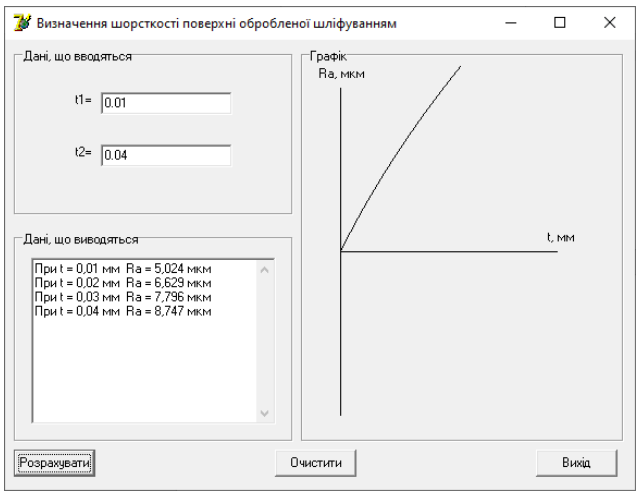
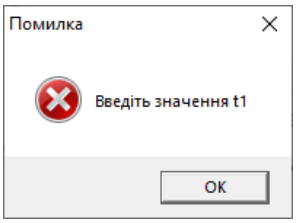
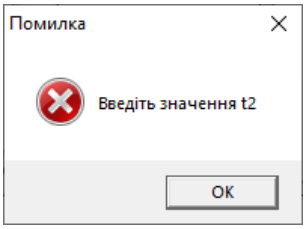


Рис. 2.3. Інтерфейс програми



а



б

Рис. 2.4. Вивід повідомлень програми: а, б

2.4.1.1. Вікно форми Form1 (Форма1)

При створенні проекту типу – **Application** (Програма), Delphi автоматично створює вікно форми «***Form1***» (Форма1) (рис. 2.5). Вікно форми «***Form1***» (Форма1) являє собою шаблон вікна розробляємої програми.

2.4.1.2. Розміщення компонентів на вікні форми Form1 (Форма1)

На вікні форми «***Form1***» (Форма1) розмістити компоненти (рис. 2.6) вказані в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) на «***Панелі компонентів***» на вкладці «***Standard***» (Стандартна) (рис. 2.7, а) та на вкладці «***System***» (Система) (рис. 2.7, б), вказати курсором миші на піктограму потрібного компонента (табл. 2.1);

б) потім вказати курсором миші у довільному місці на формі «***Form1***» (Форма1) (рис. 2.6);

в) якщо необхідно, перетягніть компонент у потрібне місце на формі «***Form1***» (Форма1) (рис. 2.6).

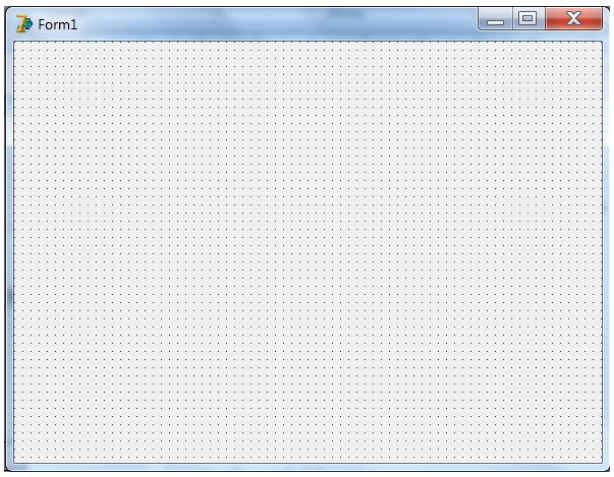


Рис. 2.5. Вікно форми «***Form1***» (Форма1)

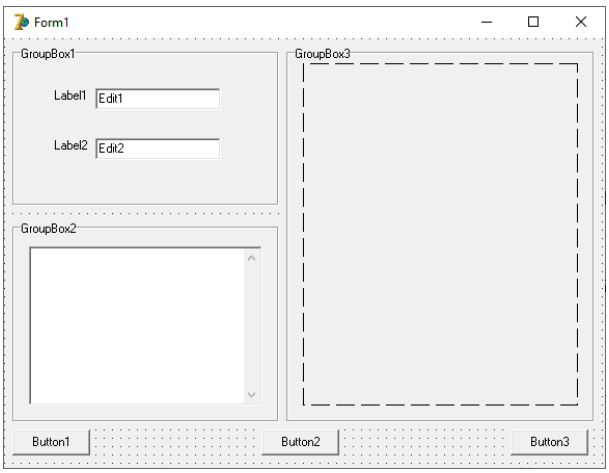
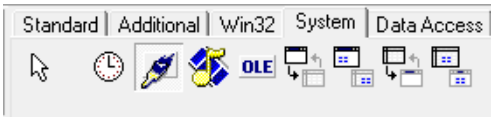


Рис. 2.6. Компоненти розміщені на вікні форми «***Form1***» (Форма1)

а



б

Рис. 2.7. «***Панель компонентів***»: а) вкладка «***Standard***» (Стандартна), б) вкладка «***System***» (Система)

Таблиця 2.1 – Компоненти та їх властивості для інтерфейсу програми

Компонент	**Піктограма**	**Властивість**	**Значення**
Form1	-	Caption	Визначення шорсткості поверхні обробленої шліфуванням
		BorderStyle	bsSingle
Groupbox1		Caption	Дані, що вводяться
Groupbox2		Caption	Дані, що виводяться
Groupbox3		Caption	Графік
Edit1		Name	edtInputt1
		Text	значення не вводимо, а тільки очистити поле
Edit2		Name	edtInputt2
		Text	значення не вводимо, а тільки очистити поле
Label1		Caption	t1=
Label2		Caption	t2=
Memo1		Name	memOutput
		ScrollBars	ssVertical
		Lines	натиснути кнопку ... У з'явившемуся вікні очистити поле, потім натиснути кнопку кнопку **ОК**
		ReadOnly	True
PaintBox1		Name	pntGraphic
Button1		Name	btnCalculate
		Caption	Розрахувати
Button2		Name	btnClear
		Caption	Очистити
Button3		Name	btnExit
		Caption	Вихід

2.4.1.3. Зміна властивостей вікна форми Form1 (Форма1) та компонентів

Змінимо властивості вікна форми «***Form1***» (Форма1) та компонентів (рис. 2.8) згідно з вказаними властивостями в таблиці 2.1.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів) (рис. 2.9) вибрати ім'я об'єкта (форми або компонентів) (табл. 2.1);

б) у лівій колонці вкладки «***Properties***» (Властивості) (рис. 2.9), де перелічені властивості об'єкта (форми чи компонентів) виділити властивість (табл. 2.1);

в) у правій колонці вкладки «***Properties***» (Властивості) (рис. 2.9), де вказано значення об'єкта (форми або компонентів) вибрати або ввести значення властивості (табл. 2.1).

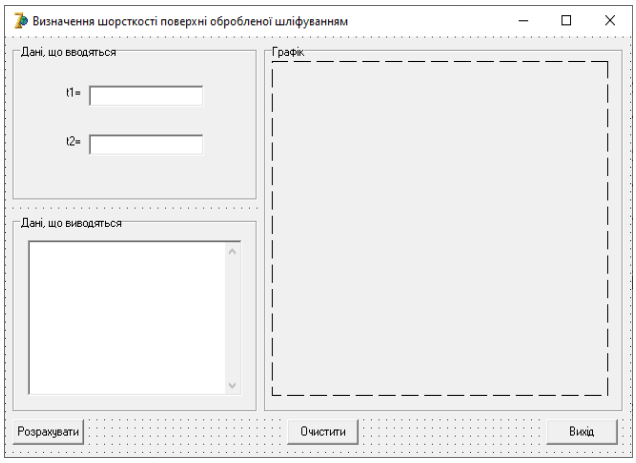


Рис. 2.8. Компоненти та вікно форми «***Form1***» (Форма1) із зміненими властивостями

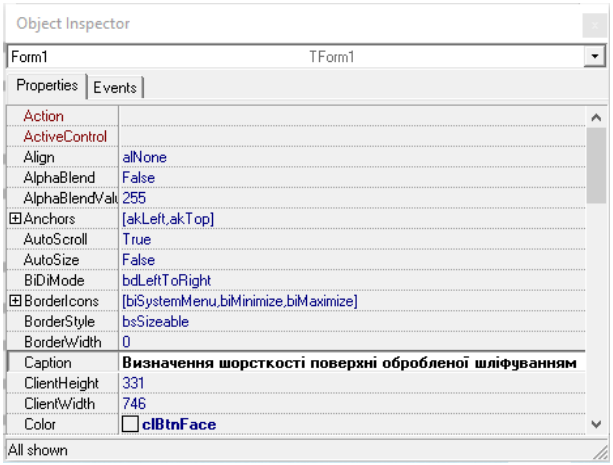


Рис. 2.9. Зміна значення властивості **Caption** (Заголовок) вікна форми «***Form1***» (Форма1) у вікні «***Object Inspector***» (Інспектор об'єктів) на **Визначення шорсткості поверхні обробленої шліфуванням**

2.4.2. Створення обробників подій

Створимо обробників подій для компонентів, вказаних у таблиці 2.2.

Для цього необхідно виконати наступні дії:

а) у верхній частині вікна «***Object Inspector***» (Інспектор об'єктів)

(рис. 2.10) вибрати ім'я об'єкта (форми або компонента) (табл. 2.2);

б) у цьому ж вікні потрібно вибрати вкладку «***Events***» (Події) (рис. 2.10);

в) у лівій колонці вкладки «***Events***» (Події) (рис. 2.10), де перелічені імена подій, які може використовувати вибраний об’єкт (компонент), вибрати ім'я події (табл. 2.2);

г) для створення функції обробки події потрібно у правій колонці вкладки «***Events***» (Події) навпроти обраної раніше події зробити подвійне клацання мишею у полі імені обробника (рис. 2.10);

д) відкриється вікно «***Unit_23.pas***» (Редактор коду), до якого буде додано шаблон процедури обробки **btnExitClick** події **OnClick** (рис. 2.11), а у вікні «***Object Inspector***» (Інспектор об’єктів) поряд з ім'ям події з'явиться ім'я функції його обробника (рис. 2.10);

Таблиця 2.2 – Обробники подій компонентів для програми

Компонент	**Подія**	**Ім'я обробника**
btnCalculate	OnClick	btnCalculateClick
btnClear	OnClick	btnClearClick
btnExit	OnClick	btnExitClick

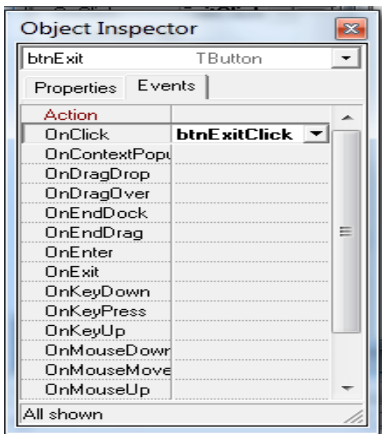


Рис. 2.10. Створення обробника **btnExitClick** події **OnClick** для компонента btnExit, на вкладці «***Events***» (Події) у вікні «***Object Inspector***» (Інспектор об'єктів)

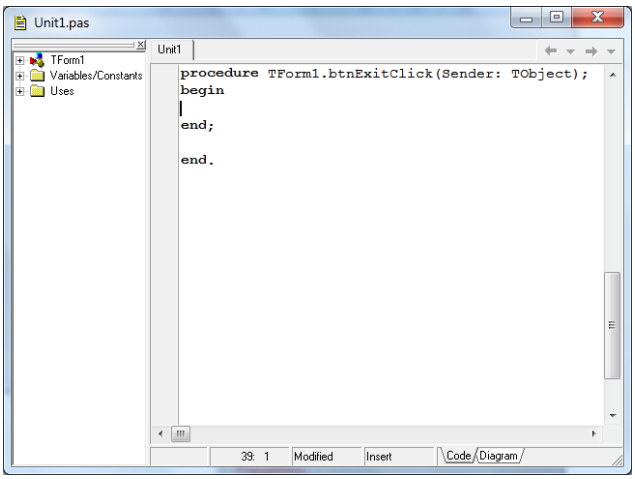


Рис. 2.11. Шаблон процедури обробки **btnExitClick** події **OnClick**, згенерований програмою Delphi

е) у вікні «***Unit_23.pas***» (Редактор коду) між словами **begin** і **end** необхідно ввести інструкції, що реалізують обробку події (рис. 2.11). У

лістингу програми, див. п.п. 2.4.3, наведені тексти інструкцій обробки події **OnClick** для кнопок: **Розрахувати** (btnCalculate), **Очистити** (btnClear) та **Вихід** (btnExit).

2.4.3. Створення вихідного коду

Вихідний код програми представлений у вигляді лістингу програми. Так як у вихідному коді використовуються математичні функції, то необхідно в розділі **uses** додати модуль **Math**.

Лістинг програми

```
Unit Unit_23;
interface
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, Math;
type
  TForm1 = class(TForm)
    btnCalculate: TButton;
    btnClear: TButton;
    btnExit: TButton;
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    GroupBox3: TGroupBox;
    memOutput: TMemo;
    edtinputt1: TEdit;
    edtinputt2: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    pntGraphic: TPaintBox;
    procedure btnExitClick(Sender: TObject);
    procedure btnClearClick(Sender: TObject);
    procedure btnCalculateClick(Sender: TObject);
  private
    { Private declarations }
```

```
  public
    { Public declarations }
  end;
var
  Form1: TForm1;
implementation
{$R *.dfm}
procedure TForm1.btnExitClick(Sender: TObject);
begin
   Application.Terminate;  // Закриття програми
end;
procedure TForm1.btnClearClick(Sender: TObject);
begin
   // Очищення полів вводу
  edtinputt1.Text:='';
  edtinputt2.Clear;
  // Очищення графіка
  pntGraphic.Repaint;
  // Очищення полів виводу
  memOutput.lines.Clear;
end;
procedure TForm1. btnCalculateClick(Sender: TObject);
const
  // Оголошення констант
  c : Integer=2;
   z : real = 0.4;
var
  // Оголошення змінних
   t, t1, t2, tmp, Ra, Ra1, Ra2, kmx, kmy, dt : real;
  x01, y01, code : integer;
begin
   // Ввід даних із захистом полів вводу
   // t1
   val(edtInputt1.text,t1,code);
   if code<>0 then
```

```
  begin
  Application.MessageBox('Введіть значення t1','Помилка', mb_IconError +
mb_OK);
  edtInputt1.SetFocus;
  exit;
  end;
  // t2
  val(edtinputt2.text,t2,code);
  if code<>0 then
  begin
  Application.MessageBox('Введіть значення t2','Помилка',mb_IconError +
mb_OK);
  edtInputt2.SetFocus;
  exit;
  end;
  // Обмін місцями початкового t1 та кінцевого t2 значення при невірному вводі
даних
  if t1>t2 then
  begin
  tmp:=t1;
  t1:=t2;
  t2:=tmp;
  end;
  memOutput.Lines.Clear;  // Очищення поля виводу
 // Цикл
  t:=t1;
  while t<=t2 do
  begin
  Ra := c*power(t*1000,z);  // Обробка даних
  // Вивід даних із форматуванням
  memOutput.Lines.Add('При t = ' + FloatToStr(t) + ' мм' +  '  Ra = ' +
  FloatToStrF(Ra,ffGeneral,4,2) + ' мкм');
  t:=t+0.01;  // Крок циклу
  end;
  pntGraphic.Repaint;  // Очищення графіка
```

```
 // Початок координат
 x01:=20;
 y01:=170;
// Найменування осі - y
 pntGraphic.Canvas.TextOut(0,0,'Ra, мкм');
// Найменування осі - x
 pntGraphic.Canvas.TextOut(210,150,'t, мм');
// Побудова осі X
 pntGraphic.Canvas.MoveTo(x01,y01);
 pntGraphic.Canvas.LineTo(x01+200,y01);
// Побудова осі
 pntGraphic.Canvas.MoveTo(x01,y01+150);
 pntGraphic.Canvas.LineTo(x01,y01-150);
 // Коефіцієнт масштабування по осі X
 kmx:=pntGraphic.Width/(t2-t1);
 // Крок побудови графіка
 dt:=1/kmx;
 // Визначення мінімального та максимального значення Ra
 Ra1:=c*power(t1*1000,z);
 Ra2:= c*power(t2*1000,z);
 // Цикл визначення мінімального та максимального значення Ra
 t:=t1;
 repeat
 Ra := c*power(t*1000,z);
 if Ra<Ra1 then Ra1:=Ra;
 if Ra>Ra2 then Ra2:=Ra;
 t:=t+dt;   // Крок циклу
 until t>=t2;
 // Коефіцієнт масштабування по осі Y
 kmy:=pntGraphic.Height/(Ra2-Ra1);
 // Цикл побудови графіка
 t:=t1;
 repeat
 Ra := c*power(t*1000,z);
 pntGraphic.Canvas.LineTo(x01+round((t-t1)*kmx),y01-round((Ra-Ra1)*kmy));
```

```
  t:=t+dt;  // Крок побудови графіка
  until t>=t2;
end;
end.
```

2.4.4. Компіляція програми

Процес компіляції складається із двох етапів. На першому етапі виконується перевірка тексту програми на відсутність синтаксичних помилок, а на другому – генерується виконувана програма (exe-файл).

Компіляція програми виконується з «***Головного***» меню **Project** (Проект) обравши команду **Compile Project** (Компіляція проекту) (рис. 2.12) або натиснути клавіші **Ctrl+F9**. Повідомлення про помилки, попередження та підказки відображаються в нижній частині вікна редактора коду (рис. 2.13).

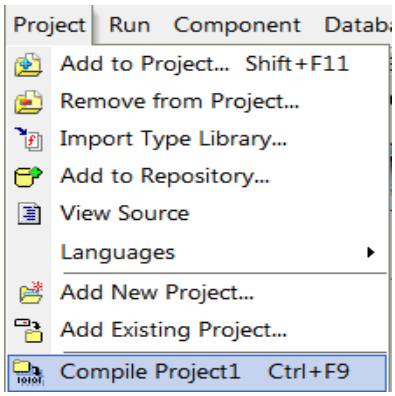


Рис. 2.12. Вибір команди **Compile Project** (Компіляція проекту) з «***Головного***» меню **Project** (Проект)

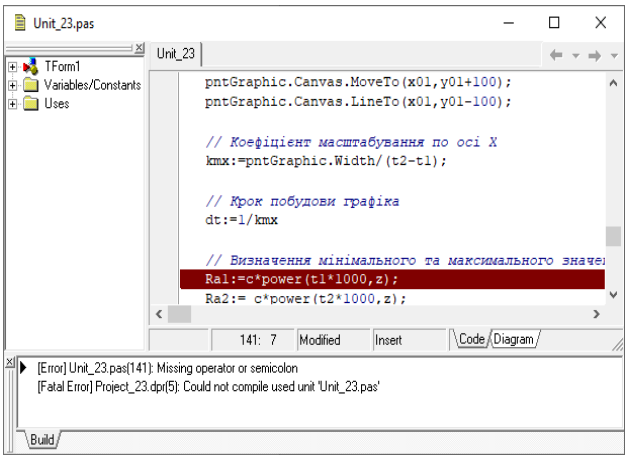


Рис. 2.13. Повідомлення компілятора про виявлені помилки

Процес усунення помилок має ітераційний характер. Зазвичай, спочатку усуваються найбільш очевидні помилки, наприклад, декларуються неоголошені змінні. Після чергового внесення змін до тексту програми виконується повторна компіляція. Слід зважати на те, що компілятор не завжди може точно локалізувати помилку. Тому аналізуючи фрагмент програми який на думку компілятора містить помилку, потрібно звертати увагу не тільки на той фрагмент коду, на який компілятор встановив курсор, але й на той, який знаходиться у попередньому рядку.

Якщо у програмі немає синтаксичних помилок, компілятор створює

виконуваний файл програми. Ім'я виконуваного файлу таке ж, як і у файлу проекту, а розширення – exe. Delphi поміщає виконуваний файл у той самий каталог, де знаходиться файл проекту.

2.4.5. Запуск програми

Запуск програми може виконуватися з «***Головного***» меню **Run** (Запуск) обравши команду **Run** (Запуск) або натиснути на кнопці  **Run** (Запуск) панелі інструментів «***Debug***» (Налагодження) (рис. 2.14) або натиснути клавішу **F9.**

Рис. 2.14. Панель інструментів «***Debug***» (Налагодження)

2.4.6. Помилки часу виконання

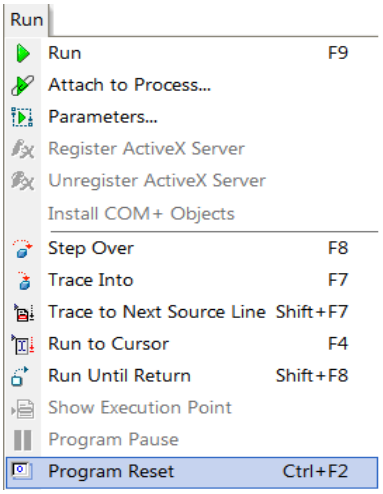


Рис. 2.15. Вибір команди **Program Reset** (Скидання програми) із «***Головного***» меню **Run** (Запуск)

Під час роботи програми можуть виникати помилки, які називаються **помилками часу виконання** (run-time errors) або **винятками** (exceptions).

У більшості випадків причинами винятків є невірні вихідні дані.

Після виникнення таких помилок програма перестане робити.

Необхідно перервати неробочу програму, вибравши із «***Головного***» меню **Run** (Запуск) команду **Program Reset** (Скидання програми) (рис. 2.15).

2.4.7. Захист полів вводу

Для виключення помилок, при вводі даних у поля вводу необхідно передбачити у вихідному коді програми захист полів вводу.

У лабораторній роботі №23 використовуємо рядкову функцію Val, яка дозволяє використовувати такі види захисту поля вводу (див. п.п. 2.4.3):

- захист поля вводу від вводу порожнього рядка;

- захист поля вводу від вводу цифри «0» нуль;

- захист поля вводу від вводу літер та деяких символів.

3. Зміст звіту з лабораторної роботи №23

Звіт з лабораторної роботи №23 складається з:

- титульного аркуша (див. Додаток Ч);

- аркуша (див. Додаток Ч) на якому розташовано: **тема** лабораторної роботи, **мета** лабораторної роботи, **розробка програми** (наведіть **умову** та **формули**, які розглядаються в лабораторній роботі №23, див. п.п. 2.4);

- аркуша (див. Додаток Ч) на якому розташовано: **створення інтерфейсу програми** (навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №23, див. п.п. 2.4.1);

- аркушів (див. Додаток Ч) на яких розташовані: **лістинг програми** (навести **лістинг** програми, який розглядається в лабораторній роботі №23, див. п.п. 2.4.3) та **висновки**.

Запитання з самоконтролю до розділу 7

1. Які об'єкти використовуються для реалізації підтримки графічних побудов в програмі Delphi??

2. Яка властивість об'єкту Canvas визначає колір і текстуру заповнення фігур?

3. Яку властивість об'єкту Canvas визначає вигляд та характеристики ліній?

4. За допомогою яких компонентів можлива побудова графіків у Delphi?

5. Яка властивість об'єкту Canvas переміщює в поточну позицію заданої координатами (x,y)?

6. Яка властивість об'єкту Canvas малює лінію від крапки до точки заданої координатами (x,y)?

Додаток С

Форма звіту з лабораторної роботи №17

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ НАУКИ І ТЕХНОЛОГІЙ

Кафедра «Технологія машинобудування»

ЗВІТ

з лабораторної роботи №17
«Структурне програмування. Підпрограма – процедура»
з дисципліни «Об'єктно-орієнтоване програмування»

Виконав ст. гр. _____
(№ групи)
_____
(П.І.Б.)

Перевірив _____
(П.І.Б.)

Дніпро 20 __

Тема:

Мета роботи:

Розробка програми
(навести **умову** та **формули**, які розглядаються в лабораторній роботі №17, див. п.п. 2.4)

Створення інтерфейсу програми та виведення повідомлень програми
(навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №17, див. п. п. 2.4.1)

Лістинг програми
(навести **лістинг** програми, який розглядається в лабораторній роботі №17, див. п. п. 2.4.3)

Висновки:

Додаток Т

Форма звіту з лабораторної роботи №18

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ НАУКИ І ТЕХНОЛОГІЙ

Кафедра «Технологія машинобудування»

ЗВІТ

з лабораторної роботи №18
«Структурне програмування. Підпрограма - функція»
з дисципліни «Об'єктно-орієнтоване програмування»

Виконав ст. гр. _____
(№ групи)
_____
(П.І.Б.)
Перевірив _____
(П.І.Б.)

Дніпро 20 ____

Тема:

Мета роботи:

Розробка програми
(навести **умову** та **формулу**, які розглядаються в лабораторній роботі №18, див. п. п. 2.4)

Створення інтерфейсу програми та виведення повідомлень програми
(навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №18, див. п. п. 2.4.1)

Лістинг програми
(навести **лістинг** програми, який розглядається в лабораторній роботі №18, див. п. п. 2.4.3)

Висновки:

Додаток У

Форма звіту з лабораторної роботи №19

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ НАУКИ І ТЕХНОЛОГІЙ

Кафедра «Технологія машинобудування»

ЗВІТ
з лабораторної роботи №19
«Текстові файли. Тип файл»
з дисципліни «Об'єктно-орієнтоване програмування

Виконав ст. гр. _____

(№ групи)

_____

(П.І.Б.)

Перевірив _____

(П.І.Б.)

Дніпро 20 ____

Тема:

Мета роботи:

Розробка програми
(навести **умову** та **формулу**, які розглядаються в лабораторній роботі №19, див. п. п. 2.4)

Створення інтерфейсу програми та виведення повідомлень програми
(навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №19, див. п. п. 2.4.1)

Лістинг програми
(навести **лістинг** програми, який розглядається в лабораторній роботі №19, див. п. п. 2.4.3)

Висновки:

Додаток Ф

Форма звіту з лабораторної роботи №20

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ НАУКИ І ТЕХНОЛОГІЙ

Кафедра «Технологія машинобудування»

ЗВІТ

з лабораторної роботи №20

«Графічні компоненти Delphi. Компонент Image»

з дисципліни «Об'єктно-орієнтоване програмування

Виконав ст. гр. _____

(№ групи)

_____

(П.І.Б.)

Перевірив _____

(П.І.Б.)

Дніпро 20 ____

Тема:

Мета роботи:

Розробка програми

(навести **умову** та **формулу**, які розглядаються в лабораторній роботі №20, див. п. п. 2.4)

Створення інтерфейсу програми та виведення повідомлень програми

(навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №20, див. п. п. 2.4.1)

Лістинг програми

(навести **лістинг** програми, який розглядається в лабораторній роботі №20, див. п. п. 2.4.3)

Висновки:

Додаток Х

Форма звіту з лабораторної роботи №21

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ НАУКИ І ТЕХНОЛОГІЙ

Кафедра «Технологія машинобудування»

ЗВІТ

з лабораторної роботи №21

«Графічні компоненти Delphi. Компонент Chart»

з дисципліни «Об'єктно-орієнтоване програмування»

Виконав ст. гр. _____

(№ групи)

_____

(П.І.Б.)

Перевірив _____

(П.І.Б.)

Дніпро 20 ____

Тема:

Мета роботи:

Розробка програми
(навести **умову**, яка розглядається в лабораторній роботі №21, див. п.п. 2.4)

Створення інтерфейсу програми та виведення повідомлень програми
(навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №21, див. п.п. 2.4.1)

Лістинг програми
(навести **лістинг** програми, який розглядається в лабораторній роботі №21, див. п.п. 2.4.3)

Висновки:

Додаток Ц

Форма звіту з лабораторної роботи №22

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ НАУКИ І ТЕХНОЛОГІЙ

Кафедра «Технологія машинобудування»

ЗВІТ

з лабораторної роботи №22

«Графічні моделі Delphi. Властивість Canvas. Властивість Pixels»

з дисципліни «Об'єктно-орієнтоване програмування»

Виконав ст. гр. _____

(№ групи)

_____

(П.І.Б.)

Перевірив _____

(П.І.Б.)

Дніпро 20 ____

Тема:

Мета роботи:

Розробка програми
(навести **умову**, яка розглядається в лабораторній роботі №22, див. п.п. 2.4)

Створення інтерфейсу програми та виведення повідомлень програми
(навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №22, див. п.п. 2.4.1)

Лістинг програми
(навести **лістинг** програми, який розглядається в лабораторній роботі №22, див. п.п. 2.4.3)

Висновки:

Додаток Ч

Форма звіту з лабораторної роботи №23

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ НАУКИ І ТЕХНОЛОГІЙ

Кафедра «Технологія машинобудування»

ЗВІТ

з лабораторної роботи №23

«Графічні моделі Delphi. Властивість Canvas. Метод LineTo»

з дисципліни «Об'єктно-орієнтоване програмування»

Виконав ст. гр. _____

(№ групи)

_____

(П.І.Б.)

Перевірив _____

(П.І.Б.)

Дніпро 20 ____

Тема:

Мета роботи:

Розробка програми
(навести **умову**, яка розглядається в лабораторній роботі №23, див. п.п. 2.4)

Створення інтерфейсу програми та виведення повідомлень програми
(навести **інтерфейс** програми та **вивід повідомлень** програми, які розглядаються в лабораторній роботі №23, див. п.п. 2.4.1)

Лістинг програми
(навести **лістинг** програми, який розглядається в лабораторній роботі №23, див. п.п. 2.4.3)

Висновки:

ЛІТЕРАТУРА

Основна

1. Керман Митчелл К. Программирование и отладка в Delphi : учеб. курс. Москва : Вильямс, 2004. 720 с.
2. Дудзяний І. М. Програмування мовою Object Pascal : навч. посіб. Львів : Видавничий центр ЛНУ імені Івана Франка, 2003. 328 с.
3. Безменов М. І. Основи програмування у середовищі Delphi : навч. посіб. Харків : НТУ "ХПІ", 2010. 608 с.
4. Основи програмування. Delphi 6 : навч. посіб. / Алексеєв М. О., Кандзюба С. П., Коротенко Л. М., Шевцова О. С. Дніпропетровськ : Національний гірничий університет, 2013. 272 с.
5. Основи об’єктно-орієнтованого програмування : навч. посіб. / Семйон І. В., Чупов С. В., Брила А. Ю., Апшай Н. І. Ужгород, 2011. 141 с.

Додаткова

6. Алхімова С. М. Об’єктно-орієнтоване програмування : підручник : у 2-х ч. Київ : КПІ ім. Ігоря Сікорського, 2019. Ч. 2 : Об’єктно-орієнтований підхід до розробки програмного забезпечення. 192 с.
7. Омельчук Л. Л. Об’єктно-орієнтоване програмування. Лабораторний практикум : навч. посіб. Київ, 2021. 265 с.
8. Лабораторний практикум з програмування : навч. посіб. / за заг. ред. проф. А. П. Власюка. Рівне : НУВГП, 2011. 495 с.

Інформаційні ресурси в Інтернеті

9. Pascal. Вступ. Алгоритм. Структура програми. Компілятор. *YouTube*. URL: https://www.youtube.com/watch?v=bws2rE6bNew (дата звернення: 10.02.2024р.).
10. Delphi: IDE Software Overview. *Embarcadero*. URL: https://www.embarcadero.com/products/delphi/ (date of access: 10.02.2024р.).

Навчально-методичне видання

Гришин Володимир Сергійович,
Карабут Владлен Миколайович

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Навчально-методичні рекомендації
до виконання лабораторного практикуму
Частина ІІІ

Електронне видання

Експертний висновок склав д-р техн. наук, проф. Володимир Анісімов

Зареєстровано НМВ УДУНТ (№ 34 від 27.03.2025)

В авторській редакції
Комп’ютерна верстка Карабут В. М.

Формат 60×84 1/16. Ум. друк. арк. 6,62. Обл.-вид. арк. 6,71.
Зам. № 43.

Видавець: Український державний університет науки і технологій
вул. Лазаряна, 2, ауд. 2216, м. Дніпро, 49010.
Свідоцтво суб’єкта видавничої справи ДК № 7709 від 14.12.2022

Адреса видавця та дільниці оперативної поліграфії:
вул. Лазаряна, 2, Дніпро, 49010