Міністерство освіти і науки України

Сумський державний педагогічний університет

імені А.С.Макаренка

В.М. Базурін

О.С.Чашечникова

Об´єктно-орієнтоване програмування на мові C#

Поглиблений рівень

Лабораторний практикум

Для студентів спеціальностей 15.01 Професійна освіта (Будівництво та зварювання), 15.17 Професійна освіта (Технологія виробів легкої промисловості)

Суми 2022

Б17

Рекомендовано до друку Вченою радою Сумського державного педагогічного університету імені А.С.Макаренка (протокол №2 від 27.09.2021 р.)

Рецензенти:

Семеніхіна О.В. – доктор педагогічних наук, професор, завідувач кафедри інформатики Сумського державного педагогічного університету імені А.С.Макаренка

Рябко А.В. – кандидат педагогічних наук, старший викладач кафедри фізико-математичної та інформатичної освіти та комп´ютерних технологій Глухівського національного педагогічного університету імені Олександра Довженка

Б17 Базурін В.М., Чашечникова О.С. Об´єктно-орієнтоване програмування на мові C#. Поглиблений рівень. Лабораторний практикум. Суми, 2021. 81 с.

У посібнику розкрито додаткові питання об´єктно-орієнтованого програмування, які застосовуються у процесі розробки програм на мові C#: переведення чисел з однієї системи числення в іншу, статичні класи, успадкування класів, створення і перевантаження методів. Для студентів вищих педагогічних навчальних закладів

Зміст

Вступ

Шановні студенти! Вашій увазі пропонується посібник, який є логічним продовженням посібника «Об´єктно-орієнтоване програмування на мові C#». Посібник налічує 6 лабораторних робіт, які мають розкривають питання, що мають важливе практичне значення: перетворення чисел з однієї системи числення в іншу, статичні класи, статичні методи, перевантаження методів, успадкування класів, створення програм, які містять вкладки і рядок статусу, таймер. Тематика лабораторних робіт пов´язана з поглибленим вивченням об´єктно-орієнтованого програмування на мові C#.

У процесі виконання завдань у студентів формуються такі компетентності:

- навички використання інформаційних і комунікаційних технологій;
- здатність використовувати сучасні інформаційні технології та спеціалізоване програмне забезпечення та інтегрувати їх в освітнє середовище;
- здатність використовувати відповідне програмне забезпечення для вирішення професійних завдань у сфері будівництва та зварювання;
- здатність використовувати у професійній діяльності основні положення, методи, принципи фундаментальних та прикладних наук.

Даний посібник призначений для студентів, які вивчають програмування на мові C#. Посібник доцільно використати під час вивчення таких дисциплін: «Інформатика», «Основи програмування з практикумом», «Програмування інженерних розрахунків», «Практикум з програмування проектних розрахунків».

Лабораторна робота №13

Тема: Перетворення чисел з однієї системи числення в іншу.

Мета: набути навичок розробки програм для перетворення даних з однієї системи числення в іншу.

Короткі теоретичні відомості

Число, подане в різних системах числення, зчитується / виводиться у рядковому форматі (string). Далі цей рядок необхідно перетворити в числовий формат (у відповідній системі числення). Загальний порядок перетворення чисел такий:

int iVar=Convert.ToInt32 (sss, From);

де sss – рядкова змінна;

From – основа системи числення, в якій подано число.

Для переведення числа з рядкового формату в десяткове число використовується команда:

int iVar=Convert.ToInt32 (sss);

Для переведення числа з рядкового формату в двійкове число використовується команда:

int iVar=Convert.ToInt32 (sss, 2);

Для перетворення числа з числового формату в рядковий існує команда:

string sss=Convert.ToString(dd, To);

де dd – запис числа у числовому форматі,

To – основа системи числення, у яку необхідно перетворити число.

При цьому слід пам’ятати, що число відображається як рядок.

Наприклад, для перетворення десяткового числа в рядок (у двійковій системі числення) слід використати команду:

string sss=Convert.ToString (x, 2);

Аналогічно, для перетворення числа з десяткової системи в вісімкову служить команда:

string sss=Convert.ToString (x, 8);

Але слід звернути увагу на те, що вісімкове число sss має рядковий формат даних.

Приклад програми

Наведемо приклад програми, яка перетворює число з десяткової системи в двійкову. Зовнішній вигляд програми наведено на рис.13.1.

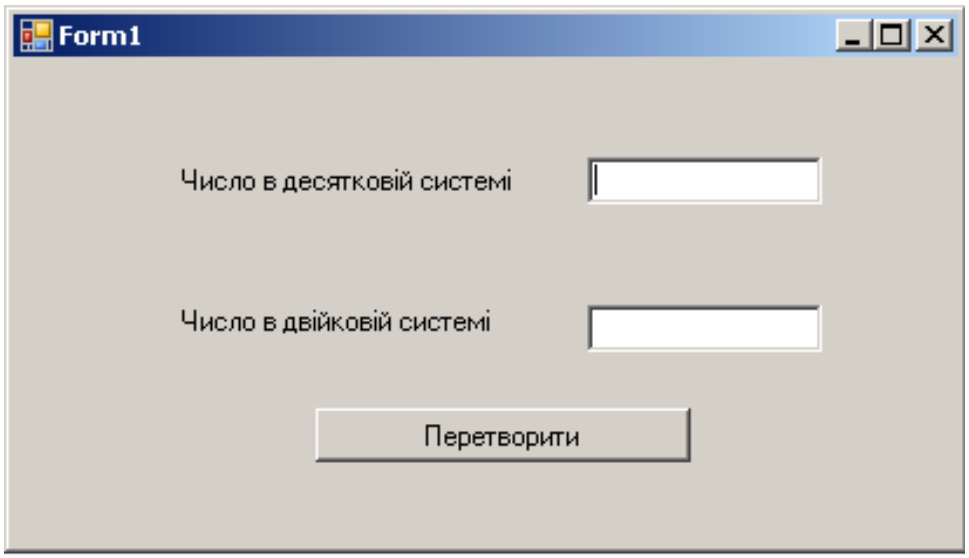

Рис.13.1. Зовнішній вигляд екранної форми

Код підпрограми, що перетворює число в двійкову систему:

```
private void button1_Click(object sender, EventArgs e)          {
        string sss = textBox1.Text;
        int x = Convert.ToInt32(sss);
        sss = Convert.ToString(x, 2);
        textBox2.Text = sss;
    }
```

Дана програма перетворює число в двійкову систему. Якщо в рядку

sss = Convert.ToString(x, 2);

замінити параметр 2 на 8, то програма буде перетворювати число у вісімкову систему.

Розглянемо приклад програми для перетворення чисел з двійкової системи в десяткову. Зовнішній вигляд екранної форми наведено на рис.13.2.

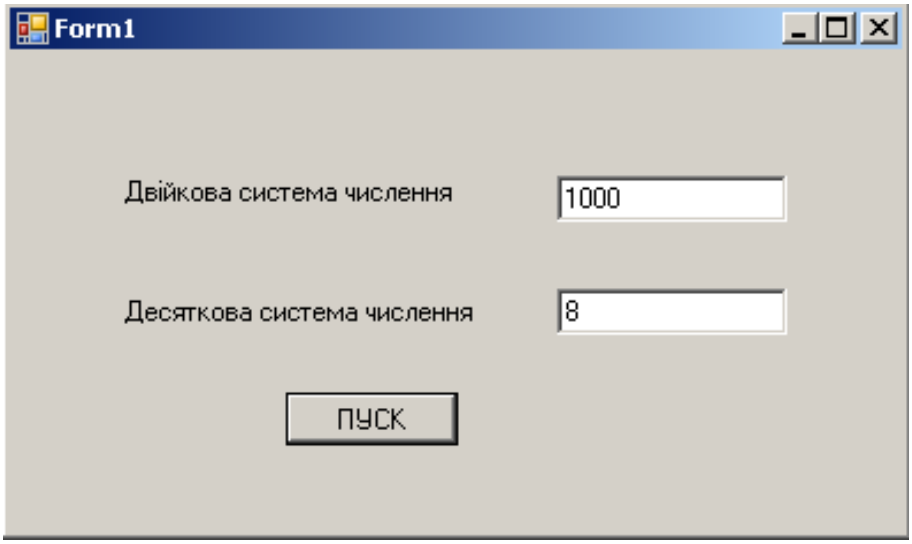

Рис.13.2. Зовнішній вигляд форми

Код програми

```
private void button1_Click(object sender, EventArgs e)
      {
         string sss = textBox1.Text;
         int x = Convert.ToInt32(sss, 2);
         sss = Convert.ToString(x);
         textBox2.Text = sss;
      }
```

Наведемо приклад консольноъ програми, яка перетворює масив цілих чисел у шістнадцяткові:

```
using System;

namespace lab13v24t2
{
   class Program
   {
      public static void Main(string[] args)
      {
         Console.WriteLine("Hello World!");
         int i=0;
         int[] masiv=new int[7];
         for(i=0;i<7;i++) {
            string sss=Console.ReadLine();
            masiv[i]=Convert.ToInt32(sss);
         }
         string []ss16=new string[7];
         for(i=0;i<7;i++) {
            ss16[i]=Convert.ToString(masiv[i],16);
         }
         Console.WriteLine("**********************");
         for(i=0;i<7;i++) {
            Console.WriteLine(ss16[i]);
         }

         // TODO: Implement Functionality Here

         Console.Write("Press any key to continue . . . ");
         Console.ReadKey(true);
      }
```

```
    }
}
```

Постановка завдання

1. Написати програму для перетворення чисел між різними системами числення.
2. Написати програму для перетворення складних структур даних з однієї системи числення в іншу.

Варіанти завдань

Варіант 1

1. Написати програму для перетворення чисел з двійкової системи числення в шістнадцяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в двійкову. Кількість елементів масиву 6, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 2

1. Написати програму для перетворення чисел з двійкової системи числення в вісімкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в трійкову. Кількість елементів масиву 4, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 3

1. Написати програму для перетворення чисел з двійкової системи числення у трійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в вісімкову. Кількість елементів масиву 9, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 4

1. Написати програму для перетворення чисел з двійкової системи числення в десяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в шістнадцяткову. Кількість елементів масиву 3, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 5

1. Написати програму для перетворення чисел з вісімкової системи числення в двійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в двійкову. Кількість елементів масиву 7, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 6

1. Написати програму для перетворення чисел з вісімкової системи числення в десяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в трійкову. Кількість елементів масиву 4, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 7

1. Написати програму для перетворення чисел з вісімкової системи числення в шістнадцяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в вісімкову. Кількість елементів масиву 5, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 8

1. Написати програму для перетворення чисел з десяткової системи числення в двійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в шістнадцяткову. Кількість елементів масиву 3, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 9

1. Написати програму для перетворення чисел з десяткової системи числення в трійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в двійкову. Кількість елементів масиву 7, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 10

1. Написати програму для перетворення чисел з десяткової системи числення в вісімкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в трійкову. Кількість елементів масиву 4, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 11

1. Написати програму для перетворення чисел з десяткової системи числення в шістнадцяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в вісімкову. Кількість елементів масиву 8, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 12

1. Написати програму для перетворення чисел з шістнадцяткової системи числення в двійкову.

2. Написати програму для перетворення масиву цілих чисел з десяткової системи в шістнадцяткову. Кількість елементів масиву 3, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 13

1. Написати програму для перетворення чисел з шістнадцяткової системи числення у трійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в двійкову. Кількість елементів масиву 4, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 14

1. Написати програму для перетворення чисел з шістнадцяткової системи числення в вісімкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в трійкову. Кількість елементів масиву 5, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 15

1. Написати програму для перетворення чисел з шістнадцяткової системи числення в десяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в вісімкову. Кількість елементів масиву 6, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 16

1. Написати програму для перетворення чисел з двійкової системи числення в десяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в шістнадцяткову. Кількість елементів масиву 7, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 17

1. Написати програму для перетворення чисел з двійкової системи числення у трійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в двійкову. Кількість елементів масиву 6, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 18

1. Написати програму для перетворення чисел з двійкової системи числення у вісімкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в трійкову. Кількість елементів масиву 5, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 19

1. Написати програму для перетворення чисел з двійкової системи числення в шістнадцяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в вісімкову. Кількість елементів масиву 4, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 20

1. Написати програму для перетворення чисел з вісімкової системи числення в двійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в шістнадцяткову. Кількість елементів масиву 3, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 21

1. Написати програму для перетворення чисел з вісімкової системи числення у трійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в двійкову. Кількість елементів масиву 4, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 22

1. Написати програму для перетворення чисел з вісімкової системи числення в шістнадцяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в трійкову. Кількість елементів масиву 5, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 23

1. Написати програму для перетворення чисел з вісімкової системи числення в десяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в вісімкову. Кількість елементів масиву 6, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 24

1. Написати програму для перетворення чисел з шістнадцяткової системи числення в двійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в шістнадцяткову. Кількість елементів масиву 7, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 25

1. Написати програму для перетворення чисел з шістнадцяткової системи числення в трійкову.

2. Написати програму для перетворення масиву цілих чисел з десяткової системи в двійкову. Кількість елементів масиву 8, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 26

1. Написати програму для перетворення чисел з шістнадцяткової системи числення в вісімкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в трійкову. Кількість елементів масиву 7, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 27

1. Написати програму для перетворення чисел з шістнадцяткової системи числення в десяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в вісімкову. Кількість елементів масиву 6, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 28

1. Написати програму для перетворення чисел з вісімкової системи числення в двійкову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в шістнадцяткову. Кількість елементів масиву 5, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 29

1. Написати програму для перетворення чисел з вісімкової системи числення в десяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в двійкову. Кількість елементів масиву 4, їх введення здійснюється за допомогою елемента управління DataGridView.

Варіант 30

1. Написати програму для перетворення чисел з вісімкової системи числення в шістнадцяткову.
2. Написати програму для перетворення масиву цілих чисел з десяткової системи в трійкову. Кількість елементів масиву 3, їх введення здійснюється за допомогою елемента управління DataGridView.

Контрольні питання

1. Що таке система числення?
2. Яка система числення називається позиційною?
3. Яка система числення називається непозиційною?
4. Наведіть приклади позиційних систем числення.

5. Які методи служать для перетворення даних між різними форматами в C#?
6. Який алгоритм дій під час перетворення даних з десяткової системи числення у двійкову?
7. До якого виду – позиційного чи непозиційного – належить десяткова система?
8. До якого виду належить двійкова система?

Лабораторна робота №14

Тема: Розробка програм з використанням вкладок
Мета: набути навичок створення прикладних програм з використанням елемента управління TabControl.
Обладнання:

Короткі теоретичні відомості

Елемент управління TabControl являє собою аналог багатосторінкової книги. На кожній вкладці цієї книги можна розмістити інші елементи управління. Кожна вкладка є об'єктом типу TabPage. Для того, щоб додати нові вкладки, слід використати властивість TabPages, яка описує колекцію сторінок.

Для додання на форму елемента TabControl необхідно задіяти простір імен System.Windows.Forms і динамічну бібліотеку System.Windows.Forms.dll.

Таблиця 14.1

Конструктори

Назва	Опис
TabControl()	Ініціалізує новий екземпляр класу TabControl

Таблиця 14.2

Деякі властивості TabControl

Властивість	Тип	Характеристика
SelectedIndex	int	Індекс вибраної сторінки вкладки. Відлік починається з нуля. Якщо сторінку не вибрано, то властивість дорівнює -1
SelectedTab	int	Повертає або задає поточну вибрану сторінку вкладки
Bottom	піксель	Повертає відстань між нижньою межею елемента управління і верхньою межею клієнтської області контейнера
Bounds	піксель	Задає або повертає розмір і положення поточного елемента управління відносно батьківського

CanFocus	bool	Вказує на те, чи може елемент отримати фокус
CanSelect	bool	Вказує на те, чи може елемент бути вибраним
ClientRectangle		Повертає прямокутник, який являє собою клієнтську площу закладки
ClientSize		Задає або повертає розміри клієнтської області закладки
ContextMenu		Повертає або задає контекстне меню для закладки
Created	bool	Повертає значення, яке вказує, чи був створений елемент із закладками
Enabled	bool	Повертає або задає інтерактивність закладок
Font		Повертає або задає шрифт текста, який знаходиться на закладках
FontHeight	піксель	Задає або повертає розмір шрифту на закладках
Height	піксель	Задає або повертає висоту закладок
HotTrack		Задає або повертає значення, яке вказує на те, чи зміниться зовнішній вигляд вкладок після наведення на них вказівника миші
ImageList		Повертає або задає зображення, які відображаються на вкладках
ImeMode		Повертає або задає режим редактора введення закладок
ItemSize		Задає або повертає розмір вкладок
Multiline	bool	Вказує на те, чи може відображатися більше однієї вкладки
Name	String	Ім’я вкладки
RowCount	int	Повертає кількість рядків, відображених на вкладці
SelectedIndex	int	Повертає або задає індекс поточної вибраної сторінки вкладки
SelectedTab	int	Повертає або задає поточну вибрану сторінку вкладки
Site		Повертає або задає місцезнаходження вкладки
Size		Повертає або задає розміри вкладки
TabCount		Повертає або задає кількість вкладок на стрічці
TabIndex		Повертає або задає послідовність переходу по сторінкам за допомогою клавіші TAB
TabPages	колекція	Повертає або задає колекцію сторінок вкладки
Top	int	Повертає або задає відстань між верхом вкладки і верхом форми

Visible	int	Повертає або задає видимість вкладки
Width	int	Повертає або задає ширину вкладки

Таблиця 14.3

Основні методи елемента TabControl

Назва методу	Характеристика
CreateGraphics()	Створює графічний об’єкт на вкладці
Focus()	Встановлює фокус на квладці
Hide()	Приховує вкладку
OnClick(EventArgs)	Викликає подію Click
OnKeyDown(KeyEventArgs)	Викликає подію KeyDown
OnKeyPress(KeyPressEventArgs)	Викликає подію KeyPress
OnKeyUp(KeyEventArgs)	Викликає подію KeyUp
OnLostFocus(EventArgs)	Втрата фокусу
OnMouseClick(MouseEventArgs)	Викликає подію MouseClick
OnMouseDown(MouseEventArgs)	Викликає подію MouseDown
Select()	Активізує вкладки
Select(Boolean, Boolean)	Активізує сторінку вкладки або інший дочірній елемент
SelectTab(Int32)	Робить активною вкладку з вказаним індексом
SelectTab(String)	Робить активною вкладку з вказаним іменем
SelectTab(TabPage)	Робить активною вкладку TabPage
Show()	Робить вкладку видимою

Таблиця 14.4

Події елемента TabControl

Ім’я події	Характеристика
Click	Клацання лівою кнопкою миші
ClientSizeChanged	Зміна розмірів клієнтської області вкладок
DoubleClick	Подвійне клацання лівою кнопкою миші
Enter	Вхід в елемент управління
FontChanged	Зміна шрифту
GotFocus	Отримання вкладкою фокуса

KeyDown	Натискання клавіші (якщо елемент управління має фокус)
KeyPress	Натискання клавіші з літерою, пробілом або BackSpace (якщо елемент управління має фокус)
KeyUp	Відпускання клавіші (якщо елемент управління має фокус)
LostFocus	Втрата вкладкою фокуса
MouseClick	Клацання мишею на вкладці
MouseDoubleClick	Подвійне клацання мишею на вкладці
MouseDown	Натискання кнопки миші, якщо вказівник знаходиться на елементі управління
MouseEnter	Поява вказівника миші на елементі управління
MouseHover	Затримання вказівника миші на елементі управління
MouseLeave	Прибирання вказівника миші з елемента управління
MouseMove	Переміщення вказівника миші по елементу управління
MouseUp	Відпускання кнопки миші
MouseWheel	Прокручування колеса миші
Move	Переміщення елемента управління
Resize	Зміна розмірів вкладки
Selected	Вибір вкладки
SelectedIndexChanged	Зміна індексу виділеної сторінки
Selecting	Відбувається перед вибором вкладки
SizeChanged	Зміна розмірів

Завдання для виконання

1. Створити екранну форму відповідно до варіанту. Усі властивості встановити за допомогою візуальних елементів вікна середовища програмування.

2. Модифікувати програму, щоб вона виконувала дії відповідно до варіанта.

Варіанти завдань

Варіант 1

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно зелений і синій, колір шрифту – відповідно чорний і жовтий. Додати на першу сторінку кнопку, мітку і текстове поле, на другу сторінку – таблицю DataGridView (5x2).

2. Створити клас Automobils з полями Name і Year (вони служать для розміщення даних про модель і рік випуску цієї моделі автомобіля). Створити масив з 5 об’єктів і заповнити його даними. Прописати процедуру (метод)

виведення відомостей з масиву об’єктів в таблицю DataGridView на другій сторінці.

3. Написати обробник події натискання кнопки на першій сторінці, який запускає підпрограму, яка визначає найстарішу модель автомобілів і виводить її назву у текстове поле.

Варіант 2

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно сірий і темно-синій, колір шрифту – відповідно чорний і білий. Додати на першу сторінку елементи управління ComboBox, 2 мітки Label, лічильник NumericUpDown і дві кнопки Button, на другу сторінку – багаторядкове текстове поле.

2. Написати програму – аналог касового апарату, яка виводить за назвою товару (береться з елемента ComboBox) і кількість товару (береться з елемента NumericUpDown) його вартість і дописує її з нового рядка в текстове поле на другій сторінці. Запускаєтсья програма натисканням кнопки на першій сторінці.

3. Додати на першу сторінку кнопку і дописати підпрограму, яка виводитиме на принтер чек (багаторядкове текстове поле на другій сторінці). Для перевірки того, як функціонує ця підпрограма, слід встановити на комп’ютер PDF-принтер і роздрукувати на ньому чек.

Варіант 3

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно сірий, білий і зелений, колір шрифту – всюди чорний.

2. На першій вкладці програма повинна обчислювати визначник матриці 2-го порядку. Вхідні дані (елементи матриці) вводяться у текстові поля, результат виводиться також у текстове поле.

3. На другій вкладці програма повинна обчислювати визначник матриці 3-го порядку. Вхідні дані (елементи матриці) вводяться в текстові поля, результат виводиться в текстове поле. Для запуску підпрограми обчислень в обох випадках використати кнопки.

Варіант 4

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно світло-сірий і оранжевий, колір шрифту – відповідно чорний і білий.

2. На першій вкладці програма повинна обчислити площу трикутника за його трьома сторонами. Вхідні дані (три сторони трикутника) вводяться за допомогою текстових полів. Результат виводиться також у текстове поле.

3. На другій вкладці програма повинна обчислити площу трикутника за його двома сторонами і кутом між ними. Вхідні дані (сторони трикутника) вводяться в текстові поля, кут виставляється за допомогою елемента NumericUpDown (значення можуть змінюватися у межах від 0 до 90°). Результат обчислення

виводиться також у текстове поле. Для запуску підпрограми обчислень в обох випадках застосувати кнопки.

Примітка. У мові програмування C# кут виражається в радіанах, тому в програмі треба дописати команду перерахунку кута з градусів у радіани.

Варіант 5

1. Створити екранну форму, що містить 3 сторінки. Колір фону сторінок відповідно синій і білий, колір шрифту – чорний.

2. На першій вкладці програма повинна обчислювати радіус кола, описаного навколо квадрата. Вхідні дані (сторона квадрата) вводяться в текстове поле. Результат виводиться також у текстове поле.

3. На другій вкладці програма повинна обчислювати радіус кола, описаного навколо правильного трикутника. Вхідні дані (сторона трикутника) вводяться в текстове поле, результат обчислень також виводиться в текстове поле. Для запуску підпрограм, які здійснюють обчислення, в обох випадках використати кнопки.

Варіант 6

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно білий і сірий, колір шрифту – відповідно чорний і жовтий.

2. На першій вкладці програма повинна обчислювати радіус кола, вписаного у трикутник. Вхідні дані (сторони трикутника) вводяться за допомогою текстових полів, результат (радіус кола) виводиться також у текстове поле.

3. На другій вкладці програма повинна обчислювати радіус кола, вписаного в опуклий чотирикутник. Вхідні дані (довжини сторін чотирикутника) вводяться в текстові поля, результат обчислень (радіус кола) виводиться також у текстове поле. Для запуску обох підпрограм використати кнопки.

Варіант 7

1. Створити екранну форму, що містить 3 сторінки. Колір фону сторінок відповідно жовтий і синій, колір шрифту – відповідно чорний і білий.

2. На першій вкладці програма повинна обчислювати об′єм піраміди, заданої сторонами трикутника і висотою піраміди.

3. На другій вкладці програма повинна обчислювати об′єм прямого конуса за його радіусом і висотою. Для введення даних на обох вкладках використати текстові поля, для виведення результатів обчислень – також текстові поля, для запуску підпрограм – кнопки.

Варіант 8

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно зелений і жовтий, колір шрифту – відповідно білий і червоний.

2. На першій вкладці програма повинна обчислювати площу паралелограма. Вхідні дані: довжини сторін вводяться в текстові поля, кут між сторонами

задається за допомогою елемента NumericUpDown. Результат обчислення (площа) повинен виводитися також у текстове поле.

3. На другій вкладці програма повинна обчислювати площу ромба. Вхідні дані: сторона ромба вводиться в текстове поле, кут між сторонами задається за допомогою елемента NumericUpDown.

Примітка. У мові програмування C# кут виражається в радіанах, тому в програмі треба дописати команду перерахунку кута з градусів у радіани.

Варіант 9

1. Створити екранну форму, що містить 3 сторінки. Колір фону сторінок відповідно білий і сірий, колір шрифту – відповідно чорний і жовтий.

2. На першій вкладці програма повинна обчислювати об′єм циліндра. Вхідні дані (радіус циліндра і його висота) вводяться в текстові поля, результат обчислень (об′єм) виводиться в інше текстове поле.

3. На другій вкладці програма повинна обчислювати об′єм прямокутного паралелепіпеда. Вхідні дані (довжини ребер паралелепіпеда) вводяться в текстові поля. Результат обчислень (об′єм) виводиться в інше текстове поле. Обидві підпрограми приводяться в дію за допомогою кнопок.

Варіант 10

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно синій і жовтий, колір шрифту – відповідно білий і чорний.

2. На першій вкладці програма повинна обчислювати площу поверхні прямого циліндра. Вхідні дані (радіус і висота циліндра) вводяться в текстові поля.

3. На другій вкладці програма повинна обчислювати площу поверхні прямого конуса. Вхідні дані (радіус і висота конуса) вводяться в текстові поля. Обидві підпрограми виводять результати обчислень у текстові поля і приводяться в дію кнопками.

Варіант 11

1. Створити екранну форму, що містить 3 сторінки. Колір фону сторінок відповідно помаранчевий і фіолетовий, колір шрифту – білий.

2. На першій вкладці програма повинна обчислювати суму n перших членів арифметичної прогресії. Вхідні дані (перший член прогресії a1, різниця прогресії d) вводяться в текстові поля, кількість елементів n – в елемент управління NumericUpDown. Результат (сума) виводиться в текстове поле.

3. На другій вкладці програма повинна обчислювати суму n перших членів геометричної прогресії. Вхідні дані (перший член прогресії b, множник q) вводяться в текстові поля, кількість членів n – в елемент управління NumericUpDown. Результат (сума) виводиться в текстове поле. Обидві підпрограми запускаються на виконання за допомогою текстових кнопок.

Варіант 12

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно жовтий і зелений, колір шрифту – відповідно чорний і білий.

2. На першій вкладці програма повинна виводити номер літери англійського алфавіту.

3. На другій вкладці програма повинна виводити символ англійського алфавіту за його номером. Для введення і виведення даних використати текстові поля, для запуску підпрограм – кнопки.

Варіант 13

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно зелений і синій, колір шрифту – чорний.

2. На першій сторінці програма повинна зберігати в файл output.txt текст, набраний користувачем. Для введення тексту використовується багаторядкове текстове поле, для запуску програми – кнопка.

3. На другій вкладці програма повинна зчитувати текст з файла input.txt і виводити його в багаторядкове текстове поле. Для виведення тексту використати текстове поле, для запуску підпрограми – кнопку.

Варіант 14

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно фіолетовий і червоний, колір шрифту – відповідно білий і чорний.

2. На першій сторінці програма повинна обчислювати кількість пробілів у введеному тексті. Для введення тексту використати багаторядкове текстове поле. Результати вивести в однорядкове текстове поле.

3. На другій сторінці програма повинна перетворити рядок тексту в масив символів. Введення тексту здійснюється в текстове поле, виведення результату – в елемент DataGridView. Запуск обох підпрограм повинен здійснюватися натисканням кнопок.

Варіант 15

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно зелений і синій, колір шрифту – білий.

2. На першій сторінці програма повинна перетворити рядок тексту у верхній регістр і вивести його в однорядкове поле.

3. На другій сторінці програма повинна вивести в стовпчик у багаторядкове поле слово, введене користувачем. Для введення даних використати однорядкові текстові поля, для запуску підпрограм – кнопки button.

Варіант 16

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно коричневий і помаранчевий, колір шрифту – відповідно білий і чорний.

2. На першій сторінці програма повинна зчитати з текстового поля рядок символів, поміняти місцями символи з парними і непарними символами, вивести результат у текстове поле.

3. На другій сторінці програма повинна зчитати рядок символів і вивести перше слово (воно відокремлене пробілом). Введення і виведення даних здійснювати в текстове поле. Для запуску обох підпрограм використати кнопки.

Варіант 17

1. Створити екранну форму, що містить 3 сторінки. Колір фону сторінок коричневий, колір шрифту – білий.

2. На першій сторінці програма повинна вивести випадковий символ англійського алфавіту в текстове поле.

3. На другій сторінці програма повинна вивести випадковий символ українського алфавіту в текстове поле. Для запуску обох підпрограм використати кнопки button.

Варіант 18

1. Створити екранну форму, що містить 3 сторінки. Колір фону сторінок зелений, колір шрифту – білий.

2. На першій сторінці програма повинна обчислювати кількість слів у рядку (слова відокремлюються одне від одного пробілами). Для введення і виведення даних використати текстове поле, для запуску підпрограми – кнопку.

3. На другій сторінці програма повинна вилучити всі літери «а» (латинськими літерами) з рядка, введеного користувачем. Рядок теж вводиться латинськими літерами. Для введення і виведення даних використовуються текстові поля, для запуску підпрограми – кнопка.

Варіант 19

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно жовтий і зелений, колір шрифту – відповідно червоний і чорний.

2. На першій сторінці програма повинна обчислювати довжину відрізка на площині. Відрізок задається координатами кінців. Для введення і виведення даних використати текстові поля, для пуску підпрограми – кнопку.

3. На другій сторінці програма повинна обчислювати довжину відрізка в просторі. Відрізок задається координатами кінців. Для введення і виведення даних використати текстові поля, для запуску підпрограми – кнопку.

Варіант 20

1. Створити екранну форму, що містить 3 сторінки. Колір фону сторінок відповідно синій, зелений і сірий, колір шрифту – чорний.

2. На першій сторінці програма повинна обчислювати довжину діагоналі прямокутного паралелепіпеда. Вхідні дані (довжини ребер паралелепіпеда)

вводяться в текстові поля, результат обчислень виводиться також у текстове поле.

3. На другій сторінці програма повинна обчислювати довжину діагоналі прямокутника. Вхідні дані (довжини сторін прямокутника) вводяться в текстові поля, результат також виводиться у текстове поле. Для запуску обох підпрограм використати кнопки.

Варіант 21

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно коричневий і помаранчевий, колір шрифту – відповідно білий і синій.

2. На першій сторінці програма повинна обчислювати довжину найкоротшого слова з рядка, введеного в текстове поле.

3. На другій сторінці програма повинна обчислювати довжину найбільшого слова з рядка, введеного в текстове поле. В обох випадках результати обчислень виводяться в текстові поля, підпрограми запускаються за допомогою кнопок.

Варіант 22

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно білий і зелений, колір шрифту – відповідно чорний і жовтий.

2. На першій сторінці програма повинна обчислювати площу поверхні кулі.

3. На другій сторінці програма повинна обчислювати площу поверхні куба. Для введення і виведення даних використати текстові поля, для запуску підпрограм – кнопки.

Варіант 23

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно зелений і синій, колір шрифту – відповідно чорний і білий.

2. На першій сторінці програма повинна обчислювати периметр паралелограма.

3. На другій сторінці програма повинна обчислювати периметр ромба. В обох випадках введення і виведення даних здійснюється за допомогою текстових полів, запуск підпрограм здійснюється за допомогою кнопок button.

Варіант 24

1. Створити екранну форму, що містить 3 сторінки. Колір фону сторінок синій, колір шрифту – білий.

2. На першій сторінці програма повинна виводити випадкове число в межах від a до b. Обидва числа вводяться за допомогою елемента NumericUpDown. Результат виводиться в текстове поле.

3. На другій сторінці програма повинна обчислювати різницю двох випадкових чисел у межах від t1 до t2. Числа t1 і t2 вводяться за допомогою NumericUpDown, результат виводиться в текстове поле. Обидві підпрограми запускаються за допомогою кнопок button.

Варіант 25

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок коричневий, колір шрифту – білий.

2. На першій вкладці програма повинна обчислювати третій кут трикутника. Два інші кути задаються за допомогою елементів NumericUpDown, результат виводиться у текстове поле.

3. На другій сторінці програма повинна обчислювати кути при вершинах ромба. Користувач вводить значення одного з кутів за допомогою NumericUpDown. Результат обчислень виводиться в елемент NumericUpDown. Обидві підпрограми запускаються за допомогою кнопок.

Варіант 26

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно синій і зелений, колір шрифту – відповідно білий і жовтий.

2. На першій сторінці програма повинна обчислювати висоту трикутника. Вхідні дані – довжини сторін – вводяться за допомогою текстових полів, а кут між цими сторонами – за допомогою NumericUpDown.

3. На другій сторінці програма повинна обчислювати площу трикутника, заданого трьома сторонами. Вхідні дані вводяться за допомогою текстових полів. В обох підпрограмах виведення даних здійснюється в текстові поля, запуск підпрограм – за допомогою кнопок.

Варіант 27

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно білий і зелений, колір шрифту – відповідно чорний і білий.

2. На першій вкладці програма повинна протабулювати функцію $y=3x^2$ на проміжку від a до b з кроком 1. Числа a до b вводяться за допомогою текстових полів. Результати виводяться в багаторядкове текстове поле.

3. На другій вкладці програма повинна обчислити відстань від точки M(x; y; z) до початку координат. Координати точки x, y, z вводяться за допомогою NumericUpDown, результат виводиться в текстове поле. Обидві підпрограми запускаються за допомогою кнопок.

Варіант 28

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно сірий і жовтий, колір шрифту – відповідно білий і зелений.

2. На першій вкладці програма повинна обчислити довжину твірної прямої піраміди, в основі якої лежить квадрат. Вхідні дані (сторона квадрата і висота піраміди) вводяться за допомогою текстових полів, результат виводиться також в текстове поле.

3. На другій вкладці програма повинна обчислити площу перерізу піраміди, який проходить через її висоту і діагональ основи. Вхідні дані (сторона квадрата

і висота піраміди) вводяться за допомогою текстових полів, результат виводиться також у текстове поле. Обидві програми приводяться у дію кнопками.

Варіант 29

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно зелений і оливковий, колір шрифту – відповідно чорний і білий.

2. На першій вкладці програма повинна обчислювати період власних коливань математичного маятника.

3. На другій вкладці програма повинна обчислювати період власних коливань пружинного маятника. Введення і виведення даних в обох випадках здійснюється за допомогою елементів NumericUpDown.

Варіант 30

1. Створити екранну форму, що містить 2 сторінки. Колір фону сторінок відповідно оливковий і синій, колір шрифту – відповідно білий і жовтий.

2. На першій вкладці програма повинна обчислювати радіус кола, описаного навколо квадрата.

3. На другій вкладці програма повинна обчислювати радіус кола, описаного навколо правильного трикутника. Вхідні дані вводяться в текстові поля, результати виводяться також у текстові поля. Для запуску підпрограм використати кнопки.

Контрольні питання

1. Що являє собою елемент управління TabControl?

2. Що являє собою елемент управління TabPage?

3. Як змінити кількість сторінок в елементі управління TabControl за допомогою візуального редактора?

4. Як змінити кількість сторінок в елементі управління TabControl програмним шляхом?

5. Які елементи управління можна розмістити на вкладках?

6. Які елементи управління можна розмістити всередині елемента TabControl?

7. Яка подія відповідає завантаженню форми?

8. Яка подія відповідає закриттю форми?

9. Який елемент управління необхідно додати на форму, щоб запускався друк текстового поля?

10. Яка властивість означає активну сторінку?

11. Яка властивість визначає кількість сторінок?

12. Яка властивість визначає назви сторінок?

Приклад програми (варіант 1)

```
using System;
```

```
using System.Collections.Generic;
using System.Drawing;
using System.Windows.Forms;
using System.Data;
using System.Windows.Forms.ComponentModel;
using System.Management;
namespace lab14v1t1
{
   /// <summary>
   /// Description of MainForm.
   /// </summary>
   public partial class MainForm : Form
   {
      public MainForm()
      {
         //
         InitializeComponent();

      }
public class Automobils {
         public string Name;
         public int Year;
         public Automobils(string NName, int YYear) {
            this.Name=NName;
            this.Year=YYear;
         }
      }

      Automobils []auto=new Automobils[5];
int i=0;
void MainFormLoad(object sender, EventArgs e)
      {

         for(i=0;i<5;i++) {
            auto[i]=new Automobils("",0);
         }
         this.dataGridView1.RowCount=2;
         /*
         auto[0]=new Automobils("Volkswagen",2011);
```

```
        auto[1]=new Automobils("Peugeot ",2019);

    auto[3]=new Automobils("Mers",2015);
    auto[4]=new Automobils("Lincoln",2001);
            */
            names[0]="Volks1";
            names[1]="Peugeot";
            names[2]="Mers";
            names[3]="Audi";
            names[4]="Lincoln";
            years[0]=2015;
            years[1]=2018;
            years[2]=2009;
            years[3]=1998;
            years[4]=2016;
            auto[0]=new Automobils("volkswagen",2015);
            auto[1]=new Automobils("Reno",2007);
            auto[2]=new Automobils("Volvo",2002);
            auto[3]=new Automobils("Jaguar",2011);
            auto[4]=new Automobils("Ford",2019);
}
void Button1Click(object sender, EventArgs e)
{
    int min=auto[0].Year;
    for (i=0;i<5;i++) {
        if(min>auto[i].Year) {
            min=auto[i].Year;
        }
    }
    string sss=Convert.ToString(min);
    textBox1.Text=sss;
}
void Button2Click(object sender, EventArgs e)
{
    this.dataGridView1.RowCount=2;
    for (i=0;i<5;i++) {
    //DataGridViewCell [0,i] c =new DataGridViewCell[0,i];
    }
    string ss=" ";
```

```
        // MessageBox.Show(names[4]);
        /*for(i=0;i<5;i++) {
           //this.dataGridView1[i,0].ValueType=string;
           this.dataGridView1[0,i].Value=names[i];
           ss=Convert.ToString(years[i]);
           this.dataGridView1[1,i].Value=Convert.ToString(years[i]);
        }
        this.dataGridView1[0,4].Value=names[4];
        */
        for(i=0;i<5;i++) {
           //this.dataGridView1[i,0].ValueType=string;
           this.dataGridView1[i,0].Value=names[i];
           ss=Convert.ToString(years[i]);
           this.dataGridView1[i,1].Value=Convert.ToString(years[i]);
        }
     }
     void Button3Click(object sender, EventArgs e)
     {
        this.dataGridView1.RowCount=2;
        string ss=" ";
           for(i=0;i<5;i++) {
           //this.dataGridView1[i,0].ValueType=string;
           this.dataGridView1[i,0].Value=auto[i].Name;
           ss=Convert.ToString(auto[i].Year);
           this.dataGridView1[i,1].Value=ss;
        }
     }
  }
}
```

Лабораторна робота №15

Тема: Таймер і рядок статусу

Мета: набути навичок створення екранних форм з використанням таймера рядка статусу.

Обладнання:

Короткі теоретичні відомості

Елемент управління Timer призначено для запуску дій, які виконуються періодично. Події таймера відбуваються з певним інтервалом.

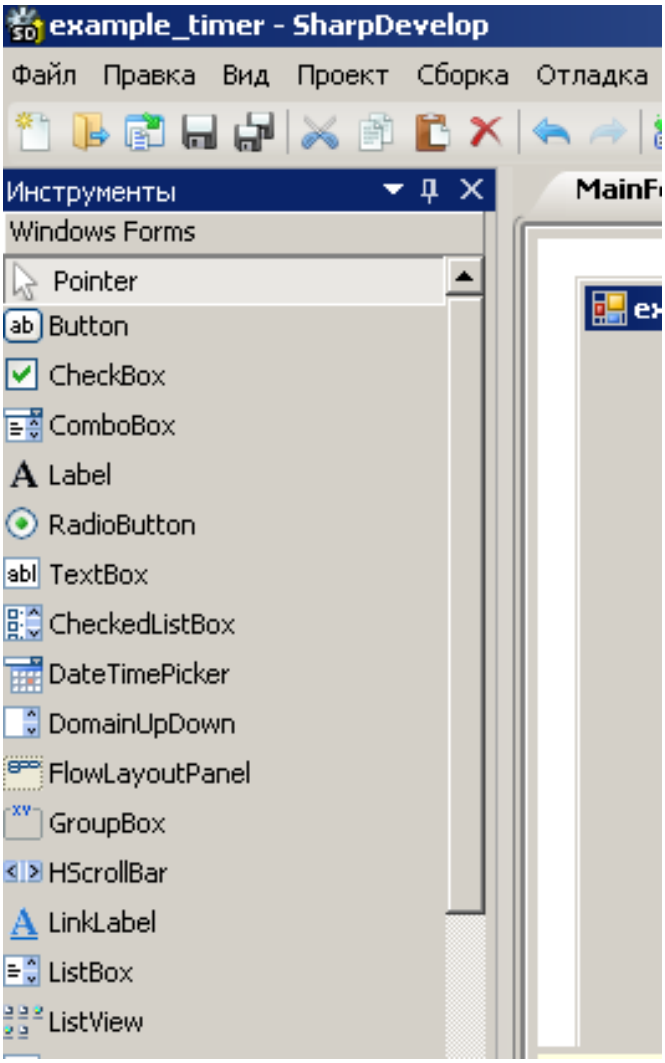

Рис.15.1. Таймер

Таймери служать для періодичного виклику певного метода (підпрограми). Цей елемент управління має єдину подію – Tick. Обробник цієї події викликається з періодичністю, заданою програмістом. Для функціонування таймера необхідний простір імен System.Timers.

Елемент управління Timer невидимий.

Послідовність створення таймера:

1) створюємо новий проект Windows Forms;

2) на панелі елементів знаходимо таймер і перетягуємо його на форму (рис.15.2);

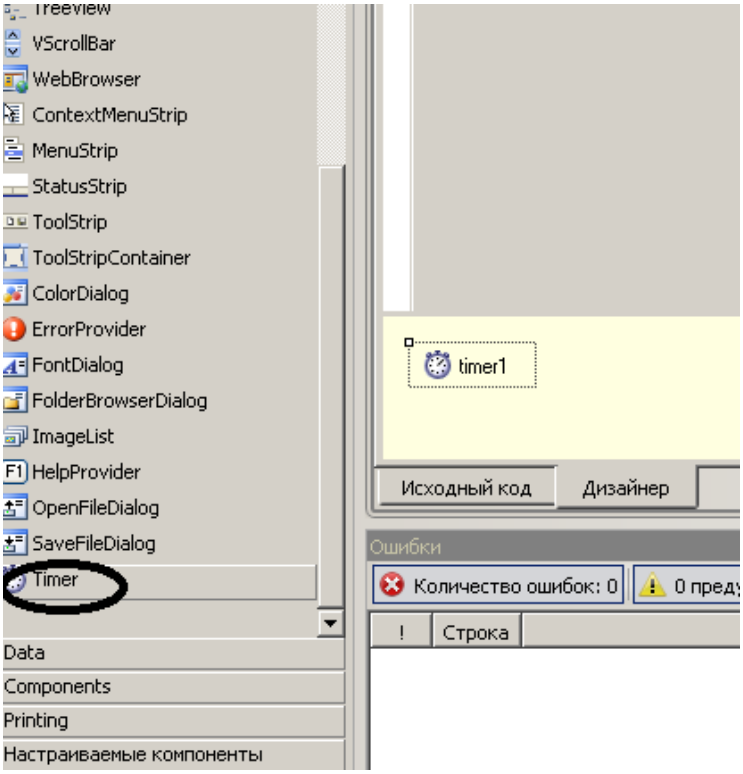

Рис.15.2. Таймер на Панелі інструментів

3) вибрати компонент на формі. Він розміщений у контейнері нижче форми (рис.15.3);

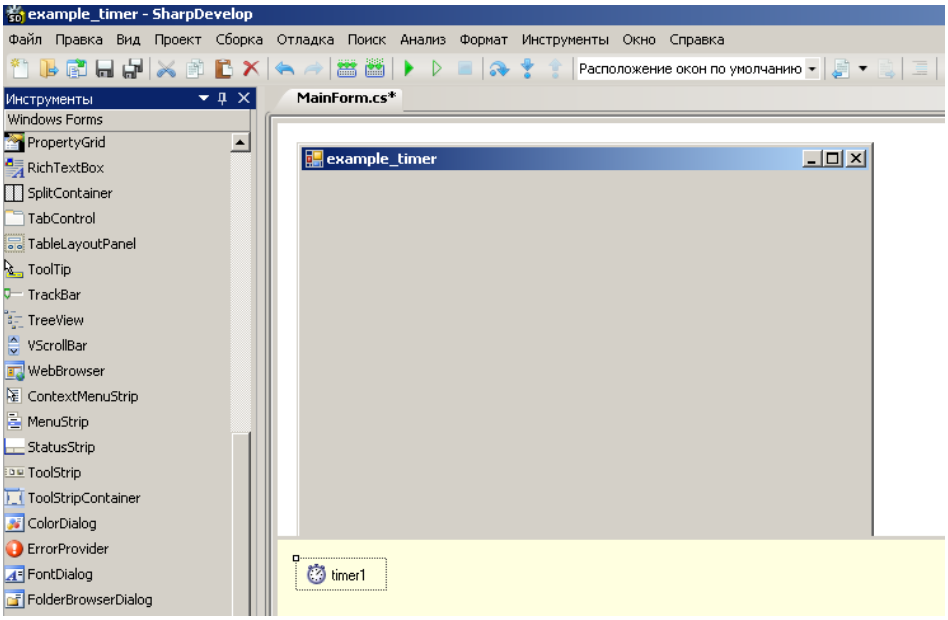

Рис.15.3. Компонент Timer на екранній формі

4) переглянути властивості елемента управління Timer (рис.15.4);

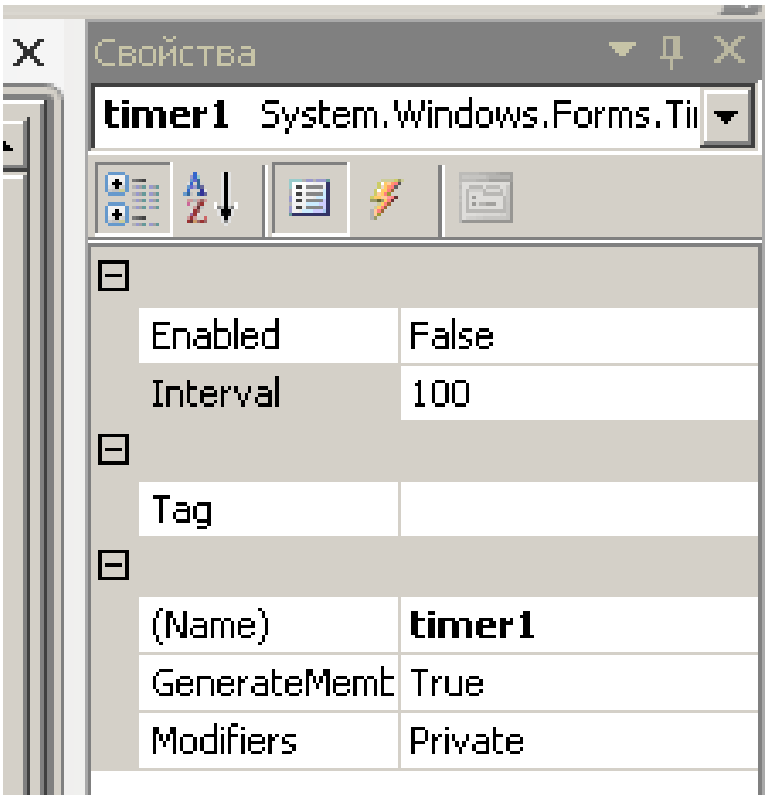

Рис.15.4.

5) переглянути значення властивостей Timer (Табл.15.1)

Таблиця 15.1

Властивості Timer

Назва	Характеристика
Name	Назва таймера (унікальне ім´я, яке не повинно повторюватися в тексті програми)
Enable	Робить таймер доступним
Interval	Значення інтервалу, після якого буде викликатися таймер. Задається в мілісекундах
Modifiers	Доступ (загальний чи захищений)

6) ознайомитися з методами Timer

Таблиця 15.2

Методи Timer

Метод	Опис
Start	Запускає таймер
Stop	Зупиняє таймер

7) розглянемо події таймера (рис.15.5):

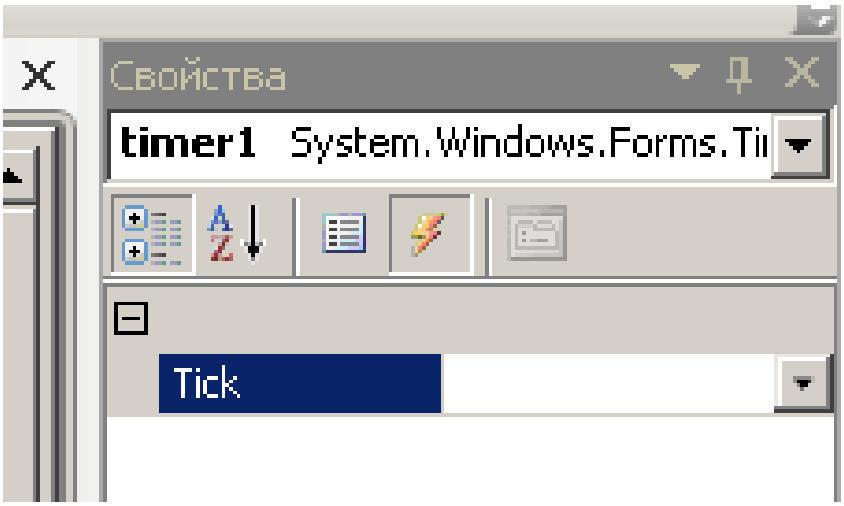

Рис.15.5. Події таймера

Створимо програму, яка виводить значення часу в компонент TextBox на формі. Для цього створимо форму, додамо на неї дві кнопки (пуску і зупинки таймера), мітку, текстове поле і таймер (рис.1.5.6).

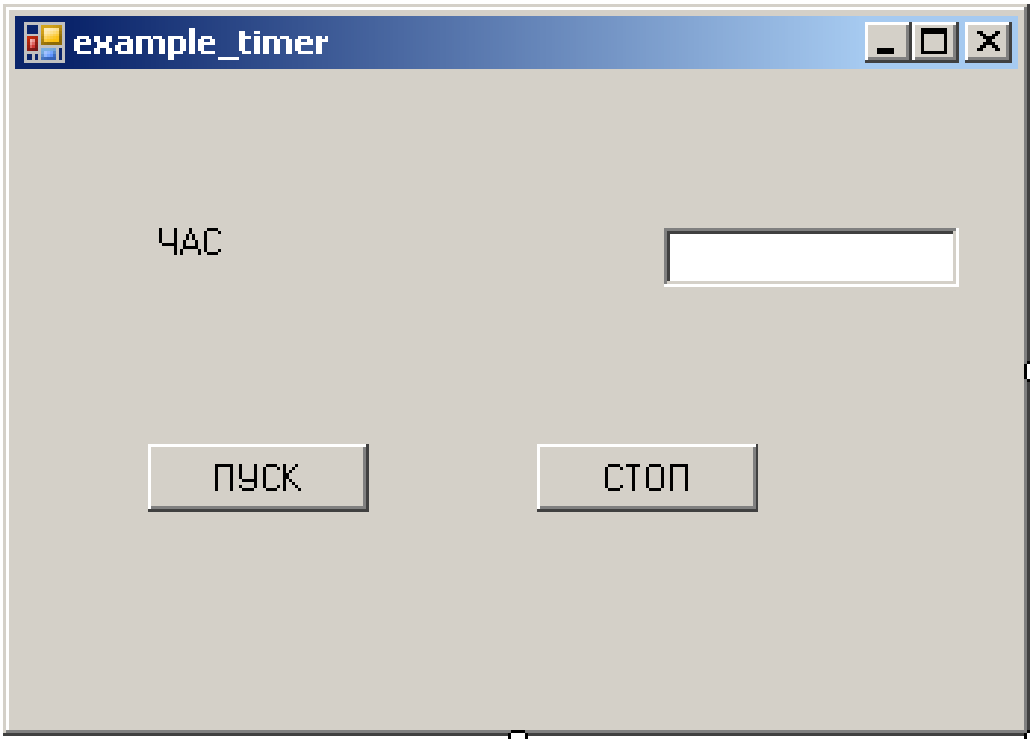

Рис.15.6. Зовнішній вигляд форми

Запрограмуємо пуск таймера. Для цього двічі клацнемо на кнопці ПУСК і введемо код підпрограми:

```
void Button1Click(object sender, EventArgs e)
      {
         timer1.Interval=1000;
         timer1.Enabled=true;
         timer1.Start();
      }
```

Далі запрограмуємо виведення часу в текстове поле. Для цього слід двічі клацнути мишею на таймері і ввести текст підпрограми:

```
void Timer1Tick(object sender, EventArgs e)
    {
      DateTime DDT=DateTime.Now.ToLocalTime();
      string ss=DDT.ToString("HH:mm:ss");
      textBox1.Text=ss;
    }
```

Наступний крок – програмування зупинки таймера. Для цього обираємо кнопку СТОП і двічі клацаємо по ній:

```
void Button2Click(object sender, EventArgs e)
    {
      timer1.Stop();
    }
```

Результат роботи програм показано на рис.15.7.

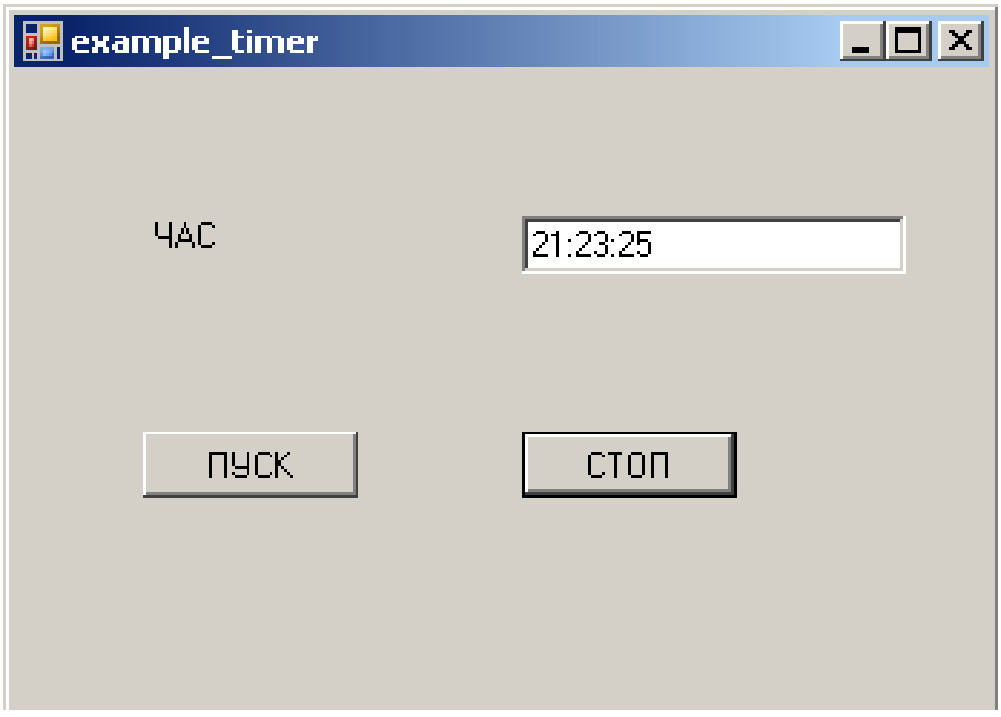

Рис.15.7. Програма у процесі роботи

Дата і час

Для роботи з датою і часом існує спеціальний об´єкт DateTime. Цей об´єкт містить поточні значення місцевого дати і часу. Ці дані виражені в мілісекундах від півночі 01.01.0001 р.

Для роботи з датою і часом необхідна бібліотека System.Runtime.dll і підключення простору імен System.Globalization.

Розглянемо властивості класу DateTime детальніше.

Властивість DateTime.Now повертає об´єкт DateTime, який містить поточні дату і час даного комп´ютера. Вони виражені у форматі місцевого часу.

Властивість Now повертає значення DateTime, яке представляє поточну дату і час на локальному комп´ютері. Наприклад:

Роздільна здатність Now залежить від системного таймера, який залежить від базової операційної системи, і може мати значення від 0,5 до 15 мілісекунд.

Властивість DateTime.Date повертає дату, яка міститься в об´єкті типу DateTime. Має вигляд

Властивість DateTime.Day повертає день місяця, який міститься в об´єкті типу DateTime. Може мати ціле значення від 1 до 31 (за кількістю днів у місяці).

Властивість DateTime.DayOfWeek повертає день тижня, яке міститься в об´єкті типу DateTime. Тип – ціле число від 0 до 6.

Властивість DateTime.DayOfYear повертає поточний день року, який міститься в об´єкті типу DateTime. Тип – ціле число від 1 до 366.

Властивість DateTime.Hour повертає значення годин, яке міститься в об´єкті типу DateTime. Тип – ціле число від 0 до 23. Ця властивість завжди виражається в 24-годинному форматі. Для того, щоб змінити формат даних на 12-годинний формат, слід використати метод DateTime.ToString(String, IFormatProvider) або формат h. Наприклад:

```
DateTime date1 = new DateTime(2008, 4, 1, 18, 53, 0);
Console.WriteLine(date1.ToString("%h"));            // Displays 6
Console.WriteLine(date1.ToString("h tt"));          // Displays 6 PM
```

Властивість DateTime.Kind виводить перелічуване значення поточного часу. Значення за замовчуванням – Unspecified. Це значення вказує, на якому часі засновано час, який представлений цим екземпляром: місцевому або UTC, або ні на тому, ні на іншому.

Властивість DateTime.Millisecond повертає кількість мілісекунд для дати, яка міститься в даному об´єкті DateTime. Може мати значення від 0 до 999. Тип значень – ціле число (Int32).

Мілісекунди можна вивести в рядок за допомогою формату “fff”:

```
DateTime date1 = new DateTime(2008, 1, 1, 0, 30, 45, 125);
Console.WriteLine("Milliseconds: {0:fff}",
                  date1);          // displays Milliseconds: 125
```

Можна вивести мілісекунди разом з іншими компонентами значення дати і часу:

```
DateTime date2 = new DateTime(2008, 1, 1, 0, 30, 45, 125);
Console.WriteLine("Date: {0:o}",
                  date2);
// Displays the following output to the console:
```

```
//      Date: 2008-01-01T00:30:45.1250000
```

Можна відобразити мілісекунди з іншими даними дати і часу за допомогою рядка налаштовуваного формату:

```
DateTime date3 = new DateTime(2008, 1, 1, 0, 30, 45, 125);
Console.WriteLine("Date with milliseconds: {0:MM/dd/yyy HH:mm:ss.fff}",
                  date3);
```

Властивість DateTime.Minute повертає хвилини, які містяться в об´єкті DateTime. Може мати значення від 0 до 59 (ціле число).

Властивість DateTime.Month повертає значення місяця, яке міститься в об´єкті DateTime. Може мати значення від 1 до 12 (тип Int32).

Властивість DateTime.Seconds повертає значення секунд, яке міститься в об´єкті DateTime. Це ціле число в межах від 0 до 59.

Властивість DateTime.Ticks повертає кількість тактів, у вигляді якої представлено значення дати і часу цього екземпляра. Має тип Int64. Це значення знаходиться в межах від DateTime.MinValue.Ticks до DateTime.MaxValue.Ticks.

У наступному прикладі ця властивість використовується для виведення кількості тактів, які пройшли з початку XXI століття.

Один такт являє собою 100 нс або 110-мільйон секунди, 10000 тактів за мілісекунду і 10000000 тактів за секунду. Значення цієї властивості – кількість 100-нс інтервалів, які пройшли з 12.00 півночі 1.01.2001 (це MinValue). Зазвичай такти представляють у відповідності до часового поясу, заданого властивістюKind.

Властивість DateTime.TimeOfDay повертає час дня для об´єкта DateTime у вигляді числа, яке вказує на інтервал часу, що пройшов від півночі до вказаного часу.

Властивість DateTime.Today повертає поточну дату типу DateTime, значення часу для якого встановлено у вигляді 00:00:00.

Властивість DateTime.UtcNow повертає поточні значення дати і часу даного комп´ютера, виражені в форматі UTC.

Роздільна здатність цієї властивості залежить від системного таймера, який залежить від операційної системи, і може становити від 0,5 до 15 мс.

Властивість DateTime.Year повертає значення року для об´єкта DateTime в межах від 1 до 9999.

Властивість Year повертає значення року поточного об´єкта у григоріанському календарі.

Методи об´єкта DateTime

Метод DateTime.ToLocalTime() перетворює значення поточного об´єкта у формат місцевого часу, поданий відповідно регіональних налаштувань операційної системи.

Місцевий час дорівнює сумі часу UTC і зміщення UTC. Також враховується перехід на літній і зимовий час.

Метод DateTime.ToLongDateString() перетворює значення поточного об´єкта DateTime в еквівалентний йому довгий рядковий формат представлення даних. Повертає значення у вигляді рядка (string).

Метод DateTime.ToLongTimeString() перетворює значення об´єкта DateTime в еквівалентний йому довгий рядок. Повертає значення рядкового типу.

Метод DateTime.ToShortDateString() перетворює значення поточного об´єкта DateTime в еквівалентний йому короткий рядок, який представляє дату.

Рядок, який повертаєтсяь цим методом, залежить від мові і регіональних параметрів.

Метод DateTime.ToShortTimeString перетворює значення поточного об´єкта DateTime в еквівалентний йому короткий рядок. Повертає значення рядкового типу.

Рядок, який повертається цим методом, залежить від мови і регіональних налаштувань локального комп´ютера.

Метод DateTime.ToString() перетворює значення поточного об´єкта DateTime в еквівалентний йому рядок. Цей метод є перевантаженим, тобто може мати різний синтаксис. Наприклад:

Таблиця 15.3

Перевантаження методу

Метод	Перевантаження
ToString(String, IFormatProvider)	Перетворює значення поточного об´єкта DateTime в еквівалентний йому рядок з використанням вказаного формату і відомостей про особливості формату для даної мови і регіональних параметрів
ToString(String)	Перетворює значення поточного об´єкта DateTime в еквівалентний йому рядок з використанням вказаного формата і домовленостей про форматування, прийнятих для поточної мови і регіональних параметрів
ToString(IFormatProvider)	Перетворює значення поточного об´єкта DateTime в еквівалентний йому рядок з використанням вказаних відомостей про форматування, пов´язаних з мовою і регіональними параметрами

ToString()	Перетворює значення поточного об´єкта DateTime в еквівалентний йому рядок за допомогою домовленостей про форматування для поточної мови і регіональних параметрів

Розглянемо різні випадки форматування DateTime за допомогою різних варіантів DateTimeFormatInfo:

```
using System;
 using System.Globalization;
public class MainClass {
   public static void Main(string[] args)  {
      DateTime dt = DateTime.Now;
      String[] format = {
         "d", "D",
         "f", "F",
         "g", "G",
         "m",
         "r",
         "s",
         "t", "T",
         "u", "U",
         "y",
         "dddd, MMMM dd yyyy",
         "ddd, MMM d \"\"\"yy",
         "dddd, MMMM dd",
         "M/yy",
         "dd-MM-yy",
      };
      String date;
      for (int i = 0; i < format.Length; i++) {
         date = dt.ToString(format[i], DateTimeFormatInfo.InvariantInfo);
         Console.WriteLine(String.Concat(format[i], " :" , date));
      }
   }
}
```

Метод DateTime.ToUniversalTime() перетворює значення поточного об´єкта DateTime в час UTC. Значення, яке повертається, також має формат DateTime.

Метод DateTime.TryFormat (Span<Char>, Int32, ReadOnlySpan<Char>, IFormatProvider) форматує значення поточного об´єкта DateTime в умовний діапазон символів. Повертає значення типу Boolean: true – у випадку, якщо форматування проведено успішно, і false – якщо трапився збій.

Рядок статусу

Елемент управління StatusStrip застосовується у формах у якості області у нижній частині вікна, у якій виводяться різні відомості про стан програми. Елемент управління StatusStrip звичайно має кілька вкладених елементів управління:

- ToolStripStatusLabel – панель елементів управління StatusStrip;
- ToolStripSplitButton – елемент управління, який складається з двох частин: кнопки і меню, яке розкривається;
- ToolStripProgressBar – елемент, який відображає стан певного процесу;
- ToolStripDropDownButton – випадне меню, з якого користувач може вибрати одиничний елемент.

Для функціонування StatusStrip необхідно підключення просторів імен System.Design, System.Drawing и System.Windows.Forms.

Постановка завдання

1. Написати програму, яка використовує таймер.

2. Написати програму, яка використовує рядок статусу.

Варіанти завдань

Варіант 1

1. Написати програму, яка здійснює підрахунок секунд між натисканням двох кнопок.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripStatusLabel повинна виводитися поточна дата.

Варіант 2

1. Написати програму, яка зафарбовує елемент управління Label по черзі в червоний і зелений колір (за принципом дії світлофора для пішоходів). Кольори повинні змінюватися кожні 15 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripSplitButton. В елемент ToolStripStatusLabel повинен виводитися поточний час (у годинах і хвилинах). Для виведення часу встановити інтервал таймера 0,5 с.

Варіант 3

1. Написати програму, яка зафарбовує елемент управління Label по черзі в червоний, жовтий і зелений колір (за принципом дії світлофора для автомобілів). Кольори повинні змінюватися кожні 15 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripDropDownButton. В елемент ToolStripStatusLabel повинно виводитися ім´я користувача.

Варіант 4

1. Написати програму, яка зафарбовує елемент GroupBox1 по черзі в 7 основних кольорів: червоний, оранжевий, зелений, синій, фіолетовий. Інтервал часу – 3 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripStatusLabel повинно виводитися поточне значення хвилин і секунд. Час повинен оновлюватися щосекунди.

Варіант 5

1. Написати програму, яка показує / приховує елемент управління Label з інтервалом 0,5 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripSplitButton. В елемент ToolStripStatusLabel виводиться інтервал часу (у хвилинах і секундах), який пройшов з моменту запуску програми. Інтервал оновлення таймера – 0,5 с.

Варіант 6

1. Написати програму, яка показує / приховує елемент управління GroupBox з інтервалом, значення якого може задати користувач у межах від 0,1 до 2 с. Для введення інтервалу слід використати елемент управління NumericUpDown.

2. Написати програму з рядком статусу, який включатиме такі елементи: 2 ToolStripStatusLabel, ToolStripDropDownButton. В елементи ToolStripStatusLabel вивести відповідно: 1) поточний місяць; 2) поточний час в секундах.

Варіант 7

1. Написати програму, яка змінює кольори елемента управління TextBox у такому порядку: червоний, синій, зелений. Інтервал таймера повинен встановлюватися користувачем за допомогою компонента NumericUpDown.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripSplitButton. В елемент ToolStripStatusLabel виводити позмінно своє прізвище і поточну дату з інтервалом 3 с.

Варіант 8

1. Написати програму, яка виводить повідомлення за допомогою MessageBox з інтервалом, заданим користувачем за допомогою елемента управління NumericUpDown. Інтервал повинен бути в межах [a; b].

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripProgressBar вивести стан процесу обчислення суми чисел від 1 до 100 з інтервалом часу 1 с для кожного кроку ітерації.

Варіант 9

1. Написати програму, яка являтиме собою секундомір і виводитиме значення проміжку часу в секундах від натискання кнопки ПУСК до натискання кнопки СТОП.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripDropDownButton. В елемент ToolStripStatusLabel вивести поточний рік.

Варіант 10

1. Написати програму, яка зберігатиме поточне значення елемента TextBox в текстовий файл. Інтервал таймера встановлює користувач у межах [a;b], які вводяться в елемент управління NumericUpDown.

2. Написати програму з рядком статусу, який включатиме такі елементи: 2 ToolStripStatusLabel, ToolStripSplitButton. В елементи ToolStripStatusLabel вивести відповідно: 1) поточний час (години: хвилини: секунди); 2) поточний місяць.

Варіант 11

1. Написати програму, яка змінюватиме колір шрифту елемента управління Label з частотою [a;b]. Частота – величина, обернена інтервалу.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripStatusLabel вивести поточну дату у форматі UTC.

Варіант 12

1. Написати програму, яка буде зберігати в файл вміст багаторядкового текстового поля через кожні 5 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: 2 ToolStripStatusLabel, ToolStripDropDownButton. В елементи ToolStripStatusLabel вивести відповідно: 1) поточне значення дня тижня; 2) поточне значення числа місяця.

Варіант 13

1. Написати програму, яка кожної секунди збільшує значення лічильника і та виводить його у багаторядкове текстове поле. Для запуску і зупинки таймера використати 2 кнопки.

2. Написати програму з рядком статусу, який включатиме такі елементи: 2 ToolStripStatusLabel, ToolStripSplitButton. В елементи ToolStripStatusLabel вивести відповідно: 1) поточний час дня; 2) поточний рік.

Варіант 14

1. Написати програму, яка підраховує кількість секунд, що пройшли від моменту запуску програми до натискання кнопки СТОП. Використати таймер.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripStatusLabel вивести поточне число місяця.

Варіант 15

1. Написати програму, яка змінює видимість елемента управління Label, на якому записаний номер вашої групи. Інтервал перемикання видимості Label – 1 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripDropDownButton. В елемент ToolStripStatusLabel вивести поточне значення годин і хвилин в регіональному форматі.

Варіант 16

1. Написати програму, яка закриває програму через кількість секунд, задану користувачем за допомогою елемента NumericUpDown. Використати таймер.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripSplitButton. В елемент ToolStripStatusLabel вивести поточне значення хвилин і секунд у регіональному форматі.

Варіант 17

1. Написати програму, яка перемикає видимість елемента управління GroupBox з частотою 2 Гц.

2. Написати програму з рядком статусу, який включатиме такі елементи: 2 ToolStripStatusLabel, ToolStripProgressBar. В елементи ToolStripStatusLabel вивести відповідно: 1) день тижня; 2) число.

Варіант 18

1. Написати програму, яка виводить в елемент управління TextBox випадкове число в межах від 1 до 100 з частотою зміни чисел 2 Гц.

2. Написати програму з рядком статусу, який включатиме такі елементи: 2 ToolStripStatusLabel, ToolStripDropDownButton. В елементи ToolStripStatusLabel повинно виводитися відповідно: 1) число, місяць; 2) день тижня.

Варіант 19

1. Написати програму, яка зафарбовує елемент управління GroupBox по черзі в червоний, синій, зелений і фіолетовий кольори. Інтервал між змінами кольору – 0,2 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripStatusLabel вивести почергово 1) назву поточного дня тижня; 2) поточну дату. Інтервал між змінами написів – 2 с.

Варіант 20

1. Написати програму, яка виводить в елемент управління TextBox 20 випадкових чисел в межах від 0 до 100 кожні 0,9 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripSplitButton. В елемент ToolStripStatusLabel виводити значення дати і часу в регінальному форматі з оновленням кожні 0,5 с.

Варіант 21

1. Написати програму, яка виводить відлік до опівночі (0:00:00) у текстове поле.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripDropDownButton. В елемент ToolStripStatusLabel вивести секундомір у форматі хв:с:мс. Інтервал оновлення – 0,1 с.

Варіант 22

1. Користувач вводить ціле число N. Написати програму, яка моделює принцип дії зворотного таймера і виводить результат у багаторядкове текстове поле. Значення змінної N повинно зменшутися щосекунди на 1.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripStatusLabel вивести значення дати і часу у форматі UTC.

Варіант 23

1. Написати програму, яка імітує спалахування і згасання лампочки (елемент Label). Інтервал вимкнення лампочки – 2 с, інтервал ввімкнення лампочки – 1с.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripDropDownButton. В елемент ToolStripStatusLabel вивести значення поточного часу в короткому регіональному форматі.

Варіант 24

1. Написати програму, яка імітує стробоскоп. Блимання здійснює елемент управління TextBox. Частоту блимання повинен встановити користувач за допомогою елемента NumericUpDown.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripSplitButton. В елемент ToolStripStatusLabel вивести значення поточного часу в довгому регіональному форматі.

Варіант 25

1. Учасникам змагань надали номери від a до b. Написати програму, яка виводить номери учасників у багаторядкове текстове поле у зворотньому порядку. Числа a і b задає користувач за допомогою елементів управління NumericUpDown. Інтервал, за який виконується 1 виведення даних, дорівнює 0,2 с і повинен задаватися програмно.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripStatusLabel виводити почергово: 1) поточний час в годинах; 2) поточний день тижня. Інтервал зміни написів – 1 с.

Варіант 26

1. Написати програму, яка виводитиме в багаторядкове текстове поле усі непарні числа, що належать проміжку [a;b]. Числа a і b задаються користувачем за допомогою елемента управління NumericUpDown. Частота виведення чисел задається користувачем за допомогою елемента упарвління NumericUpDown.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripDropDownButton. В елемент ToolStripStatusLabel вивести почергово: 1) своє прізвище та ініціали; 2) свою групу. Значення повинні чергуватися з інтервалом 0,5 с.

Варіант 27

1. Написати програму, яка виводить в текстове поле по одному символи # від 1 до N з інтервалом G с. Число N задає користувач за допомогою NumericUpDown. Програма повинна перевіряти, щоб число N не перевищувало довжину текстового поля. Інтервал встановлює користувач за допомогою елемента управління NumericUpDown.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripSplitButton. В елемент ToolStripStatusLabel вивести почергово: 1) поточне значення дати в регіональном у форматі; 2) поточне значення часу в регіональному форматі. Ці два значення повинні чергуватися з інтервалом 1 с.

Варіант 28

1. Написати програму, яка виводить у багаторядкове текстове поле прямокутник символів * розмірами 3xN з інтервалом 0,7 с. Число N вводить користувач за допомогою елемента управління NumericUpDown.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripProgressBar. В елемент ToolStripStatusLabel вивести поточне значення секунд. Інтервал оновлення – 0,4 с.

Варіант 29

1. Написати програму, яка виводить у багаторядкове поле квадрат з символів # розмірами 4Nx4N з інтервалом K. Число N та інтервал K вводить користувач за допомогою елементів NumericUpDown.

2. Написати програму з рядком статусу, який включатиме такі елементи: ToolStripStatusLabel, ToolStripDropDownButton. В елемент ToolStripStatusLabel вивести поточне значення часу в регіональному форматі з інтервалом оновлення 0,4 с.

Варіант 30

1. Написати програму, яка запускає таймер зворотнього відліку (від 100 до 1) і виводить цей таймер в поле TextBox з інтервалом 1 с.

2. Написати програму з рядком статусу, який включатиме такі елементи: 2 ToolStripStatusLabel, ToolStripSplitButton. В елементи ToolStripStatusLabel вивести відповідно: 1) поточний рік; 2) поточний місяць.

Контрольні питання

1. Для чого використовується таймер?
2. Що являє собою таймер?
3. До якої групи елементів управління належить таймер?
4. Які основні властивості таймера?
5. Що означає властивість Interval?
6. Що означає властивість Modifiers?
7. Які основні методи таймера?
8. Як запускається таймер?
9. Як зупинити таймер?
10. Які основні події таймера?
11. Для чого призначений об´єкт DateTime?
12. Які простори імен слід підключати до програми для для роботи з датою і часом?
13. Яка властивість надає можливість отримати поточне значення дати і часу?
14. За допомогою якої властивості можна отримати потоний рік?
15. За допомогою якої властивості можна отримати поточний день місяця?
16. Для призначений рядок статусу?
17. Які основні компоненти рядка статусу?

Лабораторна робота №16

Тема: Статичні класи. Статичні поля. Статичні методи.
Мета: набути навичок створення статичних класів і методів,

Короткі теоретичні відомості

Статичний клас має такі ж самі властивості, що й нестатичний клас, крім однієї: екземпляри статичного класу створити не можна. Для створення змінної типу класу не можна використати оператор new. Доступ до членів статичного класу здійснюється за допомогою імені класу. Тому для виклика метода можна скористатись командою:

Class1.MyMethod();

Якщо метод повертає значення, то команда записується дещо по-іншому:

Int x=Class1.MyMethod(a, b);

Статичний клас можна використати як звичайний контейнер для набору методів, які функціонують з використанням лише вхідних параметрів. Ці методи не повинні використовувати екземпляри класів.

Статичний клас має такі властивості:

–неможливо створити екземпляр статичного класу;

–статичний клас повинен містити лише статичні члени;

–статичний клас може містити лише методи, які не потребують зберігання даних, які є унікальними для конкретного екземпляра класу;

–статичні класи запечатані, тому їх не можна наслідувати;

–статичний клас не може містити конструктор екземплярів.

Для застосування членів класу слід записати назву класу і назву метода.

```
static class Class1 {
public static double  g;//правильно
public int F;//неправильно – треба вказати модифікатор static
}
```

Приклад створення методу у статичному класі:

```
static class ArifmActions {
public static int Rez;
public int Plus (int a, int b) // вхідні параметри – цілі числа
{
ArifmActions.Rez=a+b;
}
public int Minus (int a, int b) // вхідні параметри – цілі числа
{
ArifmActions.Rez=a-b;
}
```

```
}
```

Для статичного класу не застосовується конструктор. Статичний клас можна застосовувати як контейнер для набору методів з використанням вхідних параметрів. Статичний клас не повинен повертати або встановлювати внутрішні поля екземпляра класу.

Статичний клас може містити статичний конструктор. Статичні конструктори мають такі відмінності від звичайних:

- статичні конструктори не повинні мати модифікатор доступу і не отримують параметрів;
- в статичних конструкторах не можна використовувати ключове слово this для посилання на поточний об’єкт класу;
- можна звертатися лише до статичних членів класу;
- статичні конструктори не можна викликати у програмі вручну. Вони виконуються автоматично під час першого створення об’єкта даного класу або при першому звертанні до його статичних членів.

Статичні конструктори звичайно застосовуються для ініціалізації статичних даних або виконують ті дії, які необхідно виконати лише одного разу.

Розглянемо приклад визначення статичного конструктора:

```
class Users {
   static Users()
   {
      Console.WriteLine("Створено першого користувача");
   }
}
class Program
{
   static void Main(string[] args)
   {
      Users user1 = new Users();
      Users user2 = new Users();
      Console.Read();
   }
}
```

Статичні члени

Нестатичний клас може містити статичні методи, поля, властивості і події. Статичний член викликається для класу навіть у тому випадку, коли не створено екземпляр класу. Доступ до статичного поля, методу, властивості або події здійснюється за іменем класу, а не екземпляра. Статичні методи і властивості не

можуть звертатися до статичних полів і подій у типі, що їх містить, не можуть також звертатися до змінної екземпляра об′єкта, якщо він не передається явно в параметрі метода.

Статичні поля нестатичного класу звичайно використовуються для збереження лічильника кількості створених об′єктів або зберігання значення, яке повинно використовуватися всіма екземплярами.

Статичні методи можуть бути перевантажені, але не перевизначені. Вони відносяться до класу, а не до екземпляра класу.

У мові C# не підтримуються статичні локальні змінні, тобто ті змінні, які оголошені всередині області дії метода.

Для оголошення статичних методів використовується ключове слово static. Наприклад:

```
public class Automobile    {
   public static int NumberOfWheels = 4;
   public static int SizeOfGasTank     {
      get
      {
         return 15;
      }
   }
   public static void Drive() { }
   public static event EventType RunOutOfGas;
}
```

Статичні члени ініціалізуються перед першим доступом до статичного члена і перед викликом статичного конструктора (якщо він є). Для доступу до члена статичного класу слід використовувати ім′я класу, а не ім′я екземпляра класу:

```
Automobile.Drive();
int i = Automobile.NumberOfWheels;
```

Якщо клас містить статичні поля, повинен бути вказаний статичний конструктор, який ініціалізує ці поля під час завантаження класу.

Виклик статичного метода генерує інструкцію виклику в проміжній мові Microsoft (MSIL). Виклик метода екземпляра генерує інструкцію callvirt, яка перевіряє наявність посилань на порожні об′єкти. Однак у більшості випадків різниця у продуктивності двох видів викликів несуттєва.

Завдання для виконання

Варіант 1

1. Написати програму для обчислення площі круга, трикутника (за трьома сторонами), прямокутника (за двома сторонами). Для цього створити статичний клас PL з такими методами: Krug (double R), Triangle (double a, double b, double c), Pr (double a, double b).

2. Написати програму для пошуку в рядку двох однакових слів. Для цього створити статичний клас Ryadok з єдиним методом Equvalent(string s), який приймає рядок, заданий користувачем, і повертає у програму рядок, з якого вилучені всі слова, які повторюються.

Варіант 2

1. Написати програму для підрахунку в рядку літер, які позначають голосні і приголосні звуки, а також пробіли. Для цього створити статичний клас Bukva з методами: Golos (string sss), Prigolos (string sss), Probil (string sss).

2. Написати програму для заміни третього і п´ятого слів у рядку випадковими символами. Для цього створити статичний клас RandSymbol(), який повертає у програму випадкову літеру англійського алфавіту.

Варіант 3

1. Написати програму для шифрування і дешифрування повідомлень за шифром Цезаря. Для цього створити статичний клас Cezar з методами toCode (string ss), unCode (string ss).

2. Написати програму для порівняння площ двох трикутників, заданих трьома сторонами. Для цього створити статичний клас Triangle з єдиним методом Square (double a, double b, double c), який приймає довжини сторін і повертає значення площі.

Варіант 4

1. Написати програму для обчислення площі поверхні таких геометричних тіл: прямокутного паралелепіпеда, кулі, циліндра і конуса. Для цього створити статичний клас Space з такими методами: Paral (double a, double b), Ball (double r), Cylinder (double r, double h), Conus (double r, double h).

2. Масиви містять дані про вартість автомобілів і ініціалізуються на початку програми. Написати програму для пошуку найбільшого значення масиву. Для цього створити статичний клас Automob з єдиним методом Max (int []vart), який приймає масив даних і повертає найбільше його значення.

Варіант 5

1. Написати програму для обчислення об′ємів таких тіл: куба, кулі, циліндра. Для цього створити статичний клас BodyVolume з такими методами: Kub, Ball, Cyl.

2. Написати програму для підрахунку кількості цифр, які трапляються у тексті. Для цього створити статичний клас Cifra з єдиним методом NumberOf(string ss), який приймає рядок і повертає кількість цифр.

Варіант 6

1. Написати програму для обчислення довжин векторів: 1) на площині; 2) в просторі. Для цього створити статичний клас MyVector з такими методами: 2D (на площині), 3D (у просторі).

2. Написати програму для підрахунку кількості великих літер у рядку. Для цього створити статичний клас Litera з єдиним методом Kilk(string ss), який приймає рядок і повертає кількість великих літер.

Варіант 7

1. Написати програму для кодування і декодування рядка за афінним шифром Цезаря. Для цього створити статичний клас Caesar з такими методами: Coding, Uncoding.

2. Учасникам змагань надали номери від x1 до x2 (ці числа вводить користувач в елементи NumericUpDown). Написати програму, яка виводить на екран середнє арифметичне цих чисел. Для цього створити статичний клас Chislo з єдиним методом SA(int x1,int x2), який приймає перший і останній номери і повертає середнє арифметичне.

Варіант 8

1. Написати програму для обчислення діагоналей квадрата і прямокутника. Для цього створити статичний клас Diagonal з такими методами: Kub, Prmk.

2. Написати програму, яка визначає, чи є слово симетричним. Для цього створити статичний клас Slovo з єдиним методом Simetr(string s), який приймає слово (рядок) і повертає 1 – якщо слово симетричне і 0 – у протилежному випадку.

Варіант 9

1. Написати програму для обчислення площ ромба і прямокутника. Для цього створити статичний клас PPPL з такими методами: Romb, Prm.

2. Написати програму, яка обчислює суму цифр числа, введеного в текстове поле. Для цього створити статичний клас DDigit з єдиним методом Suma(int N), який приймає число і повертає суму цифр.

Варіант 10

1. Написати програму для обчислення площі поверхні куба, прямокутного паралелепіпеда і піраміди. Для цього створити статичний клас з методами: Kub, Pram, Pyramid.

2. Написати програму для визначення взаємного положення точки A(x;y) і ромба (діагоналі d1, d2), центр якого знаходиться у точці (0;0). Для цього створити статичний клас ROMB з єдиним методом IsIn(x,y), який приймає координати точки і повертає 1 (точка належить ромбу) або 0 (точка знаходиться поза ромбом). Діагоналі ромба задаються в самому методі.

Варіант 11

1. Написати програму для обчислення площі поверхні кулі, циліндра і конуса. Для цього створити статичний клас Figure з методами: Ball, Cylinder, Conus.

2. Масив містить 12 випадкових чисел в межах від 0 до 100. Написати програму для визначення різниці між найбільшим і найменшим елементами масиву. Для цього створити статичний клас Masiv з такими методами: Min (), Max(). Обидва методи повинні приймати масив і повертати одне число.

Варіант 12

1. Написати програму для обчислення площі бічної поверхні прямокутного паралелепіпеда, циліндра, конуса. Для цього створити статичний клас JT7 з методами: Par, Cyl, Con.

2. Написати програму, яка видаляє з рядка всі символи, що повторюються. Для цього створити статичний клас SYM з єдиним методом – Away(), який приймає рядок, введений користувачем, і повертає рядок без символів, що повторюються.

Варіант 13

1. Написати програму для обчислення кінетичної і потенціальної енергії тіла. Для цього створити статичний клас Energy з методами: Potencial(), Kinetik(). Параметри підібрати самостійно.

2. Користувач вводить рядок символів довжиною від 7 до 20. Написати програму, яка виводить слова, утворені: 1) першою, третьою і сьомою літерами; 2) другою, четвертою і п´ятою літерами. Для цього створити статичний клас NewWord з методами Word1() і Word2(). Ці методи приймають рядок, введений користувачем, і виводять відповідно перше і друге слова.

Варіант 14

1. Написати програму для обчислення площі трикутника за кожною з трьох ознак. Для цього створити статичний клас Triangle з такими методами: First, Second, Third.

2. Користувач вводить рядок в текстове поле. Рядок необхідно обмежити кількістю N символів (число N вводиться користувачем за допомогою елемента NumericUpDown). Написати програму, яка виріже частину рядка, що виходить за межі, і виведе її у текстове поле. Для цього створити статичний клас SST з єдиним методом Entered(). Цей метод приймає рядок ss (введений користувачем) і число N (довжина рядка, який потрібен), і повертає залишок рядка.

Варіант 15

1. Написати програму для обчислення суми n перших членів арифметичної та геометричної прогресій. Для цього створити статичний клас SumaN з такими методами: Arifm, Geom.

2. Користувач вводить рядок символів у текстове поле. Написати програму, яка виводить у текстове поле лише літери f, які зустрічаються в рядку. Для цього створити статичний клас SSTR з єдиним методом TakeF(), який отримує рядок і повертає інший рядок, який містить усі символи f, які є в цьому рядку.

Варіант 16

1. Написати програму для знаходження відрізка найбільшої довжини. Для цього створити статичний клас Vidrizok з методом MaxLen. Вхідні параметри: координати двох точок на площині.

2. Користувач вводить кілька речень у текстове поле. Речення розділяються крапкою, знаком оклику або знаком питання. Написати програму, яка визначає кількість речень у введеному тексті. Для цього створити статичний клас SENT з єдиним методом Dots (), який приймає рядок, а повертає кількість речень.

Варіант 17

1. Написати програму для обчислення діагоналей ромба. Для цього створити статичний клас ROMB з такими методами: Diag(a,b,β), де a,b – сторони паралелограма, β – кут між цими сторонами*.

*Примітка. В C# кут виражається в радіанах, тому програма повинна перераховувати кут з градусів у радіани.

2. Користувач вводить речення в текстове поле. Написати програму, яка обчислює довжину найбільшого слова у тексті. Для цього створити статичний клас SENTENS з єдиним методом Longest(), який отримує рядок, а повертає довжину слова.

Варіант 18

1. Написати програму для обчислення суми всіх елементів масиву. Для цього створити статичний клас Masssiv з одним методом: Suma. Вхідні параметри: Arr – масив цілих чисел.

2. Користувач вводить речення в текстове поле. Написати програму, яка виводить довжину останнього слова. Для цього створити клас WWORD з єдиним методом Longest(), який отримує рядок, а виводить довжину слова.

Варіант 19

1. Написати програму, яка вилучає з рядка тексту найкоротше і найдовше слово. Для цього створити статичний клас Slovo з такими методами: Kor, Dov. Вхідні параметри: рядок.

2. Користувач вводить рядок символів у текстове поле. Написати програму, яка виводить кількість цифр, які зустрічаються в рядку. Для цього створити клас Digits з єдиним методом Number(), який отримує рядок, а повертає кількість цифр.

Варіант 20

1. Написати програму, яка записує рядок у зворотному напрямі. Для цього створити клас RD з таким методом: ReverseR. Вхідний параметр один: рядок.

2. Написати програму, яка видаляє з рядка всі літери j. Для цього створити статичний клас Litera з єдиним методом Removed(), який отримує рядок, введений користувачем, і повертає рядок без літер j.

Варіант 21

1. Написати програму для визначення простих чисел на проміжку від a до b. Числа a і b вводяться користувачем. Для визначення того, чи є слово простим, створити статичний клас SDigit з єдиним методом IsSimpe(). Створений метод приймає один параметр – число і повертає 1 (число просте) або 0 (число складне).

2. Користувач вводить рядок у текстове поле. Написати програму, яка видаляє з рядка фрагмент з індексами [a;b]. Для цього створити статичний клас Rtext з єдиним методом Take(), який приймає три аргументи: 1) рядок; 2) індекс першого символа; 3) інтекс останнього символа. Метод має повернути рядок, з якого видалено фрагмент.

Варіант 22

1. Написати програму для визначення найдовшого слова у рядку, введеному користувачем. Рядки розділяються пробілами. Для цього створити статичний клас MyWord з єдиним методом Longest(), який приймає один аргумент – рядок і повертає найбільше слово в рядку.

2. Користувач вводить рядок у текстове поле. Написати програму, яка заміняє символи в рядку. Для цього створити статичний клас TextFirst з єдиним методом Repl(), який отримує рядок, заміняє символ g на символ t і повертає рядок.

Варіант 23

1. Написати програму, яка вирізає з рядка всі слова, що повторюються. Для цього створити статичний клас Ryadok з єдиним методом Repeated(). Створений метод приймає один аргумент (рядок) і повертає рядок (з якого вилучені всі слова, що повторюються). Слова розділяються пробілами.

2. Користувач вводить речення у текстове поле. Написати програму, яка подвоїть символи q у рядку. Для цього створити статичний клас TextGR з єдиним методом Doubled (), який отримує рядок і повертає рядок з подвоєними символами q.

Варіант 24

1. Написати програму для розбиття рядка на масив слів. Для цього створити статичний клас RRR з єдиним методом Divide(). Цей метод приймає один аргумент (рядок) і повертає масив рядків (слів). Слова розділяються пробілами.

2. Написати програму для

Варіант 25

1. Написати програму для зміни порядку слів у реченні на протилежний. Для цього створити статичний клас Sentence1 з єдиним методом Mix(). Метод Mix() повинен приймати один аргумент – рядок з правильним порядком слів, і повертати рядок з оберненим порядком слів.

2. Користувач вводить рядок у текстове поле. Написати програму, яка подвоює всі символи рядка. Для цього створити статичний клас FieldG з єдиним методом Doubled(), який отримує рядок і повертає рядок з подвоєними символами.

Варіант 26

1. Написати програму для формування випадкового набору символів українського алфавіту. Для цього створити статичний клас Symb з єдиним методом Rndm(). Цей метод не приймає аргументів, а повертає випадковий символ українського алфавіту.

2. Користувач вводить рядок у текстове поле. Написати програму, яка виводить символи, що повторюються в тексті. Для цього створити статичний клас TextW з єдиним методом Doubled(), який отримує текст, введений користувачем, і повертає рядок з символами, що повторюються в тексті.

Варіант 27

1. Написати програму для формування випадкового набору символів англійського алфавіту. Для цього створити статичний клас Xst з єдиним методом Sm(), який повертає один випадковий символ англійського алфавіту.

2. Користувач вводить речення у текстове поле. Програма повинна вилучити з речення всі символи, що повторюються. Для цього створити статичний клас Symb з єдиним методом Remm(), який отримує рядок, введений користувачем, а повертає рядок, з якого вилучені символи.

Варіант 28

1. Написати програму для визначення взаємного положення точки і квадрата (зі стороною 12), центр якого знаходиться в початку координат. Координати точки вводить користувач. Для цього в програмі створити статичний клас Dots з єдиним методом IsIn(double x, double y). Цей метод приймає координати точки і повертає 0 (якщо точка поза квадратом), 1 (якщо точка знаходиться на лінії, що обмежує квадрат), 2 (якщо точка знаходиться всередині квадрата).

2. Користувач вводить речення в текстове поле. Програма повинна видалити з тексту символ, який повторюється найбільшу кількість разів. Для цього створити статичний клас TextV з єдиним методом Symb(). Даний метод повинен отримувати рядок, введений користувачем, і повертати рядок без символа, що повторюється максимальну кількість разів.

Варіант 29

1. Між двома точками, що мають координати відповідно (x1, y1) та (x2, y2), натягнуто нитку довжиною L. Написати програму, яка визначатиме, чи вистачить нитки. Для цього створити статичний клас TThread з єдиним методом: IsIsset(double x1, double y1, double x2, double y2, double L). Цей метод повинен повертати 0 (якщо не вистачає) або 1 (якщо вистачає).

2. Користувач вводить ціле число від 1 до 20. Написати програму, яка перетворює число з арабської системи числення в римську. Для цього створити статичний клас AllDigits з єдиним методом ToRomeSystem(), який отримує ціле число, а повертає рядок (запис числа римськими цирфами).

Варіант 30

1. Написати програму для порівняння середніх арифметичних двох масивів. Для цього створити статичний клас Masiv з єдиним методом SerArifm (int arr[]), який приймає масив і виводить середнє арифметичне.

2. Написати програму, яка зчитує з файла довжини сторін трикутників і порівнює площі цих трикутників. Для цього створити статичний клас Trian з єдиним методом PL(), який приймає три дійсних числа (довжини сторін трикутника) і повертає значення площі.

Контрольні питання

1. Які класи називаються статичними?
2. У чому полягає відмінність статичного класу від звичайного?
3. Які властивості статичного класу ви знаєте?
4. Які властивості статичних полів звичайного класу?
5. Для чого використовуються статичні поля?
6. Для чого служить модифікатор static?
7. Які поля можуть бути в статичного класі?
8. Чи можуть бути статичні поля у звичайному класі?
9. Як викликати метод статичного класу?
10. Як викликати статичний метод звичайного класу?
11. Чи можна створити екземпляр статичного класу?
12. Чи можна створити дочірній клас для статичного класу?

Лабораторна робота №17

Тема: Перевантаження методів

Мета: набути навичок створення і застосування перевантажених методів.

Короткі теоретичні відомості

Під час розробки програм часто виникає необхідність створити один і той же метод, але з різними вхідними параметрами. Відповідно до вхідних параметрів застосовується певна версія метода. Ця можливість називається перевантаженням методів (method overloading). Можна створити в класі кілька методів з однаковою назвою, але з різними вхідними параметрами. Тобто, з різною сигнатурою.

Сигнатура складається з кількох компонентів:

–ім′я метода;

–кількість параметрів;

–типи параметрів;

–порядок параметрів;

–модифікатори параметрів.

У сигнатуру не входять назви параметрів. Методи повинні відрізнятися за:

–кількістю параметрів;

–типу параметрів;

–порядку параметрів;

–модифікатори параметрів.

Наприклад:

```
int Mnoj (int a, int b) {
return (a*b);
}
int   Mnoj (int a, int b, int c) {
return (a*b*c);
}
double Mnoj (double a, double b) {
return (a*b);
}
```

У прикладі представлено різні версії метода, тобто визначені перевантаження даного метода.

Сигнатури методів можна представити як:

```
Mnoj (int, int)
Mnoj (int, int, int)
Mnoj (double, double)
```

Після визначення перевантажених версій можна використати їх у програмі:

```
void main () {
int Rez1=Mnoj (3,7);
int Rez2=Mnoj (5,9,11);
double Rez3=Mnoj (5.7, 2.15);
}
```

Перевантажені методи можуть відрізнятися за використаними модифікаторами. Наприклад:

```
int Mnoj (int a, int b) {
return (a*b);
}
int   Mnoj (ref int a, ref int b) {
return (a*b);
}
```

У даному випадку обидві версії метода INCR мають однаковий набір параметрів однакового типу, але в другому випадку параметри мають модифікатор ref. Тому обидві версії метода будуть коректними перевантаженнями метода INCR.

Відмінність методів за типом, що повертається, або імені параметра не є підставою для перевантаження. Наприклад:

```
int Mnoj (int a, int b) {
return (a*b);
}
int Mnoj (int x, int y) {
return (a*b);
}
void Mnoj (int x, int y) {
Console.WriteLine (x*y);
}
```

Сигнатура цих методів буде співпадати:

```
Mnoj (int, int)
```

Тому даний набір методів не являє собою коректні перевантаження метода Mnoj і функціонувати не буде.

Перевантаження методів – це одна з форм поліморфізму. Поліморфізм – це один з основних принципів об′єктно-орієнтованого програмування.

Отже, за допомогою перевантаження програміст отримує додаткові можливості під час створення методів. Проте іноді доцільніше створити новий метод зі своїм унікальним іменем.

Завдання для виконання

Варіант 1

1. Написати програму для обчислення периметра квадрата, прямокутника, трикутника. У програмі створити статичний клас FIGURES з перевантаженим методом PERIMETER, який отримує відповідно один, два або три параметра.

2. Написати програму для заміни всіх літер у слові, введеному користувачем, на пробіли або на символ, вказаний користувачем. Для цього створити перевантажений метод RePlace(), який приймає: 1) один аргумент (слово) і заміняє всі символи на пробіли; 2) два аргументи (слово і символ) і заміняє всі символи у слові на той символ, який передано в метод.

Варіант 2

1. Написати програму для обчислення площі квадрата, прямокутника, трикутника. У програмі створити статичний клас PLL з перевантаженим методом Square.

2. Написати програму для перемішування літер у рядку, введеному користувачем, причому міняються місцями символи, що мають парні і непарні індекси. Для цього створити статичний клас SYMB з перевантаженим методом ReverseSymbols(), який отримує: 1) 1 аргумент – рядок – і міняє місцями парні і непарні символи; 2) 2 аргументи – рядок і ціле число (довжина блоку) – розбиває рядок на блоки і міняє місцями блоки з парним і непарним індексом.

Варіант 3

1. Написати програму для обчислення периметра кола, паралелограма, чотирикутника (загального вигляду). У програмі створити статичний клас AllFigures з перевантаженим методом Perim.

2. Написати програму для обчислення діагоналі куба і паралелепіпеда. Для цього створити перевантажений метод DiagFigure(), який приймає: 1) один аргумент (довжину ребра куба); 2) 3 аргументи (довжини ребер паралелепіпеда). Метод повинен повертати дійсне число – довжину діагоналі.

Варіант 4

1. Написати програму для обчислення площі кола (один вхідний параметр), паралелограма (три вхідних параметра), ромба (два вхідних параметра). У програмі створити статичний клас PloskiFiguri з перевантаженим методом Prmt.

2. Написати програму для обчислення архімедової сили, що діє на тіло сферичної форми у різних рідинах. Для цього створити клас Archim з перевантаженим методом Force(), який має 2 варіанти: 1) з одним вхідним параметром (радіус кулі) – обчислює архімедову силу, що діє на тіло у воді; 2) з двома вхідними параметрами (радіус кулі і густина рідини) обчислює архімедову силу, що діє на тіло в іншій рідині, густина якої вводиться в якості аргументу). Метод повертає значення архімедової сили.

Варіант 5

1. Написати програму для обчислення об′єму кулі (один вхідний параметр), прямокутного паралелепіпеда (три вхідних параметра), циліндра (два вхідних параметра). У програмі створити статичний клас Volums з перевантаженим методом Volls, який повертає значення об´єму.

2. Написати програму, яка визначає найбільше з двох, трьох або чотирьох дійсних чисел. Для цього створити статичний клас Digits з перевантаженим методом LargestDigit(), який приймає відповідно два, три або чотири параметри.

Варіант 6

1. Написати програму для обчислення об′єму кулі (один вхідний параметр), циліндра (два вхідних параметра) і конуса (три вхідних параметра). Для цього створити статичний клас Class6 з перевантаженим методом FigureVolum.

2. Написати програму, яка буде визначати в масиві кількість чисел, заданих користувачем. Для цього створити статичний клас ObrMasiv з перевантаженим методом IsIn(), який має 2 варіанта: 1) з одним вхідним параметром (масив) – виводить кількість нульових значень у масиві; 2) з двома вхідними параметрами (масив і число) – підраховує кількість таких чисел у масиві.

Варіант 7

1. Написати програму для обчислення площі бічної поверхні циліндра (два вхідних параметра), прямокутного паралелепіпеда (три вхідних параметра) і куба (один вхідний параметр). Для цього створити статичний клас Squares7 з перевантаженим методом Side.

2. Написати програму для обчислення суми елементів масиву. Для цього створити статичний клас SumElements з перевантаженим методом Suma(), який має 2 версії: 1) з одним параметром (масив) – обчислює суму всіх елементів; 2) з двома параметрами (масив і перемикач) – обчислює суму додатних або від´ємних елементів (у залежності від значення параметра). Метод повертає дійсне число – значення суми.

Варіант 8

1. Написати програму для обчислення діагоналі квадрата (один вхідний параметр), прямокутника (два вхідних параметра). Для цього створити статичний клас Diags з перевантаженим методом Diagonal.

2. Написати програму для визначення найменшого з двох, трьох або чотирьох дійсних чисел. Для цього створити статичний клас MIN з перевантаженим методом MinDigit(), який має 3 варіанта, що приймають відповідно 2, 3 і 4 аргумента. Метод повертає найменше з цих чисел.

Варіант 9

1. Написати програму для обчислення довжини відрізка на площині і в просторі. Для цього створити статичний клас Linee з перевантаженим методом LEN, який має різну кількість параметрів.

2. Написати програму, яка обчислює відстань між двома точками на площині. Для цього створити статичний клас FFR з перевантаженим методом Lenn(), який має 2 версії: 1) приймає 2 параметри (координати x і y) точки і обчислює відстань між цією точкою і початком координат; 2) приймає 4 параметри (координати x і y обох точок) і повертає відстань між цими точками.

Варіант 10

1. Написати програму для обчислення скалярного добутку двох векторів, заданих двома різними способами: 1) через довжини векторів і кут між ними; 2) через координати обох векторів. Для цього створити статичний клас Vectors з перевантаженим методом Scalar().

2. Написати програму, яка визначає належність точки колу. Для цього створити статичний клас KOLO з перевантаженим методом IsIn(), який має 2 версії: 1) для кола з центром у початку координат – приймає 3 параметри: координати точки і радіус кола; 2) для кола, центр якого не лежить у центрі координат – приймає 5 параметрів: координати точки, радіус кола і координати центра кола.

Варіант 11

1. Шифр Цезаря полягає у тому, що кожен символ повідомлення зсувається вправо на 1 символ. Написати програму для шифрування повідомлення за шифром Цезаря. Для цього створити клас CaesarCode з методом Coding().. Для першої версії метода прийняти один вхідний параметр – рядок повідомлення, для другої версії – два вхідних параметра – рядок повідомлення і величину зсуву вправо символів.

2. Написати програму, яка буде визначати, чи є одне число сумою двох/ трьох інших. Для цього створити статичний клас SuMa з перевантаженим методом, який має 2 версії: 1) приймає 3 дійсних числа і перевіряє, чи є кожне з цих чисел сумою двох інших; 2) приймає 4 дійсних числа і перевіряє, чи є одне число сумою трьох інших.

Варіант 12

1. Написати програму для шифрування повідомлення переставним шифром. Для цього створити клас Coding з перевантаженим методом ToCode. Версії метода повинні відрізнятися різною кількістю вхідних параметрів. Перша версія: один вхідний параметр (рядок, який треба зашифрувати, а довжиною сегмента взяти 4). Друга версія: два вхідних параметра (рядок і довжина сегмента).

2. Написати програму для обчислення радіуса кола, описаного навколо прямокутника та квадрата. Для цього створити статичний клас Around з перевантаженим методом Round, який має 2 версії: 1) з одним вхідним параметром (сторона квадрата) – обчислює радіус кола, описаного навколо

квадрата; 2) з двома вхідними параметрами (сторони прямокутника) – обчислює радіус кола, описаного навколо прямокутника.

Варіант 13

1. Написати програму для обчислення площі трикутника. Для цього створити клас Triangle з перевантаженим методом TrSquare. Перша версія метода приймає три вхідні параметри (довжини сторін) і обчислює площу за формулою Герона. Друга версія приймає два вхідних параметри (довжини катетів прямокутного трикутника).

2. Написати програму для обчислення радіуса кола, вписаного у прямокутник та квадрат. Для цього створити статичний клас InRround з перевантаженим методом Kolo, який має 2 версії: 1) з одним вхідним параметром (сторона квадрата) – обчислює радіус кола, вписаного у квадрат; 2) з двома вхідними параметрами (сторони прямокутника) – обчислює радіус кола, вписаного у прямокутник.

Варіант 14

1. Написати програму для обчислення площі бічної поверхні циліндра і конуса. Для цього створити клас SideP з перевантаженим методом SSquare(). Використати для першої версії метода два вхідні параметри (діаметр і висоту циліндра), для другої версії метода – три вхідні параметри (діаметри обох основ зрізаного конуса, висоту конуса).

2. Написати програму для обчислення суми квадратів натуральних чисел від X до N. Для цього створити статичний клас Progress з перевантаженим методом Suma(), який має 2 версії: 1) один вхідний параметр (N) – обчислює суму квадратів натуральних чисел від 0 до N; 2) два вхідних параметри (X, N) – суму квадратів натуральних чисел на проміжку [X;N].

Варіант 15

1. Написати програму для обчислення маси тіла, що має форму прямокутного паралелепіпеда і куба. Для першої версії метода використати два параметри: густину тіла і довжину ребра куба. Для другої версії метода використати чотири параметри: густину тіла і довжини ребер паралелепіпеда.

2. Написати програму для обчислення кореня n-го степеня числа x. Для цього створити статичний клас RClass з перевантаженим методом, який має 2 версії: 1) один вхідний параметр (число X) – обчислює кубічний корінь числа; 2) два вхідних параметри (число X і показник степеня n) – обчислює корінь n-го степеня з числа.

Варіант 16

1. Написати програму для обчислення тиску стовпа рідини на дно посудини. Для цього слід створити клас Liquid і перевантажений метод Pressure. Для першої версії метода використати два вхідних параметри: густину рідини і висоту

стовпа. Для другої версії метода використати один параметр: висоту стовпа рідини (густину рідини взяти за замовчуванням такою, що дорівнює густині води).

2. Написати програму, яка обчислює відстань від прямої до початку координат. Для цього створити статичний клас Coordinat з перевантаженим методом Dist(), який має дві версії: 1) отримує 3 аргументи (коефіцієнти прямої A, B, C) і обчислює відстань від прямої на площині до початку координат; 2) отримує 4 аргументи (коефіцієнти прямої A, B, C, D) і обчислює відстань від прямої у просторі до початку координат.

Варіант 17

1. Написати програму для порівняння площ двох фігур, які можуть бути кулею і прямим циліндром. .Для цього створити клас Figure Squares з перевантаженим методом Square(), який має дві версії: 1) обчислює площу поверхні кулі і отримує один аргумент (радіус кулі); 2) обчислює площу повної поверхні прямого циліндра і отримує два аргументи (радіус циліндра і висоту циліндра).

2. Написати програму для обчислення висоти трикутника. Для цього створити статичний клас Trian з перевантаженим методом TrHeight(), який має три версії: 1) обчислює висоту рівностороннього трикутника і отримує один аргумент (сторону трикутника); 2) обчислює висоту рівнобедреного трикутника і отримує два аргументи (довжини бічної сторони і основи трикутника); 3) обчислює висоту трикутника у загальному вигляді і отримує три аргументи (сторони трикутника).

Варіант 18

1. Написати програму для порівняння об′ємів двох фігур, які можуть бути кубом і прямокутним паралелепіпедом.. Для цього створити статичний клас SolidBody з перевантаженим методом xVolume(), який має 2 версії: 1) отримує один аргумент (довжину ребра куба) – і обчислює об´єм куба; 2) отримує три аргументи (довжини ребер прямокутного паралелепіпеда) і обчислює об´єм прямокутного паралелепіпеда.

2. Написати програму, яка визначає найближче число, більше за задане, яке є паліндромом. Для цього створити статичний клас Palindrom з перевантаженим методом CloserDigit(), який має дві версії: 1) отримує один аргумент (число) і обчислює найближче більше число-паліндром з непарною кількістю цифр; 2) отримує два аргументи (число і додатковий параметр) і обчислює найближче більше число-паліндром з парною кількістю цифр.

Варіант 19

1. Написати програму для порівняння периметрів прямокутника, квадрата, трикутника і чотирикутника. Для цього створити статичний клас YYR з перевантаженим методом Perimetr()

2. Написати програму для обчислення радіусів кола і кулі, центр яких знаходиться в центрі координат. Для цього створити статичний клас CRadius з перевантаженим методом Radius(), який має дві версії: 1) отримує два дійсних числа – координати точки, яка лежить на колі, – і визначає радіус кола; 2) отримує три дійсних числа – координати точки, яка лежить на поверхні кулі, – і обчислює радіус кулі.

Варіант 20

1. Написати програму для обчислення довжини гіпотенузи прямокутного трикутника і рівнобедреного прямокутного трикутника. Для цього створити статичний клас Triangles з перевантаженим методом Hipotenuza(). Версії метода: 1) має один вхідний параметр – довжину катета рівнобедреного прямокутного трикутника; 2) має два вхідних параметри – довжини катетів прямокутного трикутника.

2. Написати програму для обчислення площі квадрата і прямокутника, центр якого співпадає з центром координат. Для цього створити статичний клас Coordinats з перевантаженим методом Ploscha(), який має дві версії: 1) обчислює площу квадрата і отримує один параметр – координату x правого верхнього кута квадрата; 2) обчислює площу прямокутника і отримує два параметри – координати x і y правої верхньої вершини прямокутника.

Варіант 21

1. Написати програму для обчислення третьої сторони трикутника за теоремою синусів і теоремою косинусів. Для цього створити статичний клас Triangl з перевантаженим методом Side3(). Версії метода: 1) 3 вхідні параметри – одна сторона (дійсне число) і два кути (цілі числа) – за теоремою синусів; 2) три вхідні параметри – дві сторони (дійсні числа) і протилежний кут (ціле число) – за теоремою косинусів.

2. Написати програму, яка буде визначати координати вершин квадрата і прямокутника за координатами правої верхньої вершини фігури. Для цього створити статичний клас Figure з перевантаженим методом Coordinats(), який повертає двовимірний масив координат x і y інших вершин фігури і має дві версії: 1) отримує один параметр – координату x і y правої верхньої вершини квадрата і визначає вершини квадрата; 2) отримує два параметри – координати x і y правої верхньої вершини прямокутника і визначає координати трьох інших вершин.

Варіант 22

1. Написати програму для обчислення об´ємів куба і прямокутного паралелепіпеда. Для цього створити статичний клас Volums з такими перевантаженим методом Vol(). Версії метода: 1) 1 вхідний параметр – довжина ребра куба; 2) 3 вхідні параметри – довжини ребер паралелепіпеда.

2. Написати програму для обчислення суми всіх елементів масиву. Для цього створити статичний клас Array7 з перевантаженим методом SumaEl(), який має дві версії: 1) для обчислення суми елементів масиву цілих чисел; 2) для обчислення суми елементів масиву дійсних чисел. В обох випадках метод отримує один параметр – масив чисел. Відрізняються лише результати: в першому випадку метод повертає ціле число, у другому – дійсне.

Варіант 23

1. Написати програму для обчислення діагоналі прямокутника і квадрата. Для цього створити статичний клас GeomFigures з перевантаженим методом Diagonal(). Версії метода: 1) 1 вхідний параметр – сторона квадрата; 2) 2 вхідних параметра – сторони прямокутника.

2. Написати програму для обчислення площі прямокутника, заданого координатами двох протилежних вершин. Сторони прямокутника паралельні осям координат. Для цього створити статичний клас Z2 з перевантаженим методом Square(), який має дві версії: 1) для прямокутника звичайного виду; 2) для прямокутника, центр якого співпадає з центром координат.

Варіант 24

1. Написати програму для обчислення довжини вектора на площині та в просторі. Для цього створити статичний клас MyVector з перевантаженим методом Length(). Версії метода: 1) 4 вхідних параметри (дійсні числа) – координати x,y початку і кінця вектора на площині; 2) 6 вхідних параметрів (дійсні числа) – координати x,y,z початку і кінця вектора в просторі.

2. Написати програму, яка визначає найбільше з двох або трьох чисел. Для цього створити статичний клас MaxDigit з перевантаженим методом MaxOf(), який має дві версії: 1) з двома вхідними параметрами (дійсні числа); 3) з трьома вхідними параметрами (дійсні числа).

Варіант 25

1. Написати програму для обчислення кутового коефіцієнту прямої на площині: 1) загального виду; 2) яка проходить через початок координат. Для цього створити статичний клас iLine з перевантаженим методом getCoef() з таким набором аргументів: 1) x1, y1, x2, y2 (пряма не проходить через центр координат); 2) x1, y1 (пряма проходить через центр координат).

2. Написати програму для порівняння висот трикутників. Для цього створити статичний клас Triangle з перевантаженим методом Heit(), який має три версії: 1) для обчислення висоти рівностороннього трикутника (один вхідний

параметр); 2) для обчислення висоти рівнобедреного трикутника (два вхідних параметри – бічна сторона і основа); 3) для обчислення висоти трикутника загального виду (три вхідних параметра – сторони трикутника).

Варіант 26

1. Написати програму для обчислення кількості пробілів або інших символів у рядку. Для цього створити статичний клас RD з перевантаженим методом SymbolCount(), який приймає: 1) 1 аргумент (рядок) і підраховує кількість пробілів; 2) 2 аргументи (рядок і символ) і підраховує кількість заданих символів у заданому рядку.

2. Написати програму, яка визначає, у скільки разів найбільше число більше за найменше. Для цього створити статичний клас Cifra з перевантаженим методом HowMuch(), який має дві версії: 1) отримує 2 цілих числа і визначає, у скільки разів більше число перевищує менше; 2) отримує 3 цілих числа і визначає, у скільки разів найбільше число перевищує найменше.

Варіант 27

1. Написати програму для шифрування повідомлень шифром Цезаря. Для цього створити статичний клас Coder з перевантаженим методом toCode(): 1) приймає 1 аргумент (рядок) і здійснює зсув символів вправо на 1 символ; 2) приймає 2 аргументи (рядок і ціле число x) і здійснює зсув на x символів вправо.

2. Написати програму для обчислення коренів лінійного та квадратного рівняння. Для цього створити статичний клас Rivn з перевантаженим методом Korin(), який має дві версії: 1) отримує коефіцієнти k, b лінійного рівняння і повертає його корінь; 2) отримує коефіцієнти a, b, c квадратного рівняння і повертає масив з 2 дійсних чисел, які є коренями цього рівняння.

Варіант 28

1. Написати програму для визначення діагоналі прямокутника і прямокутного паралелепіпеда. Для цього створити статичний клас DG з перевантаженим методом Diag(), який приймає: 1) 2 аргументи (сторони прямокутника a і b) і обчислює діагональ прямокутника; 2) 3 аргументи (ребра прямокутного паралелепіпеда a, b,c) і обчислює діагональ прямокутного паралелепіпеда.

2. Написати програму для порівняння площ двох фігур – ромба і паралелограма. Для цього створити статичний клас Figures з перевантаженим методом Square(), який має 2 версії: 1) отримує 2 параметри (довжини діагоналей) і обчислює площу ромба; 2) отримує 3 параметри (довжини діагоналей і кут між ними) і обчислює площу паралелограма.

Варіант 29

1. Написати програму для обчислення площі ромба та паралелограма. Для цього створити статичний клас SQP з перевантаженим методом Square, який: 1)

приймає 2 аргументи (довжину сторони і кут між ними) і обчислює площу ромба; 2) приймає 3 аргументи (довжини двох сторін і кут між ними) і обчислює площу паралелограма.

2. Написати програму, яка обчислює найменше спільне кратне двох або трьох чисел. Для цього створити статичний клас NSK з перевантаженим методом Kratne(), який має 2 версії: 1) отримує 2 параметри (цілі числа) і повертає НСК; 2) отримує 3 параметри (цілі числа) і повертає НСК.

Варіант 30

1. Написати програму для обчислення об´ємів куба та паралелепіпеда. Для цього створити статичний клас GeomFigure з методом перевантаженим Volum, який має дві версії: 1) з одним параметром L (довжина ребра куба); 2) з трьома параметрами a,b,c (довжини ребер паралелепіпеда).

2. Написати програму для визначення найбільшого спільного дільника (НСД) двох або трьох чисел. Для цього створити статичний клас NSD з перевантаженим методом Diln(), який має 2 версії: 1) отримує 2 аргумента (цілі числа) і повертає НСД цих двох чисел; 2) отримує 3 аргумента (цілі числа) і повертає НСД цих трьох чисел.

Контрольні питання

1. Які основні принципи об′єктно-орієнтованого програмування?
2. У чому полягає принцип поліморфізму?
3. Що таке метод?
4. Який метод називається статичним?
5. Для чого ставити модифікатор public перед іменем метода?
6. Що таке перевантаження методів?
7. Наведіть приклади перевантажених методів з числа стандартних класів C#.
8. Що таке сигнатура?
9. За яких умов здійснюється коректне перевантаження методів?
10. У яких випадках доцільно застосувати перевантажений метод?
11. Чи завжди доцільно застосовувати перевантажений метод?
12. У чому полягає принцип наслідування?
13. Який клас називається статичним?
14. У чому полягає відмінність статичного класу від звичайного?

Лабораторна робота №18

Тема: Наслідування (Спадкування)

Мета: набути навичок створення класів і похідних класів.

Короткі теоретичні відомості

Успадкування є однією з головних особливостей об´єктно-орієнтованого програмування. Завдяки цій властивості один клас може успадкувати функціональність іншого класу. Головний, базовий клас, з якого починається визначення класів, називається батьківським (суперкласом). Клас, похідний від батьківського, називається успадкованим (похідним, підкласом).

Похідний клас звичайно успадковує всі поля (властивості) батьківського класу, за винятком конструктора класу. Розглянемо приклад.

Приклад 1. Написати програму, яка містить батьківський клас «Трикутник» і похідний клас «Прямокутний трикутник». Опишемо батьківський клас:

```
class Triangle{
      public double a;
      public double b;
      public double c;
      public Triangle(double aa, double bb, double cc) {
         this.a=aa;
         this.b=bb;
         this.c=cc;
      }
      public Triangle() {
         this.a=0;
         this.b=0;
         this.c=0;
      }
   }
```

Опис дочірнього класу дещо відрізняється від опису батьківського класу. Після службових слів слід вказати через двокрапку батьківський клас:

```
class PrTriangle: Triangle {
```

Клас PrTriangle успадковує всі властивості і методи батьківського класу. Єдине, що не передається під час наслідування – конструктор класу. Тому конструктор слід описати окремо.

Опишемо дочірній клас:

```
class PrTriangle: Triangle {
      public PrTriangle (){
         this.a=0;
         this.b=0;
```

```
        this.c=0;
      }
      public PrTriangle(double aa, double bb) {
        this.a=aa;
        this.b=bb;
        this.c=Math.Sqrt(Math.Pow(aa,2)+Math.Pow(bb,2));
      }
   }
```

У конструкторі дочірнього класу дві сторони задає користувач, а третю сторону створює конструктор.

Перевіримо, чи правильно функціонують конструктори класів. Це можна зробити за допомогою такого коду:

```
public static void Main(string[] args)
      {
        Console.WriteLine("Hello World!");
        Triangle TT=new Triangle();
        Triangle DD=new Triangle(2,3,4);
        PrTriangle PP=new PrTriangle(3,4);
        Console.WriteLine(" порожній конструктор: c="+TT.c);
        Console.WriteLine("конструктор з параметрами:  c="+DD.c);
        Console.WriteLine("прямокутний трикутник:     c="+PP.c);
        Console.Write("Press any key to continue . . . ");
        Console.ReadKey(true);
      }
```

У мові C# усі класи успадковуються від базового класу Object. Тому створені нами класи Triangle і PrTriangle, крім власних методів, отримують також методи ToString(), Equals(), GetHashCode(), GetType() (вони успадковуються від класу Object).

Звичайно усі класи можуть бути успадковані. Але існують певні обмеження:

–клас може спадкуватися від одного класу (від кількох класів успадкувати клас не можна);

–під час створення підкласу слід враховувати тип доступу до базового класу. Успадкований клас повинен мати такий самий або більш строгий тип доступу. Якщо базовий і успадкований клас знаходяться в різних збірках, то похідний клас може спадкувати лише від класу, що має доступ public;

–якщо батьківський клас має модифікатор sealed, то від цього класу не можна успадкувати і створювати похідні класи;

–не можна успадкувати клас від статичного класу.

Доступ до членів базового класу з клас-нащадок може мати лише у тому випадку, коли ці члени мають модифікатори доступу private protected (якщо базовий і похідний клас знаходяться в одній зборці), public, internal (якщо базовий і похідний класи знаходяться в одній збірці), protected, protected internal.

Для звертання до базового класу служить ключове слово base. Наприклад, створення конструктора дочірнього класу має такий вигляд:

```
class Worker {
      string Name;
      public Worker(){
         this.Name="";
      }
      public Worker(string ss) {
         this.Name=ss;
      }
      public void getName() {
         Console.WriteLine(this.Name);
      }
   }

class Librarian:Worker{
      public string Library;
      public Librarian(string ss, string ss1)
         :base(ss)       {
         this.Library=ss1;
      }
      public Librarian()
         :base(){
         this.Library="";
      }
   }
```

Клас Worker має конструктор, який встановлює властивість Name. Оскільки клас Librarian успадковує і встановлює ту ж саму властивість Name, то доцільно не переписувати конструктор, а викликати конструктор базового класу Worker. Може трапитися так, що в конструкторі дочірнього класу може бути більше параметрів, ніж у конструкторі базового класу.

До базового класу можна звернутися за допомогою ключового слова base. У даному випадку в конструкторі класу Librarian слід встановити ім´я і компанію. Ім´я передається в конструктор базового класу (конструктор класу Worker) за допомогою команди base (name):

```
static void Main(string[] args)
{
   Worker w1 = new Worker("Petro");
   w1.getName();
   Librarian L1 = new Librarian ("Vasyl", "LuxSoft");
   L1.getName();
   Console.Read();
}
```

Конструктори в дочірніх класах

Конструктори не передаються похідному класу при успадкуванні. Якщо в базовому класі не визначено конструктор за замовчуванням без параметрів, а лише конструктор з параметрами, то в похідному класі необхідно викликати один з цих конструкторів за допомогою ключового слова base.

Порядок виклику конструктора

Під час виклику конструктора спочатку відпрацьовують конструктори базових класів, а лише після цього – конструктори похідних класів. Розглянемо наступні класи:

```
class Worker {
      string Name;
      int Age;
      public Worker(){
         this.Name="";
         this.Age=0;
      }
      public Worker(string ss) {
         this.Name=ss;
         this.Age=0;
      }
      public Worker(string ss, int age) {
         this.Name=ss;
         this.Age=age;
      }
      public Worker(int age) {
         this.Name="";
         this.Age=age;
      }
      public void getName() {
         Console.WriteLine(this.Name);
      }
```

```
}

class Librarian:Worker{
   public string Library;
   public Librarian(string ss, int age, string ss1)
      :base(ss)       {
      this.Library=ss1;
   }
   public Librarian()
      :base(){
      this.Library="";

   }
   public Librarian(string ss, int age)
      :base(){
      //this.Library="";

   }
   public Librarian(string ss, string ss1)
      :base(){
      this.Library=ss1;

   }
}
```

Під час створення об´єкта Librarian:

```
Librarian L1 = new Librarian ("Vasyl",18, "LuxSoft");
```

У результаті буде отримано таку послідовність виконання:

1. Спочатку викликається конструктор Employee (string name, int age, string company). Він делегує виконання конструктору Person(string name, int age).
2. Викликаєтсья конструктор Person(string name, int age), який сам не виконується, а передає виконання конструктору Person(string name).
3. Викликається конструктор Person(string name), який передає виконання конструктору класу System.Object, оскільки це базовий клас для Person (за замовчуванням).
4. Виконується конструктор System.Object.Object(), потім виконання повертається конструктору Person(string name).
5. Виконується тіло конструктора Person(string name), потім виконання повертається конструктору Person (string name, int age).

6. Виконується тіло конструктора Person (string name, int age), потім виконання повертається конструктору Employee(string name, int age, string company).
7. Виконується тіло конструктора Employee(string name, int age, string company). У результаті створюється об´єкт Employee.

Мова C# і .NET підтримують лише одинарне успадкування. Це означає, що кожен клас може успадкувати члени лише одного класу. Водночас C# підтримує транзитивне успадкування, коли можна визначити ієрархію успадкування для набору типів.

Похідний клас не успадковує:

–статичні конструктори, які ініціалізують статичні дані класу;

–конструктори екземплярів, які викликаються для створення нового екземпляра класу. Кожен клас повинен визначати власні конструктори;

–методи завершення, які викликаються збиральником сміття середовища виконання для знищення екземплярів класу.

Усі інші члени базового класу успадковуються похідними класами, але їх видимість не залежить від доступності. Доступність членів впливає на видимість для похідних класів таким чином:

- закриті члени я видимими лише у похідних класах, які вкладені у базовий клас. Для інших похідних класів вони невидимі.
- захищені члени є видимими лише у похідних класах;
- внутрішні члени є видимими лише в похідних класах, які знаходяться у тій же збірці, що й базовий клас. Вони не будуть видимими у похідних класах, які розміщені в інших збірках;
- відкриті члени є видимими у довільних класах, а також входять до загальнодоступного інтерфейсу похідних класів. Успадковані відкриті члени можна викликати так само, ніби вони були визначені у самому похідному класі.
- Похідні класи можуть також перевизначати успадковані члени, тобто представляти альтернативну реалізацію. Перевизначити можна лише ті члени, які у базовому класі відмічені ключовим словом virtual (віртуальний). За замовчуванням не можна перевизначати члени базового класу, які не відмічені ключовим словом virtual. Спроба перевизначити член, який не є віртуальним,викликає помилку компілятора.
- У деяких випадках похідний клас повинен перевизначати реалізацію базового класу. Члени базового класу, які відмічені ключовим словом abstract, обов´язково повинні перевизначатися у похідних класах.

- Успадкування застосовується лише для класів та інтерфейсів. Інші категорії типів (структури, делегети і переліки) не підтримують наслідування. Через це виникає помилка компіляції.
- Неявне успадкування

–Всі типи у системі типів .NET неявно успадковуються від типу Object або його похідного типу. Спільні функції Object доступні будь-якому типу.

Неявне спадкування від класу Object робить доступними для класу SClass такі методи:

–відкритий метод ToString(), який перетворює об´єкт SClass у рядок, повертає повне ім´я типу;

–три методи, які перевіряють рівність двох об´єктів: відкритий метод екземпляра Equals(Object), відкритий статичний метод Equals(Object, Object) і відкритий статичний метод ReferenceEquals(Object, Object). За замовчуванням ці методи перевіряють рівність посилань;

–відкритий метод GetHashCode(), який обчислює значення, яке дозволяє використати екземпляр типу в хешованих колекціях;

–відкритий метод GetType(), який повертає об´єкт Type, який представляє тип SClass;

–захищений метод Finalize(), який повинен звільняти некеровані ресурси перед тим, як збиральник сміття звільнить пам´ять об´єкта;

–захищений метод MemberwiseClone(), який створює неповну копію поточного об´єкта.

У таблиці … вказано категорії типів, які можна створювати на мові C#, і вказано типи, від яких вони явно успадковують.

Таблиця

Неявне успадкування

Категорія типу	Неявно успадковує від
class	Object
struct	ValueType, Object
enum	Enum, ValueType, Object
delegate	MulticastDelegate, Delegate, Object

Успадкування і зв´язок «є»

Звичайно успадкування виражене зв´язком типу “is a” (є) між базовим класом і одним або кількома похідними класами. Похідні класи розглядаються як спеціалізовані версії базового класу, тобто як підтипи базового класу.

Постановка завдання

1. Розробити програму, в якій створюються базовий і похідний класи.
2. Розробити програму, в якій використовуються розроблені класи.

Варіанти для виконання

Варіант 1

1. Створити базовий клас Ridyna з такими полями: name (ім´я), ro (густина) і конструкторами класу (порожнім і непорожнім). Створити похідний клас Petrol з додатковим полем q (питома теплота згоряння). У програмі створити екземпляри обох класів і продемонструвати використання властивостей і конструктора.
2. Створити масиви з 12 об´єктів кожного класу, поля яких зчитуються з файлів input1.txt і input2.txt. Вивести на екран властивості найважчої рідини (RIDYNA) і палива (PETROL) з найвищою теплотою згоряння.

Варіант 2

1. Створити базовий клас Car з такими полями: name (ім´я), speed (швидкість), masa (маса) і конструкторами екземплярів. Створити похідний клас Vaen з додатковим полем wgt (вантажопідйомність). Створити масиви об´єктів кожного класу.
2. Створити масив з 10 об´єктів класу Car і масив з 5 об´єктів класу Vaen. Зчитати з файла income1.txt властивості 10 об´єктів масиву об´єктів Car, а з йфайла income2.txt – властивості об´єктів класу Vaen. Вивести на екран усі властивості у стовпчик.

Варіант 3

1. Створити базовий клас PP (прямокутний паралелепіпед) з такими полями: a,b,c (розміри), V (об´єм, обчислювана властивість). Створити конструктори класу і метод, який виводить на екран розміри прямокутного паралелепіпеда. Створити дочірній клас KUB, для якого передбачити однакові лінійні розміри.
2. Створити масив з 7 об´єктів класу PP, всі поля якого зчитуються з файла pp.txt і вивести на екран розміри прямокутного паралелепіпеда, об´єм якого найбільший. Створити об´єкт класу KUB, розміри якого зчитати з файла kub.txt, і вивести на екран його об´єм.

Варіант 4

1. Створити базовий клас Mebli з такими полями: name, masa, height, а також конструктори класу. Створити похідні класи Chair і Table, для яких передбачити додаткові поля h (висота сидіння, клас Chair), h, a і b (розміри стільниці, клас Table), S (площа стільниці, клас Table). Для класу Table створити метод, який обчислює площу стільниці.
2. Створити масив з 5 об´єктів класу Chair, розміри яких зчитати з файла one.txt, і масив з 3 об´єктів класу Table, розміри яких зчитати з файла two.txt. Вивести на екран стілець з найменшою висотою і стіл з найбільшою площею стільниці.

Варіант 5

1. Створити базовий клас Fruits з такими полями: name, color, masa, diam і конструкторами класу. Створити похідний клас Apples, який успадковує всі поля батьківського класу і має додаткове поле sort (сорт яблук).
2. Створити масив з 9 об´єктів класу Apples, усі поля об´єктів якого зчитати з файла start.txt. Вивести на екран усі поля яблук, які мають білий колір.

Варіант 6

1. Створити базовий клас Vegetables з такими полями: name, masa, color і конструкторами класу. Створити похідний клас Potatoes, який успадковує властивості батьківського класу і має додаткову властивість diameter (діаметр). Перевірити функціональність конструкторів класу для обох класів.
2. Створити два масиви об´єктів –для кожного класу. Кожен масив повинен налічувати 3 об´єкти. Під час створення об´єктів їх поля вводити з клавіатури і записувати в два окремих файла.

Варіант 7

1. Створити базовий клас PR (прямокутник) з такими полями: a, b (розміри), S (площа, обчислювана властивість). Для цього класу створити порожній і непорожній конструктори, статичний метод Diag (діагональ прямокутника). Створити дочірній клас Kvadrat, який має власні конструктори і успадковує властивості батьківського класу.
2. Написати програму, яка зчитує розміри прямокутника і квадрата з двох різних файлів і виводить розміри тієї фігури, площа якої більша.

Варіант 8

1. Створити базовий клас PUBS з такими полями: author, year, name і конструкторами класів. Створити похідні класи MAGAZIN і BOOK, які мають всі властивості батьківського класу, а крім того, клас MAGAZIN має поля number і month, а клас BOOK має поля pages, publisher. Створити для кожного похідного класу конструктори.
2. Створити два масиви з 10 об´єктів кожен – класів MAGAZIN і BOOK. Усі поля зчитати з 2 різних файлів. Написати програму, яка виводить на екран усі властивості журналів і книжок, які вийшли у 2020 році.

Варіант 9

1. Створити базовий клас GAME з такими полями: name, year, os (операційна система), а також з конструкторами класу. Створити похідні класи STRATEGY і ARCAD з конструкторами класів. Всі поля батьківського класу повинні успадкуватися похідними класами. Крім того, клас STRATEGY має додаткове поле developer (розробник програми), а клас

ARCAD має додаткові поля levels (кількість рівнів) і screen (роздільна здатність монітора).

2. Написати програму, яка створює два масиви ігор класів STRATEGY і ARCAD. Програма повинна зчитувати в один масив значення полів класу ARCAD з файла input1.txt, а в інший масив – поля класу STRATEGY з файла input2.txt. Кожен масив повинен містити по 10 екземплярів і виводитися на екран.

Варіант 10

1. Створити базовий клас SHIP з такими полями: name, deadweight (дедвейт), owner (фірма-власник), country (країна-власник) і конструкторами класу. Створити похідні класи TRAMP (суховантажне судно) і OILER (танкер), які наслідують всі поля і методи батьківського класу. Для класу TRAMP створити додаткове поле typ (тип вартажу), для класу OILER – додаткові поля speed (швидкість) і leng (довжина).
2. Написати програму, яка створює два масиви (по 6 екземплярів) класів TRAMP і OILER. Всі поля цих класів зчитуються з файлів iks1.txt і iks2.txt. Програма повинна порівняти найбільші за полем deadweight екземпляри класів TRAMP і OILER.

Варіант 11

1. Створити базовий клас AIR з такими полями: name, speed, masa і конструкторами класу. Створити похідні класи PLANE і HELICOPTER, які успадковують усі поля і методи базового класу. Для класу PLANE створити додаткові поля square (площа крил) і len (довжина пробігу). Для класу HELICOPTER створити додаткове поле capacity (вантажопідйомність).
2. Програма повинна створити два масиви (типів PLANE і HELICOPTER), поля яких зчитуються відповідно з файлів air1.txt і air2.txt, обчислити середні маси літаків і вертольотів.

Варіант 12

1. Створити базовий клас Trikutnyk з полями-сторонами. Створити конструктори класу, методи зміни полів, обчислення периметра. Створити похідний клас Rivn (рівносторонній трикутник), який має всі поля батьківського класу і поле площі. Створити метод обчислення площі і конструктори класів.
2. Вхідні дані для масиву об´єктів типу Rivn містяться у файлі file1.txt. Написати програму, яка обчислить сумарну площу усіх трикутників.

Варіант 13

1. Створити базовий клас TR (трикутник) з полями-сторонами. Створити методи зміни сторін, обчислення кутів, обчислення периметра. Створити

похідний клас PTR (прямокутний трикутник), який має додаткове поле площі. Написати метод визначення площі для похідного класу і конструктори класу для нього (з перевіркою можливості існування прямокутного трикутника).

2. У файл solution.txt записано значення полів 12 об´єктів типу PTR. Написати програму, яка зчитує дані з файла і виводить дані прямокутного трикутника, у якого найбільша площа.

Варіант 14

1. Створити базовий клас BIKE з такими полями: marka, speed, color і конструкторами класу. Створити методи зміни кольору і швидкості. Створити похідний клас LittleBike, який успадковує всі поля і конструктори класу, і має додаткові поля height і length (висота і довжина). Створити методи зміни висоти і довжини.
2. Файл input.txt містить значення полів 9 об´єктів класу LittleBike. Програма повинна зчитати ці поля, створити масив об´єктів класу LittleBike, обчислити середню довжину елементів масиву.

Варіант 15

1. Створити базовий клас ABC (алфавіт) з такими полями: name (назва), number (кількість символів), country (країна). Створити конструктори класу, методи зміни кількості символів і назви країни. Створити похідний клас Latin, який успадковує всі поля і методи базового класу, а також має додаткові поля year і language.
2. Файл income.txt містить значення полів для 8 об´єктів класу Latin. Написати програму, яка виведе на екран усі дані алфавітів, які містять більше 26 літер.

Варіант 16

1. Створити базовий клас SHIP з такими полями: name, deadweight, capacity, speed, а також конструктори класу, методи зміни полів класу і методи виведення значень полів. Створити похідні класи SUBMARINE і CRAFT. Для кожного з похідних класів створити конструктори, а також методи виведення значень полів.
2. Файл craft.txt містить значення полів 7 екземплярів класу CRAFT. Написати програму, яка виводить на екран значення полів speed усіх об´єктів класу CRAFT.

Варіант 17

1. Створити базовий клас CITY з такими полями: name, square, population, а також конструктори класу, методи, які виводять на екран значення полів (поля повинні мати модифікатор доступу public). Створити похідний клас TOWN з такими додатковими полями: region (область), district (район),

конструктори класу, методи, що виводять значення полів region, district, а також методи, які змінюють значення цих полів.

2. У файлі towns.txt містяться властивості 10 міст. Написати програму, яка виведе на екран властивості найбільшого за населенням міста.

Варіант 18

1. Створити базовий клас QUADRANGLE з такими полями: name, a1,a2,a3,a4 (поля мають модифікатор доступу public), а також конструктори класу і методи, що виводять значення полів. Створити похідний клас QUDRAT, який має додаткові поля square і perimeter, конструктори класу і додаткові методи, які виводять значення полів square і perimeter.
2. Файл input.txt містить властивості 11 квадратів. Написати програму, яка виведе на екран властивості квадрата, найменшого за площею.

Варіант 19

1. Створити базовий клас QUAD з такими полями: a1,a2,a3,a4 (сторони чотирикутника), fi1, fi2, fi3, fi4 (кути чотирикутника) (поля мають модифікатор доступу public), а також конструктори класу і методи, що виводять значення полів. Створити похідний клас PRM (прямокутник) з додатковими полями square і perimeter.
2. У файлі exist.txt містяться властивості 7 прямокутників. Програма повинна обчислити середню площу і середній периметр цих прямокутників.

Варіант 20

1. Створити базовий клас ВАК (бак) з такими полями: name, square, volume, конструкторами класу і методами, які виводять значення полів на екран. Створити похідний клас ROUND_BAK (циліндричний бак) з додатковими полями diameter і height, конструктори класу і методи, що виводять значення полів на екран і змінюють значення полів. Поля повинні мати модифікатор доступу private.
2. У файлі input.txt містяться властивості 6 об´єктів класу ROUND_BAK. Написати програму, яка виведе на екран властивості бака, найбільшого за об´ємом.

Варіант 21

1. Створити базовий клас Computer з такими полями: model, processor, harddisk, конструкторами класу, методами, які змінюють поля (крім моделі). Створити дочірні класи Netbook і Laptop, для яких встановити додаткові поля: screen_size,akumulator, а також конструктори класів.
2. Файл netbooks.txt містить властивості 5 нетбуків. Написати програму, яка виведе на екран назви усіх нетбуків, розмір екрану яких менше за 15”.

Варіант 22

1. Створити базовий клас Worker з такими полями: name, age, profession, конструкторами класу. Створити дочірній клас Translator з додатковими полями work_stage, language, методами, які змінюють стаж і мову, а також конструктори класу.
2. Файл input.txt містить властивості 12 перекладачів. Написати програму, яка виведе на екран вік усіх перекладачів, які спеціалізуютсья на англійській мові.

Варіант 23

1. Створити базовий клас BOOKS з такими полями:name, year, pages; з конуструкторами і методами для зміни полів. Створити дочірній клас Comixes з додатковими полями author і publisher, методами для зміни цих полів і конструкторами.
2. Файл comix.txt містить властивості 7 коміксів. Написати програму, яка виведе на екран усі властивості 3 найновіших коміксів.

Варіант 24

1. Створити базовий клас Bike з такими полями:name, weight, speed; з конструкторами класу і методами, що змінюють поля (крім назви). Створити дочірній клас Child_Bike з додатковими полями for_age, color, методами для зміни цих полів і конструкторами класу.
2. Файл child.txt містить значення властивостей 8 екземплярів об´єктів класу Child_Bike. Написати програму, яка виводить на екран властивості найлегшого велосипеда.

Варіант 25

1. Створити базовий клас FIELD з такими полями: name, length, beam, square, perimeter, owner, number, конструкторами класу (поля square і perimeter обчислюються під час створення нового екземпляра класу), і методами, що змінюють поля owner і number. Створити дочірній клас HAYFIELD з додатковим полем grass, конструкторами і методами, що виводять значення полів.
2. Файл input.txt містить значення властивостей 5 лугів (клас HAYFIELD). Написати програму, яка обчислить сумарний периметр цих лугів.

Варіант 26

1. Створити базовий клас MOUNTAIN з такими полями: name, height, country, з конструкторами класу і методами, які змінюють поля name і country. Створити дочірній клас HILL з додатковим полем owner (власник), конструкторами класу і методом, що змінює власника.
2. Файл hills.txt містить властивості 10 об´єктів класу HILL. Написати програму, яка виведе на екран значення усіх полів цих об´єктів,

попередньо відсортувавши їх за висотою (за зростанням). Метод сортування використати на власний розсуд.

Варіант 27

1. Створити базовий клас LAKE з такими полями: name, square, deep, country, з конструкторами класу і методами, що змінюють усі поля класу. Створити дочірній клас POND з додатковим полем river, конструкторами класу і методом, що змінює поле river.
2. Файл input.txt містить властивості 12 екземплярів класу POND (озер). Написати програму, яка виведе на екран властивості найменшого і найбільшого за площею озера.

Варіант 28

1. Створити базовий клас CONTINENT з такими полями: name, square, з конструкторами класу і методом, що змінює площу. Створити дочірній клас COUNTRY з додатковими полями: contin, population, capital; з конструктоами класу і методами, що змінюють додаткові поля.
2. Файл countries.txt містить значення полів 6 об´єктів класу COUNTRY. Написати програму, яка обчислює загальне населення цих країн.

Варіант 29

1. Створити базовий клас ANIMALS з такими полями: name, areal, color, з конструкторами класу і методами, що змінюють усі поля. Створити дочірній клас BIRDS, який повинен мати додаткові поля weight, speed, власні конструктори класу і методи, що дозволяють змінювати додаткові поля.
2. Файл bird.txt містить значення полів для 9 об´єктів класу BIRDS. Написати програму, яка обчислить середню вагу цих птахів.

Варіант 30

1. Створити базовий клас FISH з такими полями: name, weight, length; з конструкторами класу і методами, що змінюють значення полів weight і length. Створити дочірній клас SHARK з додатковим полем speed, власними конструкторами класу і методом, що змінює поле speed.
2. Файл input.txt містить значення полів 7 об´єктів класу SHARK. Написати програму, яка обчислить середнє значення поля speed для всіх 7 об´єктів.

Контрольні питання

1. Назвіть основні властивості об′єктно-орієнтованого програмування.
2. Що таке поле?
3. Що таке метод?
4. Який метод називається статичним?
5. Який метод називаєтсья перевантаженим?
6. Що таке конструктор класу?

7. Для чого треба створювати два різних конструктори класу?
8. У чому полягає принцип успадкування?
9. Який клас називається батьківським?
10. Який клас називається дочірнім?
11. Як відбувається спадкування полів?
12. Як створити конструктор класу?
13. Які властивості батьківського класу успадковує дочірній клас?
14. Як успадкувати конструктор класу?
15. Як перевантажити конструктор класу?

Список рекомендованої та використаної літератури

1. Голуб Б.М. C#. Концепція та синтаксис. Навчальний посібник. Львів: Видавничий центр ЛНУ ім.І.Франка, 2006. 136 с.
2. Зеленський О.С. C#. Посібник для 10-11 класів інформаційно-технологічного профілю. Ч.1. 2010. 58 с.
3. Златопольський Д. М. Сборник задач по программированию – 2-е изд., перераб. и доп. СПб.: БХВ-Петербург, 2007.
4. Караванова Т.П. Інформатика: основи алгоритмізації та програмув.: 777 задач з рек. та прикл.: Навч. посіб. для 8-9 кл. із поглибл. вивч. інф-ки/ Зазаг. ред. М.З. Згуровського. К : Гєнеза, 2006. - 286 с.
5. Коноваленко І.В. Програмування мовою C# 6.0. Навчальний посібникдля технічних спеціальностейвищих навчальних закладів. Тернопіль: ТНТУ, 2016. 227 с.
6. Кравець П.О. Об'єктно-орієнтоване програмування. Видавництво Львівської політехніки, 2012. 624 с.
7. Шилдт Г. Полный справочник по C#., М.: "Вильямс", 2004. 752с.