Лабораторная работа № 6

Тема: Разработка Windows – приложения

Цель работы:
Научиться разрабатывать Windows Forms приложения, использовать графические интерфейсы для ввода и вывода данных, а также интегрировать и модифицировать ранее созданные классы для создания полноценного приложения.

Задание:
Создать Windows Forms приложение на языке C#, добавив новый проект в решение из лабораторной работы №5. Реализовать интерфейс для взаимодействия с программой через несколько форм и связать формы с ранее созданными классами для выполнения различных задач.

При выходе из программы должно запрашиваться подтверждение.

Должны быть реализованы следующие формы:

1. **Игра по отгадыванию ответа значения функции.**
    - Вывести формулу на экранную форму.
    - Компоненты для вводимых и выводимых данных должны быть подписаны.
    - Добавить возможность задать количество попыток для угадывания.
    - Использовать классы для вычисления значения функции, проверки ввода и расчета времени, реализовав дополнительный контроль правильности результата.
    - Добавить таймер для отслеживания времени на разгадывание (например, 30 секунд), по истечении которого игра завершается автоматически (**опционально**).
    - Реализуйте прогресс-бар для визуального отсчета времени (**опционально**).

2. **Работа с одномерными массивами.**
    - Должно быть поле ввода для ручного задания размера массива и кнопка для создания массива с длиной по умолчанию.
    - Должна быть кнопка для генерации случайного массива.
    - Для отображения массива использовать DataGridView с возможностью редактирования в ручную значения пользователем.
    - Кнопки для выполнения операций над массивом: сортировка, нахождение максимального и минимального значения, среднее арифметическое, а также выделение цветом максимального и минимального значений в DataGridView.
    - Обязательно модифицировать и использовать класс по работе с одномерными массивами.
    - Компоненты для вводимых и выводимых данных должны быть подписаны.

3. **Игра (из 4-ой лабораторной работы).**
    - Для игрового поля использовать DataGridView или другой компонент для графического отображения состояния игры.
    - Взаимодействие с игровым полем должно быть реализовано путём клика на поле (ячейку поля), а не ввода координат через поле ввода.
    - Компоненты для вводимых и выводимых данных должны быть подписаны.

4. **Вывод информации об авторе.**

Практические задачи:

1. Класс "Книга" с массивом глав

- Создайте класс Chapter с полями: title и content. Класс Book должен содержать массив глав (Chapter[]) и методы:
    - AddChapter(Chapter chapter) — добавление главы (если есть свободное место).
    - GetTotalPages() — расчет общего количества страниц (1 страница = 100 символов текста).
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main создайте несколько книг, добавьте главы в массив и рассчитайте общее количество страниц.

2. Иерархия классов "Фигуры" с массивами

- Создайте базовый класс Shape с методами GetArea() и GetPerimeter().
- Реализуйте наследников:
    - Circle (радиус, методы для площади и периметра).
    - Rectangle (длина, ширина).
    - Triangle (стороны a, b, c).
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main создайте массив фигур (Shape[]), рассчитайте общую площадь и периметр всех фигур.

3. Класс "Группа студентов"

- Создайте класс Student с полями: name, age, grades (массив оценок).
- Класс StudentGroup содержит массив студентов и методы:
    - AddStudent(Student student) — добавление студента в массив (если есть место).
    - GetTopStudent() — нахождение студента с лучшей средней оценкой.

- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main создайте группу, добавьте студентов и найдите лучшего.

4. Иерархия классов "Транспорт" с расчетом общего времени поездки

- Создайте базовый класс Transport с полями: speed и методом GetTravelTime(double distance).
- Реализуйте наследников:
    - Car (добавьте расход топлива).
    - Bike (ручной привод, не использует топливо).
    - Airplane (высокая скорость).
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main создайте массив транспорта (Transport[]), рассчитайте время поездки для всех объектов на заданной дистанции.

5. Класс "Ученик и оценки"

- Создайте класс Student с полями: Name, Grades (массив из 10 оценок).
- Реализуйте методы:
    - AddGrade(int grade) — добавление оценки (в первую свободную ячейку массива).
    - GetAverageGrade() — расчет средней оценки.
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main создайте массив из 5 учеников, добавьте им оценки, вычислите средний балл для каждого.

6. Класс "Автомобильный парк"

- Создайте класс Car с полями: brand, model, mileage, fuelConsumption.
- Создайте класс CarFleet, содержащий массив машин (Car[]) и методы:
    - AddCar(Car car) — добавление автомобиля в парк.
    - CalculateTotalFuelConsumption(double distance) — расчет общего расхода топлива парка на заданное расстояние.
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main добавьте машины в парк, рассчитайте общий расход топлива.

7. Класс "Университет и факультеты"

- Создайте класс Faculty с полями: name и departments (массив названий кафедр).
- Класс University содержит массив факультетов (Faculty[]) и методы:
    - AddFaculty(Faculty faculty) — добавление факультета в массив.
    - GetTotalDepartments() — подсчет общего числа кафедр.
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main создайте университет, добавьте факультеты и посчитайте кафедры.

8. Класс "Онлайн-магазин и товары"

- Создайте класс Product с полями: name, price, quantity.
- Класс OnlineStore содержит массив товаров (Product[]) и методы:
    - AddProduct(Product product) — добавление товара в массив.
    - GetTotalInventoryValue() — расчет общей стоимости всех товаров.
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main создайте магазин, добавьте товары и вычислите общую стоимость.

9. Класс "Банк и клиенты"

- Создайте класс Client с полями: name, accountBalance.
- Класс Bank содержит массив клиентов (Client[]) и методы:
    - AddClient(Client client) — добавление клиента.
    - CalculateTotalBalance() — расчет общего баланса клиентов.
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main добавьте клиентов в банк и посчитайте общий баланс.

10. Класс "Кинотеатр и фильмы"

- Создайте класс Movie с полями: title, duration, rating.
- Класс Cinema содержит массив фильмов (Movie[]) и методы:
    - AddMovie(Movie movie) — добавление фильма.
    - GetLongestMovie() — возвращение самого длинного фильма.
- Реализуйте несколько конструкторов для классов с параметрами и без и, если необходимо, добавьте соответствующие свойства.
- В Main добавьте фильмы и найдите самый длинный.