math_2021_072

Міністерство освіти і науки України

Чернівецький національний університет імені Юрія Федьковича

Факультет математики та інформатики

Кафедра диференціальних рівнянь

Методика навчання учнів 9 класів візуальному програмуванню з використанням середовища Lazarus

Дипломна робота

Рівень вищої освіти – другий (магістерський)

Виконала:

Студентка 6 курсу, групи 606

Спеціальності 014.04 «Середня освіта (математика)»

Станішевська Любов Мстиславівна

Керівник к.ф.-м.н, доцент Лучко В.М.

До захисту допущено:

Протокол засідання кафедри №___

Від «__»_____ 2021 р.

Зав. кафедри _____ проф. Пукальський І.Д.

Чернівці - 2021

Анотація

У дипломній роботі вивчається середовище програмування Lazarus та його використання при вивченні основ програмування учнями 9 класів.

Зміст

Вступ

У даний час візуальне програмування має широке застосування, і має свої характерні особливості, тому не можна просто перенести методику навчання структурованого програмування на вивчення програмування у візуальних середовищах.

Використання систем візуального програмування пов'язане з можливістю спрощення навчання процесу програмування школярів. Вважається, що візуальний спосіб опису програми доступніший для розуміння програмістів-початківців, оскільки графічне зображення відображає реальний світ, тоді як текст вказує лише на його об'єкти. Крім того, багатовимірність графічних зображень може збільшити інформативність у порівнянні з одновимірним потоком тексту за рахунок використання, наприклад, форми, розміру, кольору, текстури, напрямку або відстані. Тобто візуальний метод опису був покликаний знизити рівень абстракції подання алгоритмів.

Навчання візуальному програмуванню починається з опису особливостей та переваг складання програм у візуальному середовищі, що на початку сприяє мотивації учнів. На першому етапі вивчення візуального програмування потрібно приділити увагу основним етапам створення проектів, розгляд основних вікон середовища програмування, також вивчити структуру проекту та основні поняття візуального програмування, виділити особливості конструювання графічного інтерфейсу проекту, зміни властивостей компонентів за допомогою Інспектора об'єктів та через програмний код. Початковий розгляд середовища програмування та основних понять візуального програмування тісно пов'язані із введенням компонентів, їх властивостей та елементів програмування.

Візуальне програмування актуальне не лише своєю сучасністю і простотою у використанні, а й розвитком і вихованням в учнів почуття міри, краси, гармонії. Вони виражаються у прагненні красиво помістити

компоненти, підібрати їх колір, розмір, формати шрифту та інше. Візуалізація процесу дозволяє значно швидше побачити результати своїх зусиль і робить його наочнішим.

Візуальне програмування – спосіб створення програми для ЕОМ шляхом маніпулювання графічними об'єктами замість написання їх коду. Засобами візуального програмування найчастіше вирішують завдання по створенню інтерфейсу користувача та спрощення розробки програми шляхом заміни методу «написання програми» на метод конструювання програми.

Візуальне програмування, наочно представляє інформацію та краще відповідає природі людського сприйняття. Однак, всі візуальні засоби програмування потребують доповнення, тому що не можуть бути представлені у вигляді графічних конструкцій і вимагають текстового коду. Візуальні засоби доповнюють спеціальними програмами – «скриптами», які написані різними мовами програмування.

Концепція візуального програмування реалізована у багатьох сучасні середовища розробки програмних систем.

У дипломній роботі візуальне програмування розглядатиметься з використанням середовища Lazarus.

Lazarus – мова візуального програмування, це візуальний компілятор, який працює у середовищі Windows на базі мови Object Pascal. З появою Lazarus відпала необхідність програмувати стандартні елементи керування Windows, це все вже є у вигляді готових шаблонів компонентів.

Вибір Pascal як мови для навчання зроблений невипадково:

по-перше, Pascal визнаний сам найкращою мовою для навчання основ програмування в основній школі;

по-друге, принципи програмування, закладені в Pascal, знаходять своє відображення в інших мовах програмування, отже, вивчивши Pascal, можна легко перейти до будь-якої іншої мови програмування;

по-третє, Pascal вивчається у більшості вищих навчальних закладів країни.

Вивчення Lazarus у шкільному курсі є логічним продовженням неперервного курсу інформатики у школі. Опанування основ програмування на Lazarus дозволить учням реалізувати свої творчі проекти відповідно до сучасних вимог.

Вивчення мови програмування Lazarus у шкільному курсі має низку переваг:

- Lazarus – сучасна мова візуального програмування, яка відображає всі світові тенденції у інформаційних технологій;
- концепція мови візуального програмування проста і зрозуміла вже програмісту-початківцю;
- в основі Lazarus лежить мова програмування Pascal, яка вивчається в більшості шкіл;
- вивчення Lazarus пов'язане з допоміжними розділами курсу інформатики таких, як: операційні системи, створення та редагування тексту, графічні об'єкти, мультимедіа.

Актуальність дослідження. Вибрана тема дипломної роботи актуальна, оскільки застосування візуального програмування у навчальному процесі надає нові можливості та дозволяє підвищити якість усіх видів навчальної діяльності.

Запропоновані методичні рекомендації щодо проведення уроків з візуального програмування, можуть бути використані вчителями інформатики у 9 класах, з метою формування умінь та навичок використовувати візуальне програмування з використанням середовища Lazarus.

§1. Теоретичні основи використання середовища візуального програмування на уроках інформатики

1.1. Огляд середовищ розробки візуального програмування

Для учнів дев'ятих класів з основ інформатики має бути не більше 2 уроків на тиждень. Безперервна тривалість занять не повинна перевищувати:

- у V – VI класах – 30 хв.;
- у VII – XI класах – 35 хв.

Структура уроку та форми організації навчальної роботи на ньому мають важливе значення у теорії та практиці сучасного уроку, оскільки значною мірою визначають ефективність навчання, його результативність. Орієнтовна структура уроку:

1) вивчення нового матеріалу;

2) закріплення пройденого;

3) контроль та оцінка знань учнів;

4) домашнє завдання;

5) узагальнення та систематизація знань.

Вчені-педагоги єдині в тому, що структура уроку не може бути випадковою, вона повинна відображати: закономірності процесу навчання як явища, що відображає дійсність; логіку процесу вчення; закономірності процесу засвоєння, логіку засвоєння нових знань як внутрішнього психологічного явища; закономірності самостійної розумової діяльності учня як способів його індивідуального пізнання, виражають логіку пізнавальної діяльності, логіку викладання; види діяльності вчителя та учнів як зовнішні форми прояви сутності педагогічного процесу Елементами уроку, які при відображають ці закономірності, є актуалізація, формування нових понять та способів дій та застосування засвоєного матеріалу. «Гарний» урок – це той урок, де панує ділова творча атмосфера, де бажання учнів охоче вступати у

діалог з учителем, один з одним, з авторами тих чи інших теоретичних концепцій та побажань, не побоюючись помилитися.

Вчителю необхідно звертати увагу на підготовку та план уроку, на слабо успішних учнів. У плануванні уроку та розробці технології його проведення виділяються дві взаємозалежні частини:

1) обмірковування мети уроку, кожного його кроку;

2) запис у спеціальному зошиті у тій чи іншій формі плану уроку.

Успіх уроку залежить не тільки від ретельної підготовки до нього вчителем, але й від підготовки самих учнів до роботи на майбутньому уроці, психологічного настрою, із яким вони приходять на урок. Підготовка учнів до наступного уроку у системі уроків передбачає:

- ознайомлення учнів із планом їх роботи на майбутніх уроках за даною темою;
- орієнтація учнів на їхнє попереднє знайомство з окремими розділами або темами підручника, читання науково-популярної та художньої літератури з проблем чергового уроку, проведення спостережень та нескладних доступних дослідів, які можуть сприяти вивченню нового матеріалу.

Інтегроване середовище розробки програм Lazarus. Lazarus відноситься до систем візуального програмування, які називають також системами швидкої розробки додатків [1, с. 15]. Розробка програм в Lazarus включає два взаємопов'язані етапи:

- створення інтерфейсу програми;
- визначення функціональності програми. [2, с. 15]

Інтерфейс користувача встановлює спосіб зв'язку між користувачем та додатком. Інтерфейс створюється шляхом розміщення у формі компонентів, які називають елементами управління. Для написання коду використовується вікно коду, для створення інтерфейсу програми – використовується вікно форми, вікно інспектора об'єктів та головне вікно (Рис. 1.1).

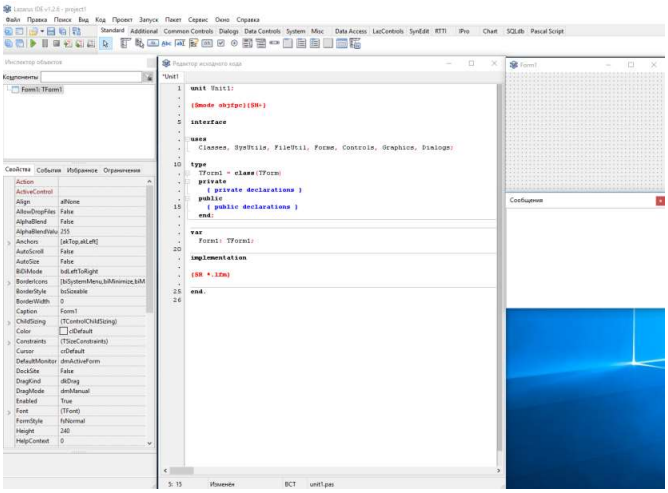

Рис. 1.1. – Вікно візуального середовища Lazarus

Між вмістом вікон форми та коду існує взаємозв'язок, оскільки розміщення на формі компонента призводить до автоматичної зміни коду програми. Тому спочатку треба сконструювати форму, розміщуючи на ній усі компоненти програми, а потім переходити до написання коду програми, забезпечуючи потрібні дії компонентів програми.

Коли відкривається вікно програми, зазвичай відображається безліч кнопок та розділів меню. Програма може містити вікна редагування, реагування, відображення списків тощо. Все це є об'єктами. Подіями є будь-які дії користувача та операційної системи. Об'єкти обмінюються між собою за допомогою обміну повідомлень.

У Lazarus компоненти – це об'єкти, що реагують на події. Компонент, розміщений на формі, як і будь-який інший об'єкт, відноситься до певного класу [1, с. 110]. Усі компоненти мають властивість Name – назву компонента. Lazarus дає кожному компоненту загальне ім'я Назва_компонента_N, де N змінюється, починаючи з 1. Найкраще змінювати ім'я, щоб воно відображало призначення даного компонента у програмі [3, с. 55]. Усі візуальні компоненти мають властивості:

- Left (Ліворуч), Top (Зверху) – місце розташування лівої верхньої точки компонента;
- Width (Ширина), Height (Висота) – розмір компонента.

Значення цих властивостей визначаються у Lazarus автоматично при розміщенні компонента у вікні форми, і їх можна змінювати в процесі виконання програми [1, с. 111].

При отриманні повідомлення об'єкту необхідно його обробити це відбувається за допомогою оброблювача подій. Побачити, які події здатний обробити компонент можна в Інспекторі об'єктів. З лівого боку представлений список подій, а з правої – назви процедур обробки подій.

Компоненти дозволено розташовувати на формі під час планування програми. Це спрощує роботу з програмою, тому що видно програму, як вона буде виглядати при запуску, вона буде швидше запускатися і працювати. Але ще такий спосіб набагато підвищить обсяг програми. [2, с. 35]

Форма програми показує звичайне вікно Windows. Вона включає в себе стандартні елементи вікна, такі як кнопки меню, розкриття, згортання, закриття та рядок заголовка. При створенні програми форма покрита сіткою з крапок. На формі за допомогою миші розташовуються компоненти. Форма є об'єктом класу TForm. Окремі властивості форми дозволяється змінювати їх значення: Name – ім'я, ActiveControl – активний об'єкт, Caption - заголовок; Font – шрифт.

Активним об'єктом є компонент, який під час виконання програми було виділено клацанням миші. Будь-який об'єкт форми програми може бути активним під час виконання методу SetFocus. При створенні форми відбуваються такі події: OnCreate – створення форми, OnShow – форма стає видимою, OnActivate – форма стає активною. При закритті та видаленні форми відбуваються такі події: OnClose Query – спроба закриття форми, OnClose – форма закривається, OnDeactivate – форма втратила фокус введення, OnHide – стала невидимою, OnDestroy – форма видаляється.

Якщо проект раніше не зберігався, то перед закриттям програми Лазарус запропонує його зберегти. Для цього в меню File потрібно знайти оператор Save Project As... (Зберегти проект як...). На моніторі з'явиться діалогове вікно збереження проекту. Потім потрібно ввести нове ім'я проекту та вибрати папку

для його збереження та вибрати кнопку Зберегти. Вікно проекту та заголовок програми змінять свою назву на те, під яким було збережено проект [4, с. 115].

Проект Lazarus включає форми, модулі, установки параметрів проекту. Це зберігається у файлах проекту. Структура проекту включає:

- ✓ файл проекту (dpr);
- ✓ файл опису форм (dfm);
- ✓ файли модулів та модулів форм (pas);
- ✓ файл параметрів проекту (dot);
- ✓ файл параметрів середовища (cfg);
- ✓ файл опису ресурсів (res);
- ✓ файл конфігурації вікон (dsk) [4, с. 117].

Файл проекту вважається основним файлом програми, що включає в себе програму. Файл опису форми формується в Lazarus машинально при створення форми і охоплює опис властивостей форми та її компонентів. Файл ресурсів включає значки, растрові зображення і курсори. Після компіляції створюється виконуваний файл з ім'ям проекту.

Форми, що відображають різноманітні повідомлення та запитують від користувача введення будь-якої інформації, називають діалоговими вікнами. Деякі форми дають можливість здійснити двосторонню передачу інформації, але головним завданням діалогових вікон є передача інформації від користувача до програми. Діалоги можна відкрити тільки в модальному режимі та на час передачі інформації.

Коли показана модальна форма, всі повідомлення, що надходять обробляються модальною формою. Вибір способу створення діалогу залежить від завдання, що розв'язується.

Головні компоненти Lazarus знаходяться на вкладках Standard та Additional знаходяться стандартні та додаткові компоненти палітри. На вкладці Standard – стандартних компонентів, розміщуються основні елементи та органи управління інтерфейсу (Рис.1.2.).

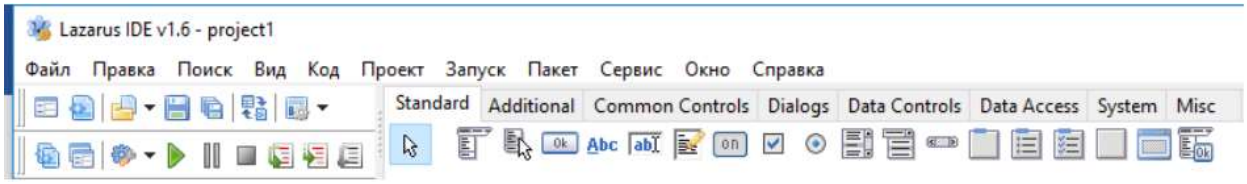

Рис. 1.2. – Вкладки основних компонентів Lazarus

MainMenu – головне меню;

Popup Menu – спливаюче меню;

Button – стандартна кнопка;

Label – мітка;

Edit – однорядковий редактор;

Memo - багаторядковий редактор;

CheckBox -прапорець;

RadioButton – залежний перемикач;

ListBox – список;

ComboBox – поле зі списком;

ScrollBar – смуга прокручування;

GroupBox – група;

RadioGroup – група залежних перемикачів;

Panel – панель;

Frames – фрейм;

ActionList – перелік дій.

Компоненти наведено в порядку їх розташування на сторінці [4, с. 124].

На вкладці Additional додаткових компонентів розміщується ряд додаткових елементів керування, застосування яких підвищує функціональні можливості програми. (Рис. 1.3.).

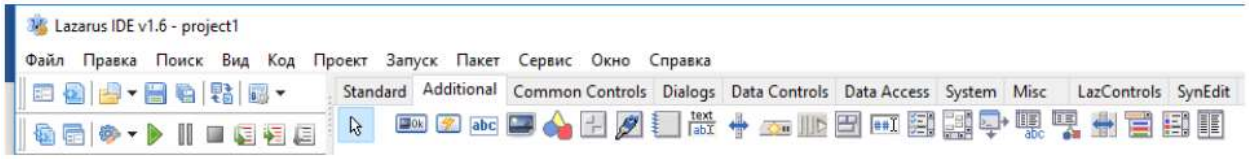

Рис. 1.3. – Вкладка додаткових компонентів Lazarus

BitButton – кнопка із малюнком;

SpeedButton – кнопка швидкого доступу;

StaticText – статичний текст;

Image – графічне зображення;

Shape – геометрична фігура;

Bevel – фаска;

LabledEdit – однорядковий редактор із написом;

Splitter – роздільник;

ControlBar – контейнер для панелі інструментів;

MaskEdit – однорядковий редактор із введенням за шаблоном;

CheckListBox – список прапорців;

ScrollBox – область прокручування;

StringGrid – таблиця рядків;

DrawGrid – таблиця;

ColorBox – комбінований перелік вибору кольорів;

Va1ueListEditor - редактор списку значень[4, с. 125].

Найбільш часто використовувані візуальні компоненти управління програми: список, форма, редактор, кнопки, перемикачі, множинний вибір та картинки.

Властивості дають можливість регулювати зовнішній вигляд та дії компонентів під час створення та відтворення програми. Деякі властивості компонентів, можна редагувати під час створення та виконання програми. Так є властивості, редагування яких можливе тільки при виконанні програми.

Проаналізуємо властивості компонентів.

Властивість Name вказує назву компонента.

Властивість Caption містить рядок заголовка компонента.

Властивість Color визначає колір поверхні компонента.

Властивість Enabled типу Boolean визначає здатність компонента реагувати на повідомлення, що надходять [4, с. 127].

Текст, передбачає напис і як правило застосовується в якості заголовків для інших елементів управління, які не мають властивості Caption. Найчастіше для виведення написів застосовується компонент Label.

Ключові властивості компонента Label:

Caption – напис, що відображається цим компонентом;

AutoSize типу Boolean – що регулює автоматичні зміни розмірів мітки.

Візуальні компоненти здатні створювати та обробляти декілька десятків подій різних видів. Розглянемо найзагальніші події.

Подія OnClick з'являється при підборі елемента керування, та називаються подією натискання, тому що вона відбувається при натисканні мишею на компонент.

Клацання кнопки миші на компоненті робить дві події OnMouseDown та OnMouseUp: перше – при натисканні кнопки миші, друге – при відпусканні кнопки.

При подвійному натисканні лівою кнопкою миші в області компонента, крім того, генерується подія OnDblCIick.

Коли вказівника миші знаходиться, над візуальним компонентом спрацьовує подія OnMouseMove, в процедуру якої передаються дані про компонент, над яким знаходиться покажчик, та координати покажчика.

Компонент кнопка належить елементам керування інтерфейсом програми та необхідні для подачі команд для певних функціональних процесів. Кнопка називається компонентом Button. Властивість кнопки Заголовок можна змінити інше значення.

Подія - OnClick, виникає при натисканні мишею на кнопку або натисканням комбінації клавіш, заданих як Caption.

Кнопка з малюнком Lazarus визначається компонентом BitBtn. Поряд із написом компоненти може відобразити графічне зображення. Зображення кнопки представлене властивістю Glyph і спочатку має значення None – немає зображення.

Компонент Edit допускає введення та редагування тексту. Зміст компонент визначається властивістю Text; типу string - рядок символів, що відображається цим компонентом [5, с. 132].

Компонент Memo є редактором багаторядкового тексту, в якому текст відображається однаково відповідно до загального стилю. Компонент володіє багатьма функціями текстового редактора, такими як виділення тексту, роботу з буфером обміну, скасування останніх команд. Весь текст компонента Memo, розташований одним рядком типу string, усередині якого використовуються роздільникипереводу з одного рядка. Для видалення тексту в редакторі Memo треба використовувати метод Clear.

Основною властивістю вікна Memo є Lines, що включає текст у вигляді списку, який має тип TStrings. Клас TStrings має такі методи:

Add(const s : string): Integer – додає до кінця тексту рядок s і повертає її номер;

Delete (n : Integer) - видаляє рядок із номером n;

Insert(n: Integer; s : string) - вставляє рядок s у позицію n;

Exchange (n1, n2, Integer) – змінює місцями рядки з номерами n1 та n2;

Move(n1, n2 : Integer) – переміщує рядок із номером n1в позицію n2;

IndexOf (s: string): – повертає номер рядка s або – 1, якщо рядка у тексті немає;

LoadFromFile(FileName.string): Integer завантажує текст із файлу з ім'ям FileName;

SaveToFile(FileName:string) - зберігає текст у файлі з ім'ям FileName [4, с. 134].

Є компоненти, які використовуються для редагування інформації, вони мають спільні властивості, події та методи. Властивість MaxLength визначає максимальну кількість символів у тексті, що вводиться; довжина тексту не обмежена. Якщо властивість ReadOnly встановлено, то текст призначений лише для читання та без можливості редагування. Список є впорядкованою сукупністю текстових елементів.

Простий список ListBox являє собою прямокутну область, у якій розташовані його рядки. Елементи списку можна відсортувати за абеткою у порядку зростання за допомогою властивості Sorted типу Boolean. Якщо встановлено значення True це означає, що елементи списку будуть

автоматично відсортовані. Доступ до рядка списку здійснюється за номером у масиві рядків Items. При виконанні програми потрібно працювати з окремими рядками списку за допомогою властивостей та методів властивості Items. Властивість лише для читання Count вказує кількість рядків у тексті. Номер першого рядка дорівнює 0, номер останнього дорівнює Count – 1. Властивість Index визначає номер вибраного елемента списку. Основна подія - OnClick, виникає при натисканні кнопки миші на елемент списку. Його можна використовувати для обробки вибраних рядків.

Для видалення списку ListBox необхідно виконати його метод Clear. Таблиця рядків StringGrid є компонентом, який дозволяє відображати текстові дані, подані у вигляді таблиці. Таблиця рядків дає можливість відображати як текстову так і графічну інформацію, але у цьому випадку зберігання графічної інформації та їх малювання програміст реалізує самостійно.

Розміри таблиці встановлюються властивостями ColCount та RowCount, що задають максимальну кількість рядків та стовпців. Значення властивостей можна задавати як у процесі проектування програми, так і в процесі виконання програми, що призводить до негайної зміни розмірів таблиці. Крайні ліві стовпці та верхні рядки таблиці можна робити фіксованими. Фіксація використовується для оформлення заголовків, оскільки під час перегляду інформації фіксовані елементи не рухомі. Число фіксованих стовпців та рядків визначають властивості FixedCols та FixedRows. Для доступу до певної комірки таблиці призначається властивість Cells[ACol, ARow] типу string, що є двовимірним масивом рядків. Індекс ACol визначає номер поточного стовпця, а індекс ARow – номер поточного рядка вибраної комірки в таблиці [4, с. 81].

Компонент CheckBox це прапорець або незалежний перемикач, у якого є два положення «Увімкнено» або «Вимкнено». Прапорець діє спільно в рамках однієї групи компонентів: якщо вибирається один, то всі інші мають бути вимкнені.

Основні властивості компонента CheckBox:

Caption - пояснювальний текст;

Checked типу Boolean вказує, чи відмічено прапорець: якщо Checked то значення True, то прапорець відмічено, якщо False, то не відзначено;

State вказує на стан прапорця:

cbChecked - включений;

cbUnchecked - не включено;

cbGrayed – недоступний [4, с. 143].

Коли змінюється стан прапорця, виникає подія OnClick. У обробник цієї події зазвичай знаходяться інструкція, яка перевіряє стан прапорця та виконує необхідні дії.

Перемикачі зазвичай розташовуються групами, візуально виділеними у формі. Вибір перемикача є взаємовиключним, тобто при виборі одного перемикача всі інші стають невибраними. Lazarus підтримує автоматичне групування перемикачів. Кожен перемикач, поміщений у контейнер, переміщується у групу , що знаходиться у цьому контейнері . Контейнерами зазвичай є такі компоненти, як форма Form, панель Panel, група GroupBox [4, с. 145].

При розробці програм часто виникає завдання об'єднати або згрупувати різні елементи управління. Об'єднання елементів виконується за допомогою компонентів –контейнерів. Контейнер є візуальним компонентом, на поверхні якого можна розміщувати інші компоненти. Форма програми є основним контейнером, з якого зазвичай і починається проектування інтерфейсу. Форма є власник всіх розміщених на ній компонентів.

При створенні простих графічних зображень використовуються графічні компоненти, такі як Shape – геометрична фігура, Bevel – межа та Image - графічне зображення.

Для відображення геометричних фігур використовується компонент Shape. Вигляд фігури визначається однойменною властивістю Shape, яка може приймати значення: stCircie - коло, stEmpse - еліпс, stSquare - квадрат,

stRectangle - прямокутник, stRoundRest - прямокутник із заокругленими кутами, stRoundSquare - квадрат із округленими кутами [5, с. 148].

Компонент Bevel являє собою прямокутні області, рамки і лінії якими можна скористатися для візуального виділення або поділу інших елементів форми. Стиль відображення Bevel визначається властивістю Style, яка може приймати значення bsLowered.

Компонент Image використовується для відображення зображення.

Основна властивість – Picture, що визначає зображення, яке розміщується всередині компонента зображення. Зображення з графічного файлу можна завантажувати на етапі проектування програми та при його виконанні. Потрібно враховувати, що зображення, підключене під час проектування, значно збільшує обсяг файлу програми, що виконується. Щоб уникнути цього, рекомендується завантажувати великі зображення динамічно.

У вкладці Dialogs панелі компонентів знаходяться компоненти, які виконують основні стандартні дії з діалоговими вікнами:

OpenDialog - вибір файлу, що відкривається;

SaveDialog - вибір файлу, що зберігається;

OpenPictureDialog - вибір графічного файлу, що відкривається;

SavePictureDialog - вибір графічного файлу, що зберігається;

FontDialog – налаштування параметрів шрифту;

ColorDialog – вибір кольору.

Для виклику будь-якого стандартного діалогового вікна використовується метод Execute – функція, що повертає логічне значення. При закритті діалогу кнопкою Відкрити функція Execute повертає значення True, а при скасуванні діалогу кнопкою Скасування – значення False [3, с. 110].

Інтегроване середовище розробки програм Microsoft Visual Studio.

Microsoft Visual Studio – це інтегроване середовище розробки (IDE – Integrated Development Envirionment) розроблена спеціально для створення, запуску та налагодження програм, написаних мовами програмування [6, с. 14].

Головне вікно Microsoft Visual Studio, аналогічно до інших програм, включає рядок меню, що містить (Рис. 1.4):

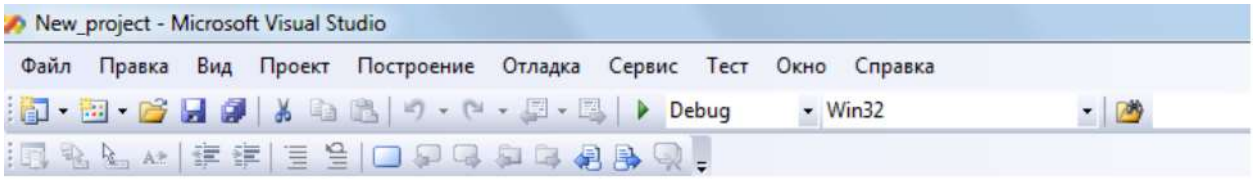

Рис. 1.4 – головне меню Microsoft Visual Studio

Вкладка файл включає в себе дії для відкриття, створення, додавання, закриття та друку програми. Вкладка *редагування* містить основні команди редагування: копіювати, вирізати, вставити, виділити. Вкладка *вид* включає команди для закриття і відкриття всіх вікон і панелей інструментів. Вкладка *проект* містить команди для створення, відкриття, збереження проектів, а також інспектор об'єктів. Вкладка *побудова* включає команди компіляції програми. Вкладка *налагодження* містить команди для налагодження програми. Вкладка *сервіс* включає команди додаткових параметрів і інструментів для налаштування Microsoft Visual Studio. Вкладка *вікно* включає управління розташуванням вікон. Вкладка *довідка* включає виклик довідкової інформації [6, с. 21].

Панель інструментів розміщена під рядком меню, і містить вкладені панелі кнопок, що запускають команди із встановленої групи або керують середовищем розробки Microsoft Visual Studio.

У лівій частині діалогового вікна можна вибрати тип проекту. Список типів проектів обумовлений тим, які мови програмування були вибрані під час інсталяції Microsoft Visual Studio. При розробці нового проекту у середовищі Microsoft Visual Studio представляється великий список типів проектів, але з них всього три основні види програм: Windows-додаток, Консольний-додаток та Бібліотека класів. Інші варіанти додатків будуються з використанням різних шаблонів або майстрів, які забезпечують автоматичне виконання деяких початкових дій.

Після вибору типу проекту відкриваються основні вікна візуального середовища Microsoft Visual Studio (Рис. 1.5.)

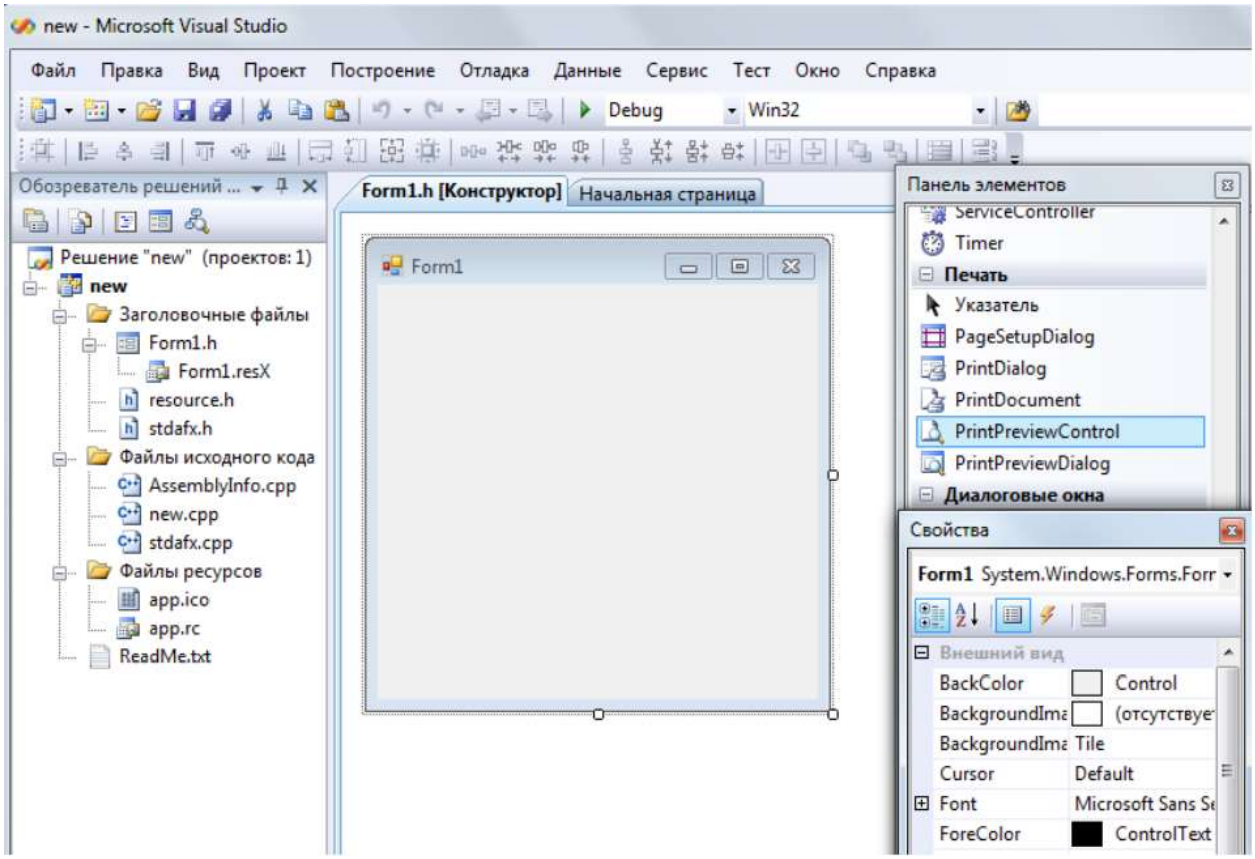

Рис 1.5 – Вікно візуального середовища - Microsoft Visual Studio

Під час створення проекту використовується три основних вікна. В центрі розміщується головне вікно Form1, призначене для створення візуальних компонентів та написання коду програми. Зліва розташовується вікно Оглядача рішень, що дозволяє побачити, які файли входять до складу проекту. А праворуч знаходиться вікно панелі інструментів та вікно інспектора властивостей (Properties), що містить список властивостей активного об'єкта. Вікно Оглядача рішень містить деревоподібну конструкцію, що складається з елементів проекту (Рис. 1.6). У дереві відображається логічне розташування рішень, проектів та елементів рішення. Рішенням називається набір проектів, у тому числі складається додаток. Модулі та файли, необхідні для створення програми можуть бути представлені за допомогою проектних компонентів. Якщо потрібно відредагувати

компонент проекту, його потрібно активувати подвійним клацанням у вікні оглядача рішень [6, с. 35].

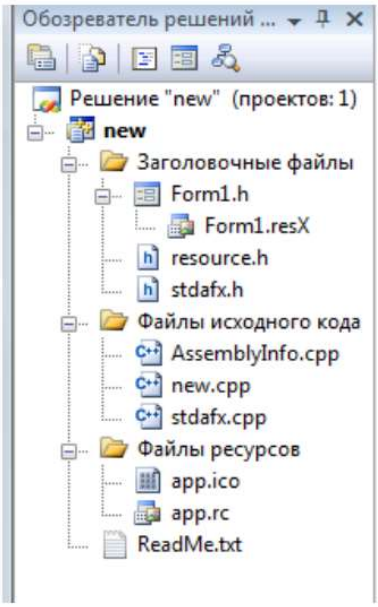

Рис. 1.6. Вікно оглядача рішень

Вміст проекту можна змінювати за допомогою контекстних пунктів меню вікна браузера рішень. Для створення нової програми потрібно вибирати у вікні створення проекту тип проект Win32 і шаблон: Консольний додаток Win32. Після потрібно в полі введення потрібно ввести ім'я проекту та натиснути кнопку ОК. Після чого буде відкрито основне вікно розробки проекту (рис. 1.7).

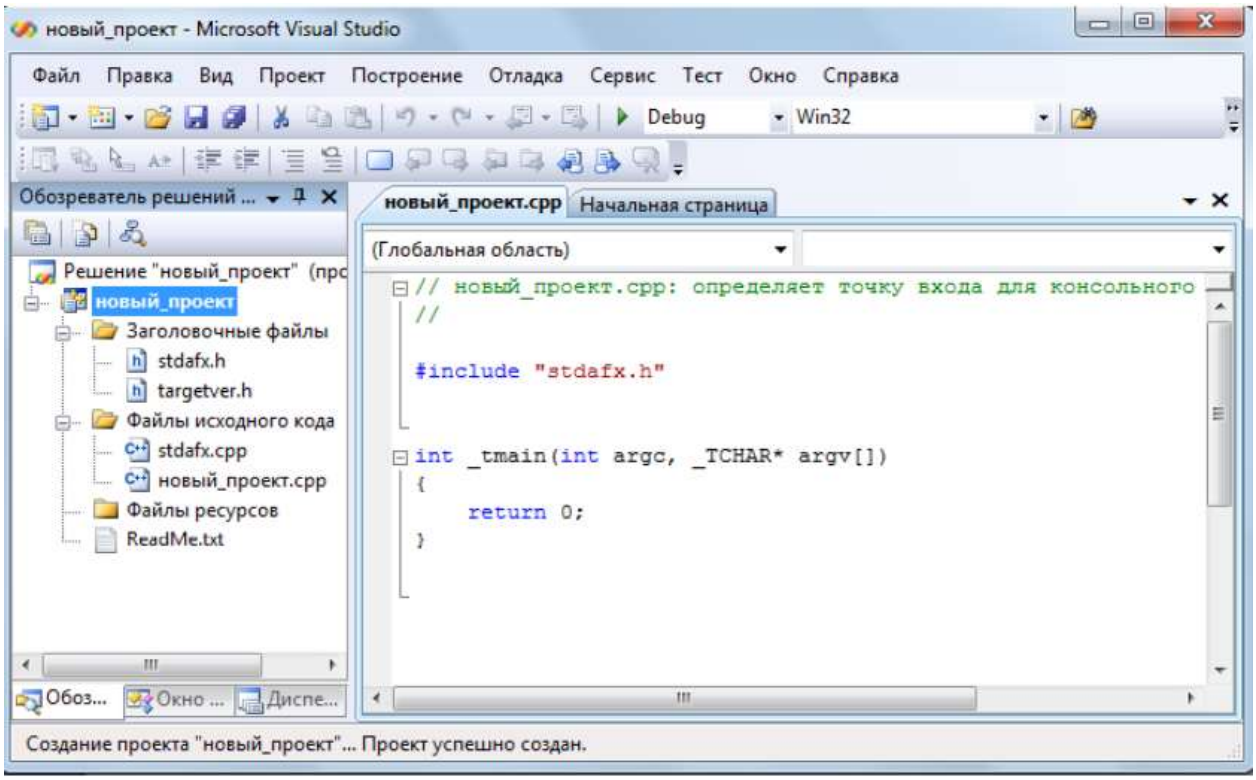

Рис. 1.7. - Новий проект

Є низка способів пошуку логічних помилок у програмі Microsoft Visual Studio. Для виявлення логічної помилки у програмі потрібно організувати точки зупинки програми для вивчення зміни вмісту однієї чи кількох змінних. Для цього потрібно під час роботи програми увійти в режим зупинки, та переглянути код у редакторі коду [6, с. 55].

Режим зупинки програми дає можливість переглянути програму під час її виконання. Точка зупинки показує місце у програмі, де буде здійснено зупинку. Для встановлення точки зупинки потрібно клацнути на сірому поле зліва від вікна з вихідним кодом програми, потім навпроти, вибраного оператора з'явиться червона точка зупинки. Також можна перемістити курсор на потрібний рядок програми та вибрати команду Точка зупинки в меню Налагодження, що призведе до створення точки зупинки, якщо її там не було, або навпаки, прибере її, якщо вона вже була встановлена на цьому рядку. Тепер, після запуску програми в режимі налагодження, її виконання зупиниться при досягненні точки зупинки. Жовта стрілка на червоному

кружку, яка позначає точку зупинки, вона вказує, де точка зупинки перервала виконання програми. Палітра компонентів Microsoft Visual Studio (Рис. 1.8)

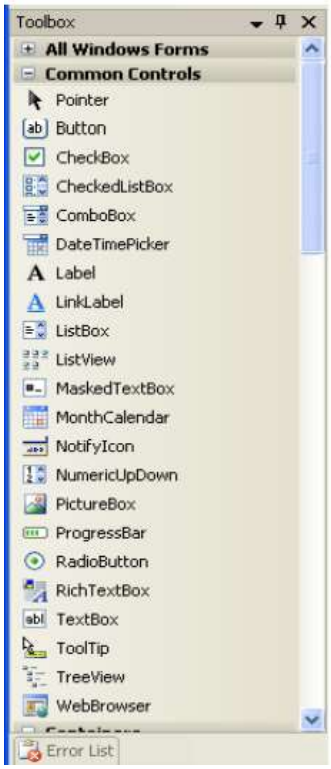

Рис. 1.9. - Палітра компонентів Microsoft Visual Studio

Кнопка Button вважається основним елементом для керування інтерфейсом програми, і призначена для того, щоб приймати та виконувати команди користувача.

Кнопки CheckBox це кнопки відкладеної дії при натисканні на які не повинно виконуватися жодні дії, оскільки з допомогою їх користувач лише вводить параметри елементів. Компонент CheckBox має три стани - помічений, непомічений і змішаний.

Кнопка RadioButton це радіокнопка за допомогою якої є можливість вибрати лише один параметр елемента.

GroupBox – блок угруповання – допомагає візуально об'єднати декілька елементів управління в одну групу і служить для покращення користувацького інтерфейсу.

Label – мітка – використовується лише для відображення тексту.

TextBox – поле введення – використовують як для відображення тексту, так і введення тексту.

MaskedTextBox – поле введення шаблону – дає можливість вказати формат тексту для відображення.

RichTextBox – розширене поле введення – дає можливість користувачеві вводити та обробляти великі обсяги інформації та дозволяє редагувати колір тексту, шрифт та додавати зображення.

ListBox — список, що перегортається – дозволяє вибирати один або декілька елементів, що зберігаються в списку.

ComboBox – список, що випадає.

TrackBar – смуга прокручування.

ProgressBar – індикатор – застосовується для відображення ступеня завершеності.

NumeicUpDown – поле введення числових значень.

TreeView – дерево – використовується для відображення даних як дерева.

ImageList – список зображень – це компонент, у якому зберігаються зображення, що відображаються іншими елементами керування.

DataTimePicker це універсальний візуальний компонент за допомогою якого відбувається подання інформації про час; Timer – таймер [7, с. 121].

1.2. Методика вивчення основних розділів візуального програмування

Даний пункт присвячений опису методики вивчення тем візуального програмування у шкільному курс інформатики. Весь курс інформатики можна поділити на дві частини: теоретичний та практичний матеріал. До теоретичного матеріалу можна віднести: основні поняття візуального програмування, структуру програми Lazarus, а також файли проектів. До практичної частини відноситься все інше це інструменти Lazarus, компоненти

програм, способи роботи з компонентами, форма, властивості, події, методи, розробка проектів та додатків, а також графічні можливості Lazarus.

Тема: «Основні поняття візуального програмування». Ця тема відноситься до теоретичної частини курсу інформатики. Оскільки тема досить велика, то для її вивчення необхідно відвести щонайменше дві навчальні години.

На уроках мають бути представлені такі поняття, як об'єкт, правила опису об'єктів, класи об'єктів. Також слід розглянути візуалізацію об'єктів, опис класів, компоненти, поля, методи та властивості [8, с. 55].

Вимоги до знань та вмінь учнів. Учні повинні знати:

- що таке об'єкт та опис об'єкта;
- що таке клас та його опис;
- призначення полів класу;
- що таке метод та призначення методів;
- що таке компоненти;
- у чому полягає ідея візуалізації об'єкта;
- про динамічний характер об'єктів;
- як використовувати об'єкт у програмі Lazarus.

Тема: «Візуальне середовище програмування Lazarus. Інструменти Lazarus. Основні компоненти Lazarus: властивості, події, методи». Під вивчення цієї теми відводиться трохи більше трьох годин. У процесі вивчення теми учні познайомляться з вікнами в Lazarus (головне вікно, вікно форми, вікно редактора об'єктів, вікно (інспектор об'єктів та вікно повідомлення), їх структурою. Необхідно розглянути властивості та навести приклади властивостей. Потрібно пояснити учням, що властивості компонентів можна змінювати як через вікно Інспектора об'єктів, а також за допомогою написання коду програми. Так само увагу слід приділити і вивченню інспектора об'єктів. На останньому уроці по цій темі рекомендується дати завдання для створення додаток з найпростішим обробником події.

Вимоги до знань та вмінь учнів. Учні повинні знати:

- призначення основних вікон Lazarus;
- призначення основних команд меню;
- призначення основних та додаткових компонентів Lazarus;
- інспектор об'єктів;
- способи змін властивостей.

Учні повинні вміти:

- змінювати властивості;
- написати найпростіший обробник події.

Тема: «Структура програми в Lazarus. Проект. Файли проекту. Опис файлів». Тема вивчається як теоретично, так практично. До теоретичної частини відноситься вивчення структури застосування в Lazarus, основних типів і призначень файлів, що входять до Lazarus-проекту, а до практичної частини відноситься відкриття файлу форми, проекту та модуля форми вже збереженого проекту за допомогою текстового редактора та їх вивчення.

Вимоги до знань та вмінь учнів. Учні повинні знати:

- що входить у процедуру розробки сценарію Lazarus-додатків;
- які основні типи файлів входять у Lazarus-додаток та їх призначення.

Учні повинні вміти:

- створювати папки проектів і розуміти призначення збережених у них файлів;
- встановлювати властивості компонентів за допомогою інспектора об'єктів;
- описувати методи опрацювання подій.

Тема: «Управління компонентами під час проектування. Форма, її властивості, події, методи». Тема вивчається практично. У темі буде розглянута робота з компонентами, такими як: переміщення об'єктів на форму, виділення компонент, видалення компонент, копіювання; переміщення; що таке форма та її властивості, задання розмірів та положення, задання кольорів. До змісту теми також належать основні події форми. Це події створення та появи форми.

Вимоги до знань та вмінь учнів. Учні повинні знати:

- способи роботи з компонентами Lazarus;
- основні властивості форми;
- основні події форми.

Учні повинні вміти:

- поміщати компоненти на форму;
- виділяти компоненти;
- копіювати компоненти;
- змінювати властивості компонентів;
- задавати розміри та положення форми на моніторі;
- змінювати заголовок.

Тема: «Розробка та реалізація простого додатку». Тема вивчається лише практично. При вивченні цієї теми учні користуючись раніше отриманими знаннями, навчаються створити прості програми. Розробка програми складається з наступних етапів. Постановка задачі, що включає: точне формулювання задачі, яка розв'язується; зображення на папері того, що планується побачити на екрані, тобто створення кадру; написання сценарію роботи програми. Розробка форми. Опрацювання подій.

Вимоги до знань та вмінь учнів. Учні повинні вміти:

- чітко формулювати завдання;
- зображати на папері майбутній кадр;
- писати найпростіші сценарії;
- розробляти просту форму, задаючи шрифти, кольори, розміри, розташування на екрані;
- задавати стиль форми;
- описувати обробку нескладних подій.

Тема: «Розробка проекту». Ця тема є продовженням попередньої, але вивчення відбувається на більш високому рівні. Учні навчаються працювати з компонентами меню, писати обробник подій для Головного меню та діалогового вікна.

Вимоги до знань та вмінь учнів. Учні повинні вміти:

- користуватися та створювати меню, перемикачі, діалогові вікна;
- створювати та підключати допоміжні форми.

Тема: «Графічні можливості Lazarus». Тема вивчається лише практично. Учні детальніше вивчають способи виведення графічної інформації за допомогою середовища Lazarus. Графічна інформація поділяється на два види: растрова та векторна, причому Lazarus може використовувати обидва види. Однак якщо растрову графіку можна просто зобразити на формі за допомогою компонента Image, то векторну графіку можна формувати програмним шляхом за допомогою компонента Shape та властивості Canvas.

Вимоги до знань та вмінь учнів. Учні повинні знати:

- які графічні можливості надає Lazarus.

Учні повинні вміти:

- розміщувати на формі готову картинку;
- створювати свою картинку, використовуючи Image Editor;
- зображати прості геометричні фігури, використовувати штрихування та забарвлення;
- формувати зображення програмним способом.

Висновок до першого параграфу

У першому параграфі описано структуру уроку та форми навчальної роботи, з урахуванням особливостей учнів, що навчаються у 9 класах. Використання систем візуального програмування пов'язується із можливістю спрощення навчання під час вивчення програмування учнями. Вважається, що візуальний спосіб опису програми доступніший для мислення програміста-початківця, оскільки картинка відображає реальний світ, тоді як текст лише свідчить про об'єкти реального світу.

Порівнявши дві програми візуального програмування Lazarus та Microsoft Visual Studio, що стосуються систем візуального програмування, було вибрано середовище візуального програмування Lazarus. Перевагою середовища Lazarus є те, що вона безкоштовна, з відкритим середовищем розробки та відкритою ліцензією на відміну від Microsoft Visual Studio. Мова програмування Pascal у середовищі Lazarus є більш структурованою мовою, простіше передавати логіку програми, що позначається краще на програмному навчанні.

§2. Методика навчання візуальному програмуванню у 9 класах

2.1. Система завдань із візуального програмування у середовищі Lazarus

Практичне завдання №1.

Мета: вивчити структуру програми Lazarus, вивчення модуля та основних його компонентів.

Завдання:

1. Завантажте програму Lazarus.

2. Виконайте дії та запишіть результат у таблицю 2.1.:

Таблиця 2.1. – Отримання результату

Дія	Результат
Виберіть у полі Caption Form1 та змініть назву на – ФОРМА	
Надайте властивості Width – значення 150	
Надайте властивості Color – значення clHighlight	
Надайте властивості Height – значення 150	

3. Вивчіть стандартні компоненти Lazarus та запишіть їх призначення (Рис 2.1):

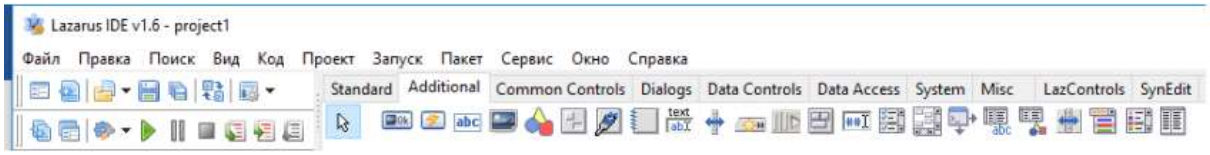

Рис. 2.1. Основні компоненти Lazarus

Запишіть призначення цих компонентів:

ComboBox –

Label –

Edit –

Memo –

Button –

CheckBox –

DrawGrid –

BitButton –

ScrollBar –

RadioGroup –

ListBox –

ScrollBar –

Panel –

Практичне завдання №2.

Мета - навчитися додавати об'єкти та підключати процедуру подій до об'єктів.

Створіть новий проект – форму, додавши на неї дві кнопки Button (Рис. 2.2.).

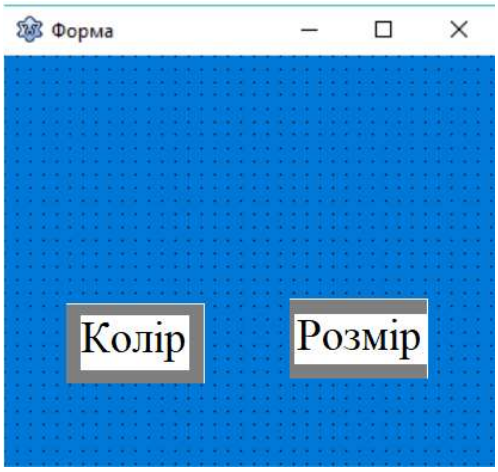

Рис. 2.2. - Інтерфейс програми Lazarus.

Змініть властивості форми та властивості кнопок за таблицею 2.2. Клацніть у полі Caption і змініть назву Form1 на Форма.

Таблиця 2.2. - зміни властивості форми.

Властивість Форми	Значення
Color	clHighlight

Width	300
Height	250
Властивість Button1	Значення
Caption	Колір
Font	Times New Roman – 14 – звичайний
Top	152
Left	40
Height	44
Width	80
Властивість Button2	Значення
Caption	Розмір
Top	150
Left	176
Font	Times New Roman – 14 – звичайний
Height	45
Width	80

Збережіть проект у папці: Перший проект.

Виділіть кнопку Button1, у вікні Інспектора об'єктів, відкрийте вкладку Події, знайдіть подію OnClik і подвійним клацанням по порожньому полю підключіть подію за кнопкою – клацання мишею, у вас у вікні редактора повинен з'явитися шаблон процедури події на кнопці Button1 (Рис. 2.3).

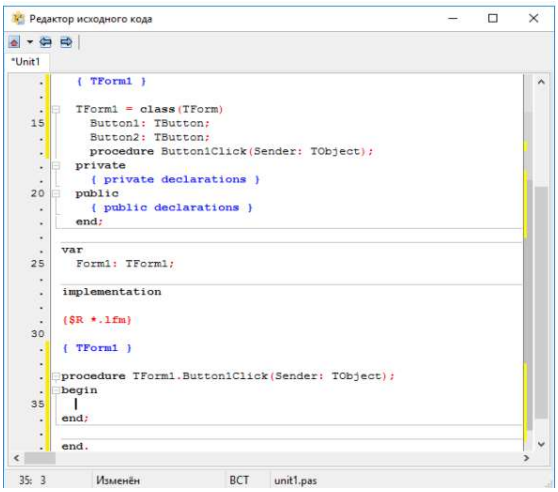

Рис. 2.3. Вікно редактора програми Lazarus.

У вікні редактора коду у процедурі напишіть код програми:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Form1.Color:=clPurple;
end;
```

Далі підключіть процедуру події за клацанням до кнопки Button2 і у процедурі напишіть код програми:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
Form1.Width: = 156;
end;
```

Збережіть проект. Після цього запустіть програму. Натисніть на кнопку Колір, а потім на кнопку розмір. Що змінилося у програмі? Запишіть у зошиті алгоритм створення проекту.

Практичне завдання 3.

Мета – навчитися використовувати компоненти StringGrid та поле Memo, для роботи із масивом.

Розглянемо програму, яка обчислює середнє арифметичне значення елементів масиву Діалогове вікно програми наведено на (Рис. 2.4.). Компонент StringGrid використовується для введення масиву, компонентів Label1 та

Label2 – для виведення пояснювального тексту та результату розрахунку, Button1 – для запуску процесу розрахунку (Рис. 2.4).

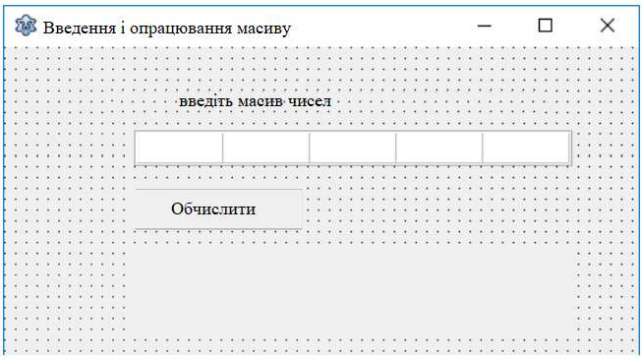

Рис. 2.4. Вікно інтерфейсу програми

Додається компонент StringGrid у форму так само, як і інші компоненти. Після додавання компонента до форми необхідно виконати його налаштування відповідно до таблиці 2.3. Значення властивостей Height та Width потрібно за допомогою миші встановити такими, щоб розмір компонента дорівнював розміру рядка.

Таблиця 2.3. Значення властивостей.

Властивість	Значення
ColCount	5
FixedCols	0
RowCount	1
DefaultRowHeight	22
Height	26
DefaultColWidth	64
Width	324
Options.goEditing	True
Options.AlwaysShowEditing	True
Options .goTabs	True

На кнопку введіть процедуру:

```
procedure TForml.ButtonlClick(Sender: Tobject);
var
a: array[1..5] of integer; // масив
summ: integer; // сума елементів
sr: real; // середнє арифметичне
i: integer; // індекс
begin
// Введення масиву
// вважаємо, що якщо комірка порожня, то відповідний
// їй елемент масиву дорівнює нулеві
for i := 1 to 5 do
if Length(StringGridl.Cells[i-1, 0]) <>0
then a[i] := StrToInt(StringGridl.Cells[i-1,0])
else a[i]: = 0;
// обробка масиву
summ: = 0;
for i :=1 to 5 do
summ: = summ + a [i]; sr: = summ / 5;
Label2.Caption :='Сума елементів: ' + IntToStr(summ)+ #13+ 'Середнє
арифметичне:’ + FloatToStr(sr);
end;
```

Для того, щоб курсор автоматично переходив до наступної комірки таблиці, додамо процедуру обробки події OnKeyPress. Текст процедури Опрацювання події OnKeyPress наведено нижче.

```
Procedure Tform1.StringGridlKeyPress(Sender: Tobject; var Key: Char);
Begin
case Key of
#8,’0’..’9’ : ; // цифри та клавіша <Backspace>
#13: // клавіша <Enter>
```

```
if StringGridl.Col < StringGridl.ColCount-1
then StringGridl.Col := StringGridl.Col+1;
else key := Chr(0); // інші символи заборонені
end;
end;
```

Запустіть програму, перегляньте, як змінилася робота програми. Запишіть алгоритм роботи програми собі у зошит.

Практичне завдання 4.

Мета - Навчитися використовувати властивість об'єкту Canvas для малювання графіки.

Створимо форму, встановимо розміри форми по горизонталі Heigth - 450, та по вертикалі Width - 450. Внизу форми помістіть кнопку Button1 і задайте їй значення Caption - «Малювати». Замініть Form1 значення Caption на «Малюємо фігури». При запуску програми та клацанні по цій кнопці на формі прорисуються різні фігури (Рис. 2.6.). Нижче наведена програма, яка показує роботу перерахованих методів. Результат роботи програми наведено на (Рис. 2.6.).

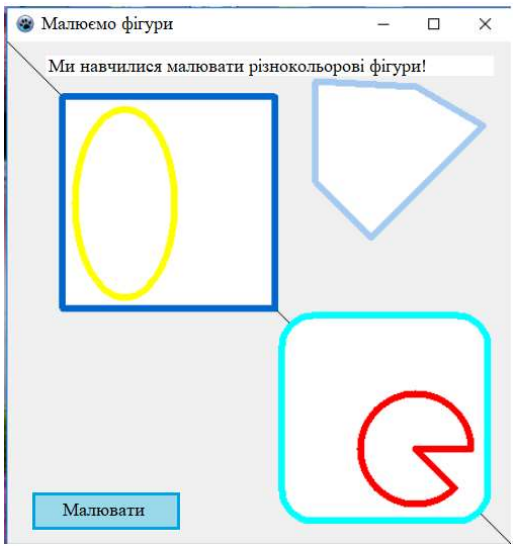

Рис. 2.6. Приклад застосування методів малювання фігур

```
unit Unit1;
{$mode objfpc}{$H+}
interface uses Classes,
```

```
SysUtils, Lresources, Forms, Controls, Graphics, Dialogs, StdCtrls;
type { Tform1 } Tform1 = class(Tform) Button1: Tbutton;
procedure Button1Click(Sender: Tobject);
private { private declarations } public { public declarations }
end;
var
Form1: Tform1;
implementation
{Tform1}
procedure Tform1.Button1Click(Sender: Tobject);
var
t: array [1..5] of Tpoint;
begin
Form1.Canvas.LineTo(500,500); / / Малювання лінії
//Зміна кольору та товщини лінії
Form1.Canvas.Pen.Color:= clHotLight;
Form1.Canvas.Pen.Width:= 6;
//Малювання прямокутника
Form1.Canvas.Rectangle(49,49,240,240);
Form1.Canvas.Pen.Color:= clolive;
//Малювання еліпса.
Form1.Canvas.Ellipse(60,60,100,200);
Form1.Canvas.Pen.Color:= clAqua;
//Малювання прямокутника з округленими кутами
Form1.Canvas.RoundRect(245,245,430,430,55,55);
Form1.Canvas.Pen.Color:= clRed;
//Малювання сектора кола
Form1.Canvas.Pie(315,315,415,415,365,365,515,515);
Form1.Canvas.Pen.Color:= clSkyBlue;
//Масив координат вершин п'ятикутника
```

```
T[1].x:=275; t[1].y:=35;
t[2].x:=365; t[2].y:=40;
t[3].x:=425; t[3].y:=75;
t[4].x:=325; t[4].y:=175;
t[5].x:=275; t[5].y:=125;
Form1.Canvas.Polygon (t);
Form1.Canvas.TextOut(35,15,' Ми навчилися малювати різнокольорові фігури!');
end;
initialization
{$I unit1.lrs}
end.
```

Завдання для самостійного виконання.

- Самостійно складіть програму для малювання п'яти кіл, із зсувом від центру кола на крок h=45.
- Самостійно складіть програму для малювання олімпійських кілець.

2.2. Методичні рекомендації щодо використання системи завдань

Управління учнями під час вивчення середовища Lazarus – це керування пізнавальною діяльністю. В умовах навчання розвивається важливий не тільки результат цієї діяльності – знання, уміння, навички учнів, а й формування її структурних компонентів, способів досягнення результатів. Для цього пізнавальна діяльність має бути спеціально організована з урахуванням психологічних процесів.

Важливо відзначити низку принципів організації пізнавальної діяльності на уроці:

- активізація пізнавальних процесів;
- розвиток уміння учня керувати власними пізнавальними процесами;

- розвиток пізнавальних здібностей.

Стан уваги учнів на уроці – найважливіша умова продуктивності пізнавальної діяльності. Необхідно вміти розподіляти увагу між викладеним матеріалом та спостереженням за учнями. Тому теоретичний матеріал, слід видавати невеликими блоками, постійно ілюструючи його демонстрацією роботи у програмі Lazarus. Для закріплення теоретичного матеріалу, учні повинні виконати практичні завдання, які повинні розташовуватися по зростаючій, із ускладненням завдань для складання програм.

Для управління увагою учнів необхідно:

1) вміти визначати у кожний момент уроку ступінь зосередженості окремих учнів та класу в цілому;

2) підтримувати увагу, спираючись на закономірності;

3) усувати ситуативні причини зниження концентрації уваги;

4) формувати увагу учнів.

Увага підлітків переважно довільна, вони здатні зосередитися на нецікавій та важкій роботі заради її майбутнього результату. Однак краще вести урок, використовуючи прийоми, які дозволяють посилити інтерес до предмета, використовувати більше довідкових матеріалів, які будуть допомагати учневі у складанні програми.

Дана система завдань може використовуватися як допоміжний інформаційний матеріал, так і для самостійної роботи на уроці.

Вивчення теми *Візуальне програмування у середовищі Lazarus* починається з основних понять. Після вивчення теми учням запропоновано виконати *Практичну роботу №1*.

Далі можна приступити до вивчення наступної частини теми: *Середовище візуального програмування Lazarus*, *та її основні компоненти, такі як властивості, події, методи*. Необхідно під час обговорення навести приклади властивостей. Після чого учні виконують *Практичну роботу №2*.

Далі можна розпочати вивчення наступної частини теми: *Структура програми у середовищі Lazarus. Проект. Файли проекту*. Після чого виконують *Практичну роботу №3*.

Далі можна приступити до вивчення наступної частини теми: *Використання якості об'єкта Canvas для малювання графіки*, в якій відбувається покрокове виконання *Практичної роботи №4*.

Висновок до другого параграфа

Виходячи з цілей та завдань навчання учнів 9 класів, була розроблена система завдань на тему «Візуальне програмування» середовища Lazarus. Система завдань складається з чотирьох практичних робіт із поступовим поглибленням інформації. Також були розроблені методичні рекомендації щодо використання розробленої системи завдань.

Висновок

Метою дипломної роботи була розробка системи завдань із візуального програмування з використанням візуального середовища програмування Lazarus для учнів 9 класів.

Для досягнення мети було вирішено такі завдання:

- проведено аналіз психолого-педагогічної, науково-методичної, навчально-дидактичної літератури з теми дослідження;
- проведено огляд візуальних середовищ програмування, що використовуються для навчання;
- вивчено методики викладання інформатики з використанням візуальних засобів;
- показано використання візуального середовища під час навчання програмування;
- розроблено систему завдань та методичні рекомендації.

Дана робота складається із двох частин. У першій частині спочатку розглянуто теоретичні питання, пов'язані з цією темою, та вивчено можливості використання візуального програмування під час уроків інформатики.

У другому параграфі розроблено систему завдань із візуального програмування в середовищі Lazarus, так само були розроблені методичні рекомендації щодо використання розробленої системи завдань. Складено план проведення дослідно-експериментальної роботи з використанням розробленої системи завдань. Наведено основні етапи проведення експерименту.

Результат описаної дослідно-експериментальної роботи показує, що система завдань, що використовується, підвищує інтерес учнів 9 класів до предмета інформатики.

Список використаної літератури

1. Алексєєв Є. Р. Самовчитель з програмування на Free Pascal та Lazarus /Є. Р. Алексєєв, О. В. Чеснокова, Т. В. Кучер. – Донецьк : ДонНТУ, Технопарк ДонНТУ УНІТЕХ, 2011. - 503 с.

2. Алексеев Е. Р. Free Pascal и Lazarus : учебник по программированию / Е. Р. Алексеев, О. В. Чеснокова, Т. В. Кучер. – М. : ALT Linux : Изд. дом ДМК-пресс, 2010. – 440 с.

3. . Гуріков С. Р. Програмування в середовищі Lazarus для школярів та студентів : навч. посібник / С. Р. Гуріков. – К. : Форум : НДЦ ІНФР, 2016. – 336 с.

4. Бєлов В. В. Програмування в Delphi : процедурне, об'єктивно орієнтоване, візуальне: Навч. посібник для вузів. / В. В. Бєлов, В. І. Чистякова. - 2-ге вид. - К.: Гаряча лінія - Телеком, 2014. - 240 с.

5. Мансуров К. Т. Основи програмування у середовищі Lazarus /К. Т. Мансуров. - К.: Palmarium Academic Publishing, 2013. - 772 с.

6. Зибиров В. В. Visual Basic 2010 на примерах / В. В. Зибиров. – СПб. : БХВ-Петербург, 2010. – 338 с.

7. Пахомова Б. И. C/C++ и MS Visual C++ 2010 для начинающих / И. Б. Пахомова. – СПб. : БХВ-Петербург, 2011. – 736 с.

8. Семакін І. Н. Основи програмування : підручник / І. М. Семакін, А. П. Шестаков. - К.: Вища школа, 2002. - 432 с.

9. Інформатика : підруч. для 9-го кл. загальноосвіт. Навч. закл. / Й. Я. Ривкінд [та ін.]. – Київ. : Генеза, 2017. – 288 с.: іл.

Додаток

Конспект заняття з візуального програмування

Предмет: Інформатика.

Клас: 9 клас [9].

Тема: *Візуальне середовище програмування Lazarus. Інструменти Lazarus*.

Мета заняття: Ознайомити учнів із середовищем візуального програмування Lazarus та її основними можливостями. Навчити створювати свій перший проект та зберігати його.

Завдання:

- формувати науковий світогляд учнів на тему програмування;
- ознайомитись із основними елементами робочого вікна програми Lazarus;
- познайомити з компонентами середовища Lazarus;
- дати уявлення про загальні принципи візуального програмування;
- підвищити інтерес учнів до процесу програмування в інформатиці;
- показати учням основні прийоми ефективного використання візуального програмування;
- навчити створювати, відкривати та зберігати програму.

Форми організації роботи:

- фронтальне та індивідуальне опитування;
- розповідь, евристична розмова;
- практична робота.

Технічні умови: комп'ютерний клас, проектор, інтерактивна дошка.

Цифрові освітні ресурси: презентація «Візуальне програмування у середовищі Lazarus.ppt».

1) Організаційний частина. 2 хв

2) Оголошення теми і завдань уроку. Мотивація навчальної діяльності учнів. 4 хв

3) Пояснення нового навчального матеріалу. 15 хв

4) Правила виконання вправи. 2 хв

5) Робота на комп’ютері для формування практичних навичок та їх оцінювання. 19 хв

6) Підбиття підсумків заняття. Домашнє завдання. Рефлексія. 3 хв

Хід уроку

1. Організаційна частина

Доброго дня учні. Сідайте.

Сьогодні у нас із вами нова тема, ми починаємо вивчати візуальне програмування середовище Lazarus.

2. Оголошення теми і завдань уроку. Мотивація навчальної діяльності учнів.

На минулому занятті ми з вами розглянули основні поняття візуального програмування. Ознайомились із такими поняттями як об'єкт, клас, компонент, поговорили про візуалізацію програми. Сьогодні ми познайомимося з інструментами та основними компонентами Lazarus. Тема нашого заняття: Візуальне середовище програмування Lazarus. Інструменти Lazarus.

Наша мета: Вивчити основні елементи робочого вікна програми Lazarus. Навчиться створювати відкривати та зберігати програму.

3. Пояснення нового навчального матеріалу.

Lazarus відноситься до вільного програмного забезпечення. Щоб познайомитись з основними інструментами середовища розробки, запустимо середовище програмування. Все, що ви бачите на екрані під час роботи в різних додатках, всі елементи (кнопки, бігунки, меню) можна реалізувати у середовищі Lazarus. У Lazarus використовується технологія візуального програмування. Користувач для створення графічного інтерфейсу програми використовує готові компоненти, значки яких знаходяться на панелі компонентів. Після того, як розміщується компонента на формі, програмний

код для неї генерується автоматично. Вручну залишається запрограмувати лише ті дії, які буде виконувати програма.

Процес створення програми можна розділити на такі етапи:

1. Створення проекту. У результаті екрані з'являється порожня форма.
2. Створення графічного інтерфейсу проекту – розташування необхідних елементів, задання розмірів, зміна властивостей.
3. Написання програмного коду, який визначить, що буде робити програма.
4. Налагодження програми.

Щоб познайомитись з основними інструментами середовища розробки, запустимо середовище програмування.

Для цього потрібно виконати команду:

Пуск => Всі програми => Lazarus => Lazarus.

При цьому запускається оболонка створення програм. На екрані з'явиться набір вікон.

Ми бачимо всі основні інструменти середовища розробки Lazarus:

1. Вікно форми – вікно майбутнього додатка.
2. Головне вікно, що містить три панелі: меню, панель інструментів, палітру компонентів. Палітру компонентів будемо використовувати для створення інтерфейсу користувача.
3. Вікно Інспектор об'єктів, що містить файли проекту та вікно з вкладкою Властивості, в якій налаштовуються властивості розміщених на форму об'єктів.
4. Вікно Редактор вихідного коду, в якому прописується програмний код.

Дамо вікнам, що з'явилися, коротку характеристику.

Головне вікно. Тут розташовуються меню, панель інструментів та палітра компонентів. На Палітрі компонентів, є багато тематичних вкладок, на яких розташовуються візуальні та не візуальні компоненти для майбутньої програми. Не візуальні компоненти видно лише на першому етапі створення програми – під час редагування. Головне вікно залишається відкритим весь час

під час роботи програми. Закриваючи його, ви тим самим закриваєте Lazarus і всі відкриті в ньому вікна.

4) Правила виконання вправи. Інструктаж.

Ознайомтеся з правилами техніки безпеки при роботі на комп’ютері.

5. Робота на комп’ютері для формування практичних навичок та їх оцінювання.

Усі сідаємо за комп'ютери. Знаходимо програму на головному вікні, запускаємо її. Після запуску програми ми бачимо вікно форми, головне вікно, Вікно Інспектор об'єктів, вікно редактора коду та вікно повідомлень. Файл – Зберегти як… – Даємо назву програмі, після даємо назву коду програми. Якщо необхідно відкрити існуючий документ виконуємо команди Проект – Відкрити проект. З'явиться діалогове вікно, у якому вибираємо потрібну програму та відкриваємо його подвійним клацанням мишки або натисканням кнопки Відкрити, наша програма відкрита. Якщо Ви відредагували файл і хочете зберегти його під іншим ім'ям, не змінюючи старий файл, виконуємо команду Файл - Зберегти як ..., і знову з'явиться діалогове вікно, в якому вказуємо папку та ім'я файлу.

Створіть новий проект – форму, додавши на неї дві кнопки Button. Натисніть поле Caption і змініть назву Form1 на Форма. Змініть властивості форми та властивості кнопок за таблицею:

Властивість Форми	Значення
Color	clHighlight
Width	300
Height	250
Властивість Button1	Значення
Caption	Колір
Font	Times New Roman – 14 – звичайний
Top	152
Left	40

Height	44
Width	80
Властивість Button2	Значення
Caption	Розмір
Top	150
Left	176
Font	Times New Roman – 14 – звичайний
Height	45
Width	80

Збережіть проект у папці: Перший проект. Виділіть кнопку Button1, у вікні Інспектора об'єктів, відкрийте вкладку Події, знайдіть подію OnClik і подвійним клацанням по порожньому полю підключіть подію за кнопкою - клацання мишею, у вас у вікні редактора має з'явитися шаблон процедури події кнопки Button1. У вікні редактора коду у процедурі напишіть код програми:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
Form1.Color:=clPurple;
end;
```

Далі підключіть процедуру події натисканням кнопки Button2 і у процедурі напишіть код програми:

```
procedure TForm1.Button2Click(Sender: TObject);
begin
Form1.Width: = 156;
end;
```

Збережіть проект. Після цього запустіть програму. Натисніть на кнопку колір, а потім на кнопку розмір. Що змінилося у програмі?

6. Підбиття підсумків заняття. Домашнє завдання. Рефлексія.

1. Вам було цікаво на занятті?

2. Ви дізналися щось нове на занятті?

3. Чи був доступний матеріал, що вивчався?

4. Ви його зрозуміли?

Домашнє завдання повторити компоненти вікон та їх властивості середовища Lazarus.