Індивідуальне завдання

Виконання індивідуального завдання з дисципліни “Об’єктно-орієнтоване програмування” дозволяє студентам набути практичних навичок в аналізі та формалізації задачі по створенню класів програмних об’єктів, розробці, тестуванню та налагодженню програм в середовищі візуального програмування Delphi, сприяє закріпленню та поглибленню студентами теоретичних знань по основним розділам дисципліні, що дозволить студентам підготуватись до успішного виконання курсової роботи.

Теми індивідуальних завдань охоплюють два розділи дисципліни “Об’єктно-орієнтоване програмування” : “Об’єктна модель Delphi” та “Властивості класів”.

Порядок виконання роботи

Протягом семестру студентом виконується 1 індивідуальне завдання, на виконання якого учбовим планом відведено 8 тижнів.

Варіант індивідуального завдання обирається студентом зі списку варіантів, що наведений нижче у методичних вказівках. Номер варіанта індивідуального завдання відповідає номеру студента у списку академічної групи.

Порядок виконання індивідуального завдання:

1. Розробка класу програмного об’єкту.
2. Розробка програмного інтерфейсу користувача.
3. Створення екземпляру класу та його використання.
4. Оформлення звіту з індивідуального завдання.

Розглянемо виконання кожного етапу на конкретному прикладі.

1. Розробка класу програмного об’єкта

Завдання: *Визначити клас для цілочисельної змінної. Поля класу: значення змінної, ідентифікатор змінної. Методи: зведення значення змінної у квадрат.*

Розпочнемо з аналізу полів та методів майбутнього класу змінної TVariable.

Перше поле – призначене для зберігання цілочисельного значення змінної, тому його можна описати типом integer:

Value : integer;

В другому полі зберігатиметься назва змінної. Згідно з правилами мови програмування Паскаль назва змінної – це послідовність символів, яка не може починатися з цифри та включати спеціальні символи. Довжина ідентифікатора не може перевищувати 63 символи, тому для опису поля обирається тип string [63].

Name : string [63];

Згідно з завданням для класу необхідно створити метод, який зводить значення змінної у квадрат.

procedure Square;

Крім цього необхідно описати конструктор, в якому буде проводитись ініціалізація полів класу:

constructor Create (V:integer; N:string);

Для відображення на екрані значень полів класу можна створити метод ShowFields, параметрами якого будуть дві мітки L1, L2:TLabel. Перша L1 буде використовуватися для відображення значення поля Name, друга L2 – поля Value. Надалі при виклику цього метода, йому будуть передаватися імена двох міток головної форми, у яких будуть виводитися ці значення.

Опис класу набуває такий вигляд:

```
Type
TVariable = class
   Value: integer;
   Name:string [63];
   constructor Create (V:integer; N:Name);
   procedure Square;
   procedure ShowFields (L1, L2:TLabel);
end;
```

Бажано при виконанні завдання поля класу описати приватними та зробити властивості для доступу до них. Тоді опис класу буде:

```
Type
TVariable = class
private
   FValue: integer;
   FName:string [63];
protected
procedure SetValue (V:integer);
procedure SetName (N:string);
function GetName :string;
function GetValue :integer;
public
   property Value:integer read GetValue write SetValue;
   property Name:string read GetName write SetName;
   constructor Create (V:integer; N:Name);
   procedure Square;
   procedure ShowFields (L1, L2:TLabel);
end;
```

Опис класу бажано проводити в окремому програмному модулі. Для того щоб додати до проекту новий програмний модуль в середовищі Delphi необхідно в меню обрати File / New… і у вікні (рис.1) вибрати піктограму Unit.

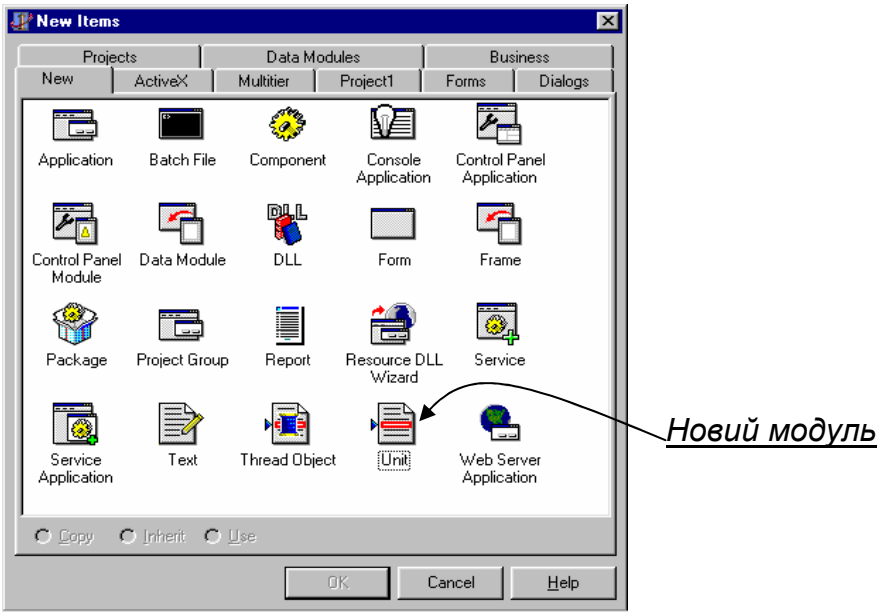

Рисунок 1. Вікно відкриття нової складової проекту

Після того як в розділі ***interface*** нового модуля буде описаний клас можна скористатися засобом **Class Completion** (завершення класу), що викликається при натисненні Ctrl+Shift+C. Якщо попередній програмний код був набраний без помилок, то Delphi автоматично генерує заготовки для всіх описаних процедур та функцій. Опис класу завершується після опису вмісту методів класу. Остаточний вигляд класу можна подивитися в додатку 1.

Зверніть увагу на те, що в методі запису SetName проводиться перевірка на правильність формування ідентифікатора, а у методі Square при обчисленні квадрату результат перевіряється на переповнення.

2. Розробка програмного інтерфейсу користувача

Після того як буде завершений опис класу і виправлені всі помилки необхідно продумати, які інтерфейсні елементи будуть використанні на формі для:

- відображення поточного значення полів класу;
- введення (редагування) значення полів;
- виклику методів класу для демонстрації їх роботи.

В класі TVariable два поля, для відображення їх значення можна використати дві мітки (Label), для введення значень цих полів – два компонента Edit.

Конструктор класу можна буде викликати при створенні форми з обробника OnCreate, тому спеціальні інтерфейсні елементи для його виклику не потрібні.

Метод ShowFields буде необхідно викликати кожен раз, коли відбувається зміна полів.

Для виклику метода Square можна використати просту кнопку Button, задавши її обробник OnClick.

Вибір того чи іншого з інтерфейсних елементів для відображення “начинки” класу не обмежується, треба тільки не порушувати умов функціонального призначення компонентів.

В результаті розробки програмного інтерфейсу головне вікно набуває вигляд, що поданий на рис.4.

3. Створення екземпляру класу та його використання

Для того щоб створити екземпляр класу TVariable (або об’єкт) необхідно:

1. під’єднати модуль, де описаний клас до модуля головної форми, додавши його ім’я до рядка uses;
2. в розділі var модуля головної форми оголосити екземпляр класу TVariable:

```
Var
    First:TVariable; //об’єкт класу TVariable з ім’ям First
```

3. в обробнику OnCreate форми викликати конструктор класу TVariable, задавши якісь початкові значення полів, та відобразити вміст нового класу на формі, викликавши метод ShowFields :

```
First:=TVariable.Create(0,'X');
First.ShowFields(Label1, Label2); //Label1, Label2 мітки на Form1
```

4. задати для інтерфейсних елементів на Form1 всі необхідні обробники подій згідно їх призначення.

Остаточний варіант тексту програми можна переглянути у додатку 2.

Індивідуальне завдання вважається закінченим, якщо отриманий результат по всіх тестових прикладах роботи програми, оформлений звіт згідно з вимогами, що наведені в методичних вказівках. Під час здачі індивідуального завдання студент вільно орієнтується в теоретичному матеріалі, правильно відповідає на запропоновані запитання.

Кожного тижня студент доповідає викладачу про хід виконання роботи.

4. Оформлення звіту з індивідуального завдання

При оформленні звіту необхідно дотримуватися державних стандартів, що висуваються до оформлення технічної документації:

- звіт оформлюється на листах білого паперу формату А4;
- текст розташовується на листі, додержуючись таких розмірів полів: ліве – 2.5 см, праве – 2 см, верхнє та нижнє – 2.5 см;
- міжрядковий інтервал – 1.5;
- розмір шрифту (за умови машинного набору) – 14 пт;
- форматування абзаців – по ширині з відступом 1,27;
- розділи звіту (пункти та підпункти) повинні мати заголовки. Після заголовку крапка не ставиться, заголовки не підкреслюються. Після заголовку перед текстом пропускається один рядок.
- ілюстрації (рисунки, діаграми, схеми алгоритмів, роздрук екранних форм

проекту) розміщуються безпосередньо після тексту, де він вперше згадується або за браком місця на наступній сторінці. На всі ілюстрації мають бути посилання у тексті.

Ілюстрація позначається словом “Рисунок <номер> ”, яке разом з назвою ілюстрації розміщують безпосередньо після неї. Наприклад, “Рисунок 2. Структурна схема програмних модулів” ;

- таблицю розташовують безпосередньо після тексту, де є посилання на неї, або на наступній сторінці. Над таблицею розташовується заголовок всієї таблиці у формі:

Таблиця < номер > – < назва таблиці >

Якщо таблиця розривається, то над частинами таблиці пишуть – “Продовження таблиці – < номер > ” .

- Формули та рівняння розташовуються безпосередньо після тексту, де є посилання на них. Вище нижче кожного рівняння або формули залишається по одному вільному рядку. Формули і рівняння розташовуються по центру рядка та нумеруються. Номер ставиться з правого боку сторінки у дужках ().

Для детального ознайомлення з вимогами до оформлення звітів дивіться [5] або стандарти “Єдиної системи конструкторської документації” та “Єдиної системи програмної документації”.

Звіт з індивідуального завдання включає:

1. Титульний лист (див. додаток 4).
2. Зміст.
3. Завдання до роботи.
4. Постановку задачі - опис класу (рисунок з діаграмою класу (класів)):

- опис полів класу – призначення кожного поля, тип поля, обмеження на характер та розмірність інформації, що обробляється;
- опис методів класу – призначення методу, опис вхідних параметрів та результатів роботи, математичну постановку розрахунків, опис алгоритму роботи (схема алгоритму).
- опис властивостей для доступу до захищених полів класу.

Діаграма класів будується за UML-нотацією або нотацією Г. Буча і може бути подана так як показано на рис.2.

В залежності від степені деталізації діаграми, опис поля може включати ім’я поля, тип і значення, що присвоюється за замовчанням, а також ознаку видимості:

< ознака видимості > < ім’я > : < тип > = < значення >

При описі методів вказують:

< ознака видимості > < ім’я > (< список параметрів >) : < тип результату >

Для полів та методів признак видимості може приймати такі значення:

- + загальнодоступний;
- # захищений;
- – приватний.

Якщо спадкування проводилося від класу з бібліотеки візуальних компонентів, то

при описі полів і методів слід вказувати тільки ті, що використовуються або перекриваються в дочірньому класі.

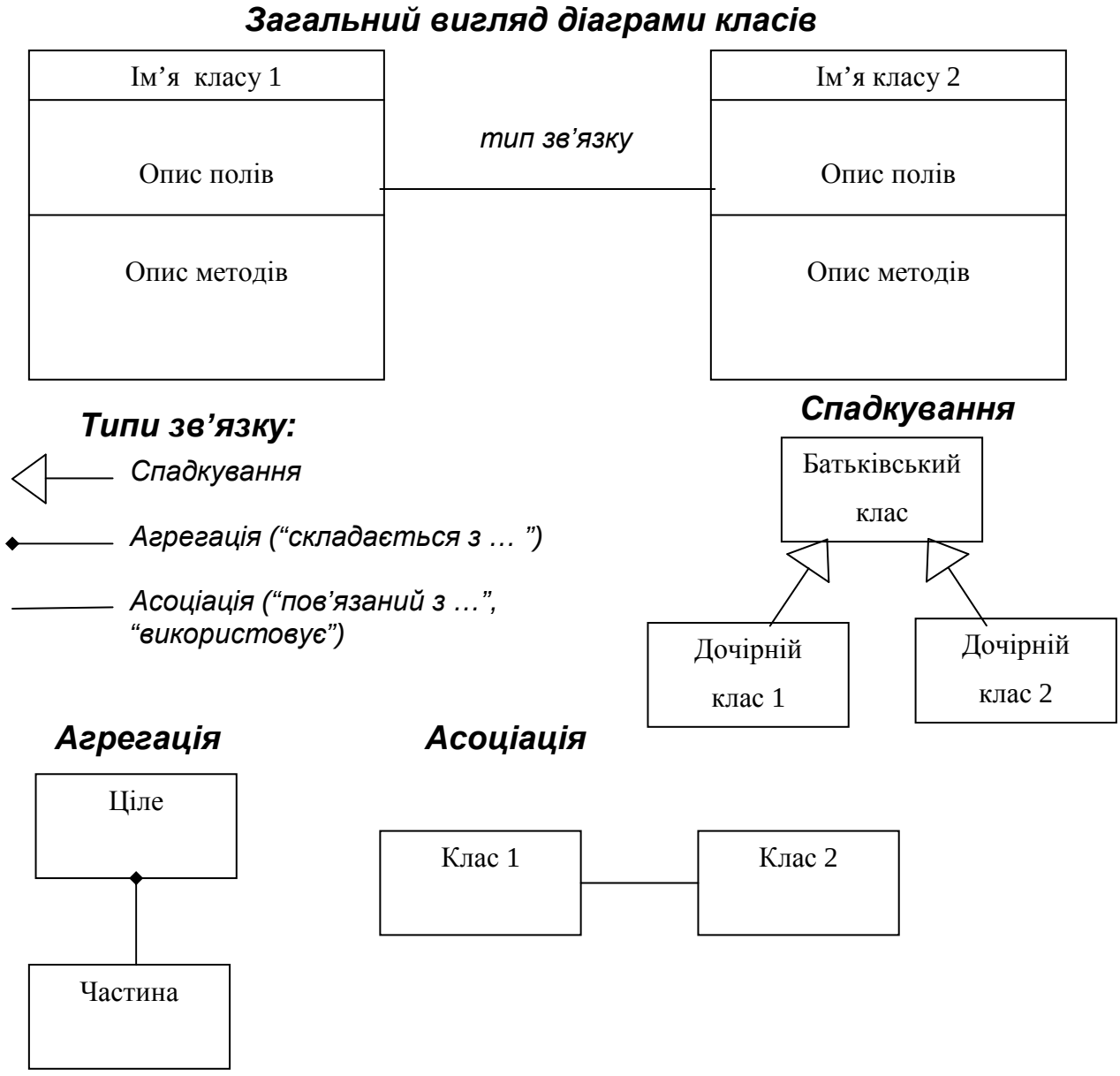

Рисунок 2. Діаграма класів

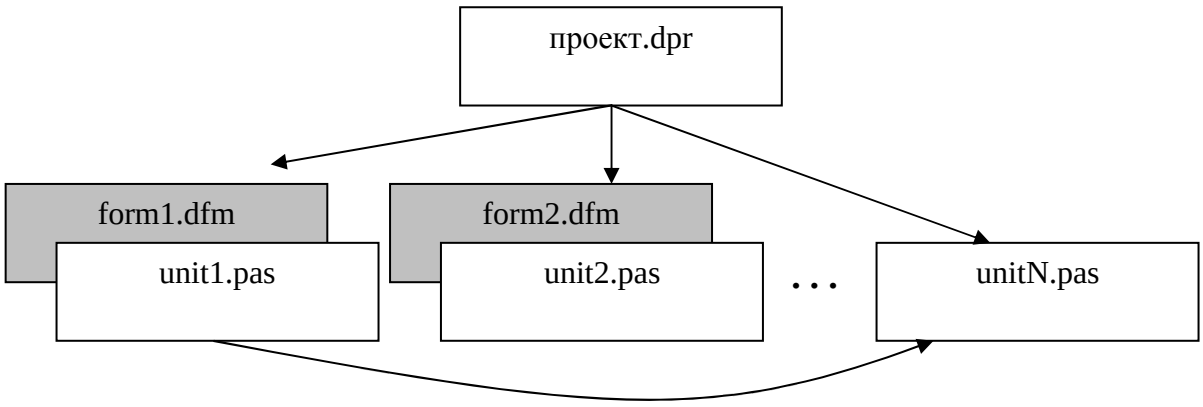

Рисунок 3. Структурна схема програмних модулів

5. Опис структури проекту, до складу якого входять головний файл проекту з розширенням dpr, форми та модулі форм, модуль з описом класів. Таким чином проект складається з N модулів та M форм. Загальна структурна схема проект подана на рис. 3. Якщо з одного модуля є посилання на інший, то цей зв'язок вказується на рисунку (unit1 - unitN).

Для кожного модуля описується його призначення, специфікації основних методів та обробників подій, задекларованих структур даних та основних змінних.

6. Опис програмного інтерфейсу.

Для кожної форми проекту наводиться роздрук вікна у run-time, на якому вказуються імена елементі управління (рис. 4). Якщо не має можливості роздрукувати вікна, то рисунки виконуються схематично вручну.

Для кожного елемента управління наводиться опис призначення та використані обробники подій. Наприклад для форми, що наведена на рис. 4:

GroupBox1 – об'єднує інтерфейсні елементи, в яких виводяться значення полів об'єкту;

Label1 – виведення значення поля Name класу;

Label2 – виведення значення поля Value класу;

Button1 – обчислення квадрату значення поля Value (обробник OnClick);

і так далі про всі елементи.

7. Роздрук текстів програмних модулів наводиться у додатках до пояснювальної записки. Додаткам передує лист, на якому по центру розміщується заголовок

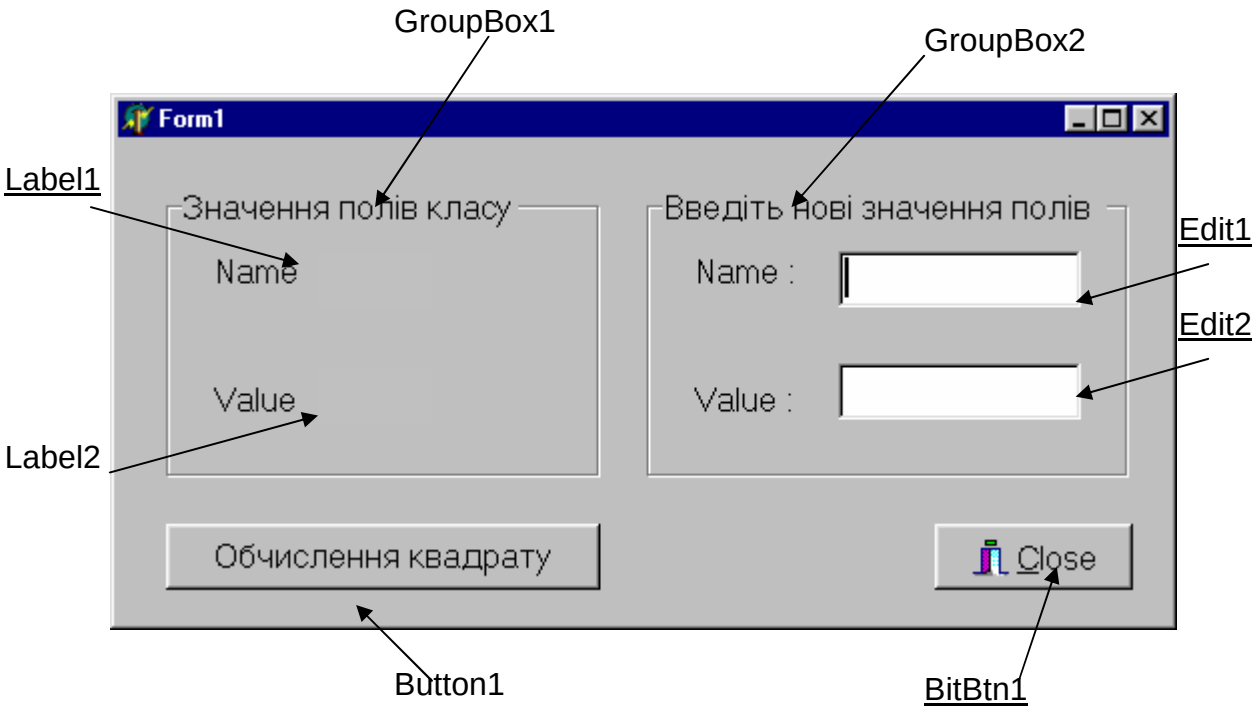

Рисунок 4. Головне вікно програми обчислення квадрату значення змінної (Form1).

“ДОДАТКИ”. Всі додатки нумеруються та супроводжуються заголовками виду: “Додаток 1. Роздрук тексту програмного модуля Unit1.pas” .

8. Список використаної літератури.
9. Додатки.

Варіанти індивідуального завдання

ПРИМІТКА для відмінників!

Для класів чисел (варіанти №1-4) як батьківський використати клас TVariable, що описаний вище.

Для класів плоских геометричних фігур за базовий батьківський клас або як складову класу взяти клас точки. Для об’ємних фігур за базовий взяти клас плоскої фігури, що лежить в її основі.

Для класів шахових фігур визначити один загальний клас для будь-якої шахової фігури, в якому метод переміщення буде віртуальним і перекриватиметься в нащадках. Для відображення шахової фігури на формі можна створити простеньке графічне зображення (наприклад, в ImageEditor або Paint), а потім вивести його на форму в компоненті Image (палітри Additional).

Для класів лічильників можливо успадковувати свій клас від компоненту таймер Timer, додавши до нього відповідні поля та методи.

1. Визначити клас для комплексних чисел. Поля класу - дійсна та комплексна частини числа. Методи класу - додавання, віднімання, добуток, комплексних чисел.
2. Визначити клас для дійсних чисел. Поля класу - ціла та дробова частини числа. Методи класу - додавання, віднімання, добуток, ділення дійсних чисел.
3. Визначити клас для десяткових дробів. Поля класу - чисельник, знаменник та ціла частина дробі. Методи класу - додавання, віднімання, добуток, ділення, нормалізація дробі.
4. Визначити клас для числа в заданому степені. Поля класу – значення числа, значення показника степені. Методи класу – зведення до степені, добуток та ділення степенів.
5. Визначити клас для відрізка. Поля класу – координати початку та кінця відрізка. Методи – визначення довжини та координат центру.
6. Визначити клас для прямокутника. Поля класу – координати лівого верхнього та правого нижнього кутів. Методи – визначення периметру та площі.
7. Визначити клас для трикутника. Поля класу – координати вершин. Методи – визначення довжини сторін, периметру та площі.
8. Визначити клас для кола. Поля класу – координати центра та радіус. Методи – визначення довжини та площі.
9. Визначити клас для квадратного рівняння. Поля класу – коефіцієнти рівняння. Методи – визначення коренів.
10. Визначити клас для кулі. Поля класу – координати центра та радіус. Методи – визначення площі поверхні та об’єму.

11. Визначити клас для конуса. Поля класу – радіус основи та висота. Методи – визначення площі поверхні та об’єму.
12. Визначити клас для прямокутного паралелепіпеда. Поля класу – довжини ребер. Методи – визначення площі поверхні та об’єму.
13. Визначити клас для циліндра. Поля класу – радіус основи та висота. Методи – визначення площі поверхні та об’єму.
14. Визначити клас для шахової фігури - ферзь. Властивості: колір, координати на дошці, ім’я файлу з графічним зображенням. Методи: переміщення по дошці за встановленими правилами та відображення на формі однієї шахової фігури.
15. Визначити клас для шахової фігури - король. Властивості: колір, координати на дошці, ім’я файлу з графічним зображенням. Методи : переміщення по дошці за встановленими правилами та відображення на формі однієї шахової фігури.
16. Визначити клас для шахової фігури - слон. Властивості: колір, координати на дошці, ім’я файлу з графічним зображенням. Методи : переміщення по дошці за встановленими правилами та відображення на формі однієї шахової фігури.
17. Визначити клас для шахової фігури - кінь. Властивості : колір, координати на дошці, ім’я файлу з графічним зображенням. Методи : переміщення по дошці за встановленими правилами та відображення на формі однієї шахової фігури.
18. Визначити клас для шахової фігури - ладдя. Властивості : колір, координати на дошці, ім’я файлу з графічним зображенням. Методи : переміщення по дошці за встановленими правилами та відображення на формі однієї шахової фігури.
19. Визначити клас для шахової фігури - пішка. Властивості : колір, координати на дошці, ім’я файлу з графічним зображенням. Методи : переміщення по дошці за встановленими правилами та відображення на формі однієї шахової фігури.
20. Визначити клас для цифрового лічильника - змінної з обмеженим діапазоном, яка обнуляється, коли її цілочисельне значення дорівнює визначеного максимуму. Властивості: поточне, мінімальне та максимальне значення. Методи : встановлення максимального та мінімального значення лічильника, збільшення значення на одиницю.
21. Визначити клас для електронного годинника. Властивості: поточне, мінімальне та максимальне значення, встановлений час будильника. Методи : встановлення поточного та мінімального значення часу, збільшення значення, встановлення будильника.
22. Визначити клас для лічильника електроенергії. Властивості: поточне, мінімальне та максимальне значення. Методи : встановлення максимального та мінімального значення лічильника, збільшення значення на одиницю, зміна швидкості зміни показників лічильника в залежності від зміни напруги (кількості увімкнутих ламп).
23. Визначити клас для спідометра автомобіля. Властивості: поточне, мінімальне та максимальне значення, поріг швидкості, після перевищення якого дається повідомлення про небезпеку. Методи : встановлення мінімального значення на

спідометрі, збільшення чи зменшення значення в залежності від швидкості автомобіля.

24. Визначити клас для одновимірного масиву. Поля класу – поточна кількість елементів, та поле-масив елементів. Методи - визначення середнього арифметичного елементів масиву, виведення елементів на екран, пошук найбільшого та найменшого елемента.
25. Визначити клас для рядка символів, що “знає” свою довжину та кількість слів. Поля класу: ім’я рядка, вміст рядка.