Приклад виконання курсової роботи

Зміст

1. Вступ

Необхідно розробити програму для відділу кадрів, яка призначена для ведення списку співробітників. Програма повинна мати наступні функціональні можливості:

- додавання співробітника;
- видалення співробітника;
- збереження інформації у файл;
- читання інформації з файлу;
- сортування співробітників за віком;
- виведення інформації о співробітниках трудовий стаж яких перевершує заданий.

Основної функцією буде – ведення списку співробітників. Розглянемо дерево функцій ІС (рис. 1).

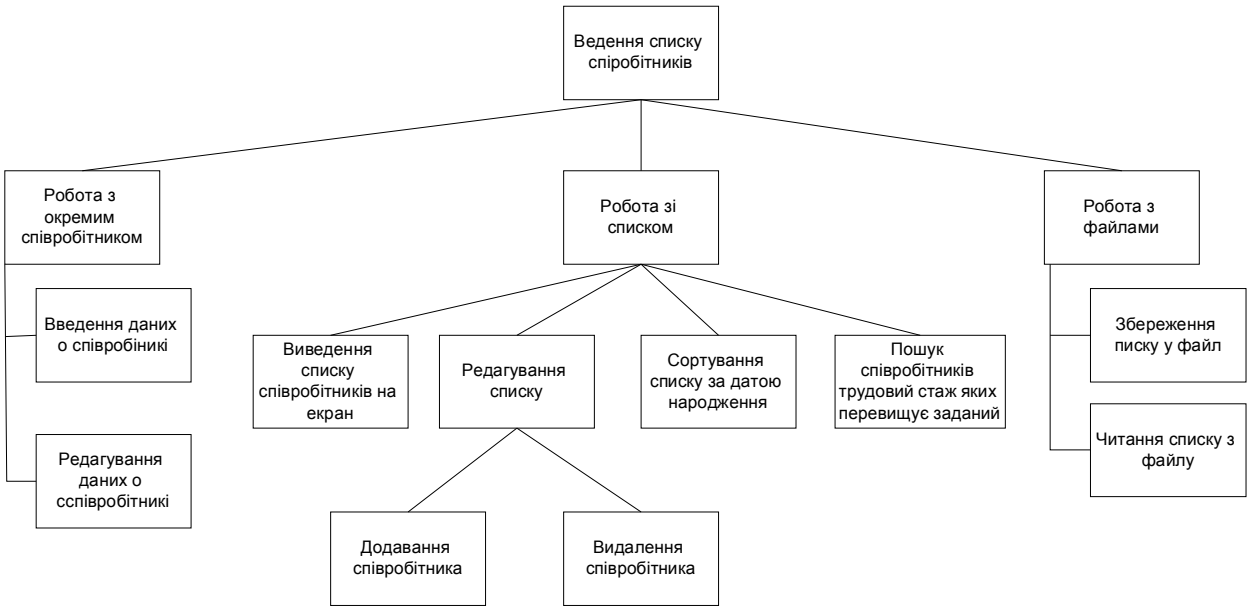

Рисунок 1. Дерево функцій

Виділимо основні сутності предметного середовища:

1. Сутність – співробітник, яка призначена для зберігання даних о конкретнім співробітнику. Дана сутність буде мати наступні атрибути:

- Табельний номер співробітника;
- ПІБ співробітнника;

- Рік народження співробітника;
- Стаж роботи співробітника

2. Список співробітників – сутність яка призначена зберігати інформацію о всіх співробітниках підприємства. Має наступні атрибути:

- Кількість співробітників у списку.

2.UML-діаграма прецедентів

Розглянемо діаграму прецедентів рис. 2.

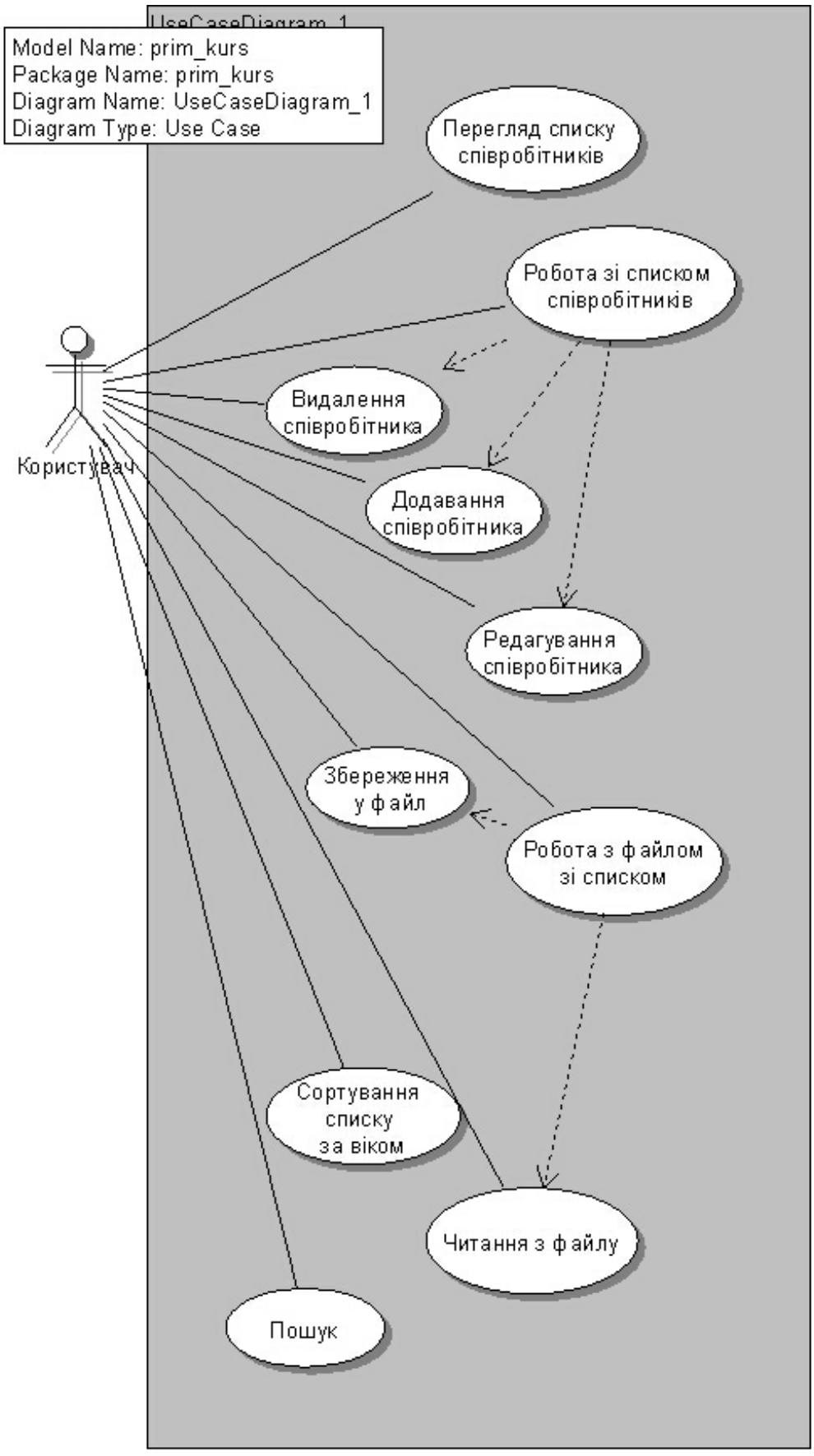

Рисунок 2. Діаграма прецедентів

Опишемо основні прецеденти:

1. Перегляд списку співробітників

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може переглянути список співробітників.

Передумови. Користувач запустив програму.

Результат (Постумова). На екрані користувача з‘явилася головна форма програми зі списком співробітників.

Основний успішний сценарій (або основний процес):

Користувач переглядає список співробітників.

2. Робота зі списком співробітників

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може працювати зі списком співробітників.

Передумови. Користувач запустив програму.

Результат (Постумова). На екрані користувача з‘явилася головна форма програми зі списком співробітників.

Основний успішний сценарій (або основний процес):

Користувач починає роботу зі списком співробітників.

3. Видалення співробітників

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може видаляти співробітника.

Передумови. Користувач почав роботу зі списком співробітників.

Результат (Постумова). Список співробітників змінився, обраний співробітник був видалений зі списку.

Основний успішний сценарій (або основний процес):

1. Користувач обирає співробітника зі списку, якого бажає видалити.
2. Користувач натискає на кнопку «Видалити».
3. Співробітника успішно видалено зі списку
4. Система виводить список з видаленим користувачем.

4. Додавання співробітників

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може додавати співробітника.

Передумови. Користувач почав роботу зі списком співробітників.

Результат (Постумова). Список співробітників змінився, обраний співробітник був доданий до списку.

Основний успішний сценарій (або основний процес):

1. Користувач вводить дані о співробітниках, а саме табельний номер, ПІБ, рік народження та робочий стаж.
2. Користувач натискає на кнопку «Додати».
3. Співробітника успішно доданий до списку.
4. Система виводить список з новим користувачем.

Розширення (або альтернативні потоки):

1а. Користувач вводить дані о співробітниках з помилкою.

2а. Користувач натискає на кнопку «Додати».

3а. Виводиться вікно з повідомленням о помилки при додаванні.

4а. Додавання не відбувається.

5. Редагування співробітника

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може редагувати дані о співробітниках.

Передумови. Користувач почав роботу зі списком співробітників.

Результат (Постумова). Список співробітників змінився, дані о обраном співробітнику були змінені.

Основний успішний сценарій (або основний процес):

1. Користувач обирає співробітника для редагування
2. Користувач натискає на кнопку «Редагувати».
3. З‘являється форма з полями для редагування, а саме з полями табельний номер, ПІБ, рік народження та робочий стаж.
4. Користувач редагує дані о співробітниках.

5. Користувач натискає на кнопку «Зберегти».

6. Дані співробітника змінено с списку.

7.Форма для редагування закривається.

8. Система виводить список з оновленою інформацією.

Розширення (або альтернативні потоки):

1а. Користувач обирає співробітника для редагування

2а. Користувач натискає на кнопку «Редагувати».

3а. З‘являється форма з полями для редагування, а саме з полями табельний номер, ПІБ, рік народження та робочий стаж.

4а. Користувач вводить дані о співробітниках з помилкою.

5а. Користувач натискає на кнопку «Зберегти».

6а. Виводиться вікно з повідомленням о помилки при редагуванні.

7а. Форма редагування не закривається, йде очікування введення без помилок.

6. Робота з файлом зі списком

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може працювати з файлом зі списком співробітників.

Передумови. Користувач запустив програму.

Результат (Постумова). На екрані користувача з‘явилася головна форма програми зі списком співробітників.

Основний успішний сценарій (або основний процес):

Користувач починає роботу з файлом зі списком співробітників.

7. Збереження у файл

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може зберігати список у файл.

Результат (Постумова). Список співробітників збережено у файл.

Основний успішний сценарій (або основний процес):

1. Користувач натискає кнопку «Зберегти».

2. Список успішно збережений у файлі.

Розширення (або альтернативні потоки):

1а. Користувач натискає кнопку «Зберегти».

2а. Виникла помилка при запису списку до файлу.

3а. Виводиться вікно з повідомленням о помилці під час запису.

4а. Збереження списку не відбулося.

8. Читання з файлу

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може читати список з файлу.

Результат (Постумова). Список співробітників прочитано з файлу і відображено на головній формі.

Основний успішний сценарій (або основний процес):

1. Користувач натискає кнопку «Читати з файлу».
2. Користувач обирає файл.
3. Список успішно прочитано з файлу.
4. Система виводить новий список, який було прочитано з файлу.

Розширення (або альтернативні потоки):

1а. Користувач натискає кнопку «Читати з файлу».

2а.Користувач обирає файл.

3а. Виникла помилка під час зчитування з файлу.

4а. Виводиться вікно з повідомленням о помилці під час читання.

5а. Зміна списку не відбулася.

9. Сортування списку за віком

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може сортувати список.

Результат (Постумова). Список співробітників відсортовано за віком.

Основний успішний сценарій (або основний процес):

1. Користувач натискає кнопку «Сортування».
2. Система виводить список відсортований за віком.

10. Пошук

Основний виконавець. Користувач.

Зацікавлені особи та їх вимоги. Користувач. Може шукати співробітників зі стажем роботи більше заданого.

Результат (Постумова). Список співробітників зі стажем роботи більше заданого.

Основний успішний сценарій (або основний процес):

1. Користувач вводить у поле необхідний стаж роботи.

2.Користувач натискає кнопку «Пошук».

3. Система виводить список відсортований за стажем більше заданого.

Розширення (або альтернативні потоки):

1а. Користувач вводить у поле необхідний стаж роботи з помилкою.

2а. Виводиться вікно з повідомленням о помилці при введені стажу.

3а. Список не виводиться

3.UML-діаграма класів

Розглянемо діаграму класів системи (рис.3).

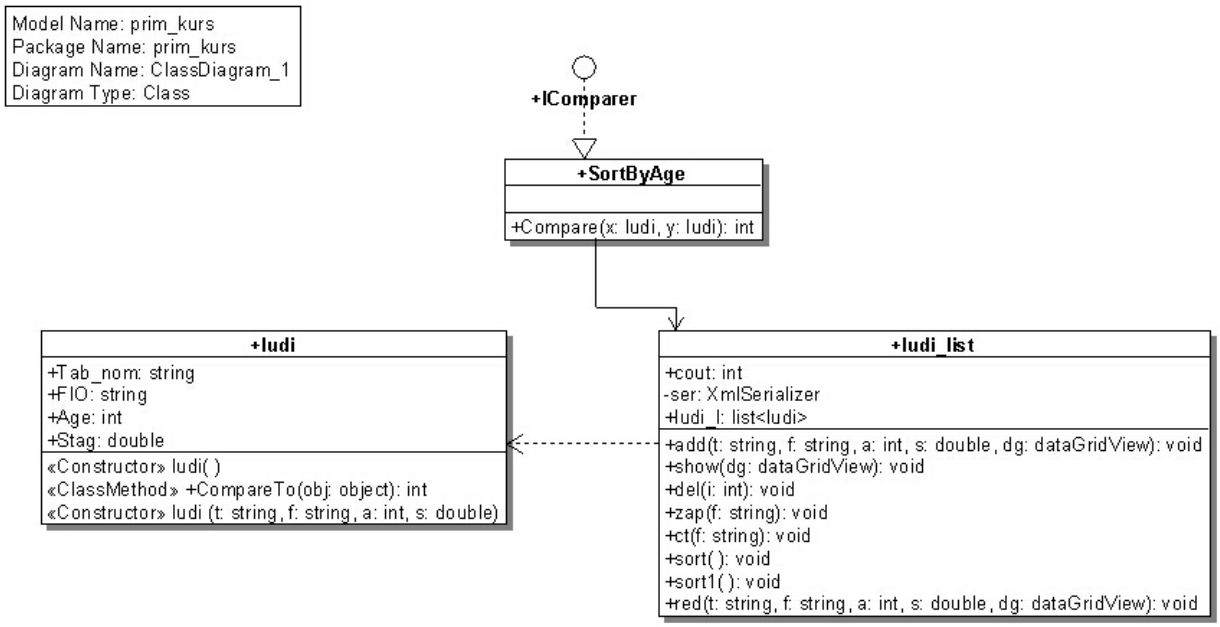

Рисунок 3. Діаграма класів

Опишемо кожен з класів більш докладніше.

1. Клас ludi

Клас призначений для відображення даних про співробітників компанії.

Даний клас має наступні поля:

- public string Tab_nom – атрибут призначений для збереження табельного номеру співробітника;
- public string FIO - атрибут призначений для збереження ПІБ півробітника;
- public int Age – атрибут призначений для збереження ріку народження співробітника;
- public double Stag – атрибут призначений для збереження стажу співробітника.

Даний клас має конструктори:

- ludi() - конструктор без параметрів, у якому усім полям буде присвоєно значення за замовченням.
- ludi (string t, string f, int a, double s) – конструктор з параметрами, у якому атрибутам присвоюється значеня параметрів.

Даний клас має наступні методи

- public int CompareTo(object obj) – метод призначений для реалізації інтерфейсу IComparable, а саме для порівнянн стажу роботу співробітника.

2.Клас SortByAge

Клас призначений для перевантаження стандартного інтерфейсу IComparer.

Даний клас має наступні методи

- public int Compare(ludi x, ludi y)– метод призначений для реалізації інтерфейсу IComparer, а саме для перевизначеннння методу Compare, для порівняння віку двох співробітників.

3. Клас ludi_list

Клас призначений для роботи зі списком співробітників.

Даний клас має наступні поля:

- public int cout – атрибут для збереження кількості співробітників у списку.
- private XmlSerializer ser – атрибут для проведення серіалізації, при збереженні та читанні списку з файлу.

- public list<ludi> ludi_l – атрибут списку об’єктів класу ludi.

  Даний клас має наступні методи
- public void add(string t, string f, int a, double s, dataGridView dg) - метод для додавання співробітника до списку;
- public void show(dataGridView dg) - метод для виведення списку на екран;
- public void del(int i) – метод для видалення співробітника зі списку;
- public void zap(string f) – метод ля запису списку у файл;
- public void ct(string f) – метод для читання списку з файлу;
- public void sort() – метод ля сортування списку за віком;
- public void sort1() – метод ля сортування списком за стажем, який більше за заданий;
- public void red(string t, string f, int a, double s, dataGridView dg) – метод для редагування запису про співробітника.

4. Опис системних операцій та поведінки програми у вигляді UML-діаграм послідовностей

Опишемо поведінку системи у вигляді діаграми послідовностей для основних операцій системи, а саме:

1. Додавання співробітника у список (рис. 4)

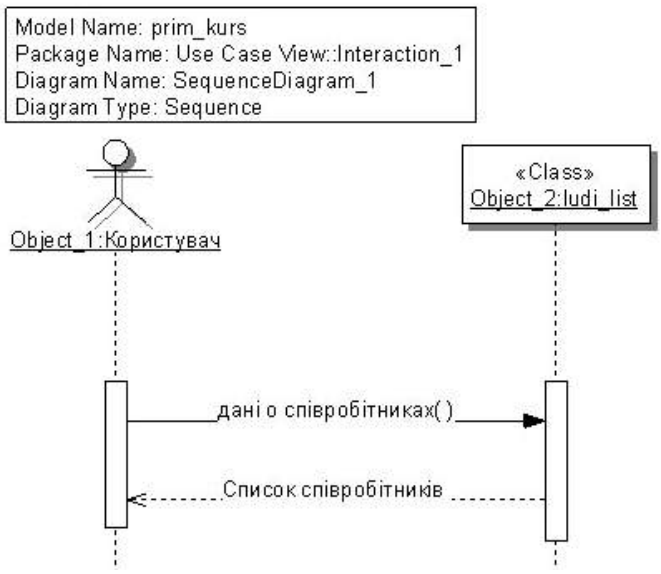

Рисунок 4. Операція додавання у список

2. Редагування даних о співробітниках (рис. 5)

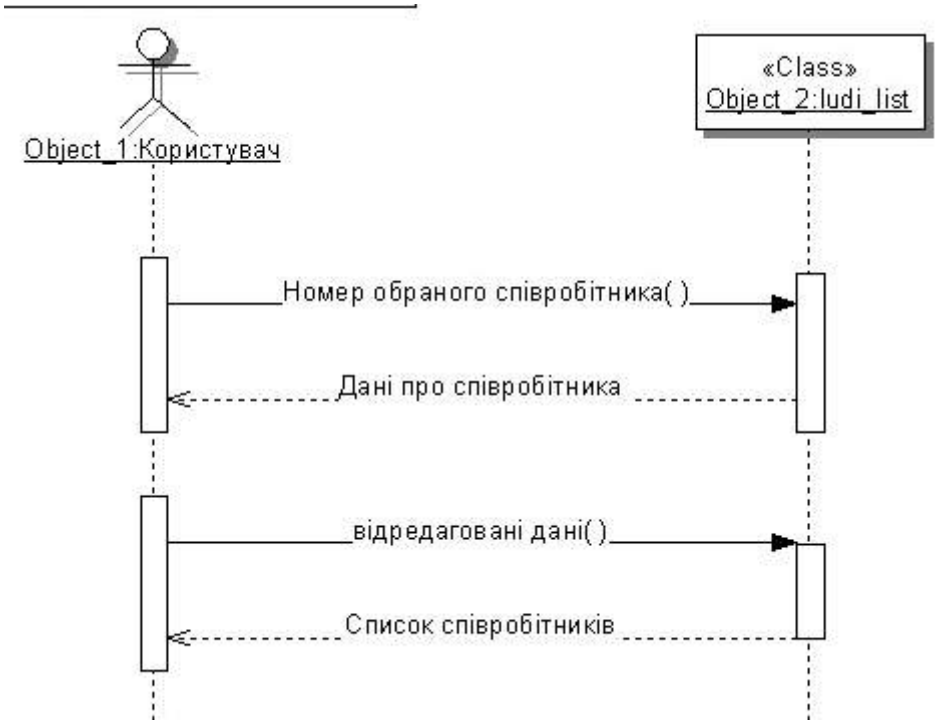

Рисунок 5. Операція редагування даних

3. Сортування даних за віком(рис. 6).

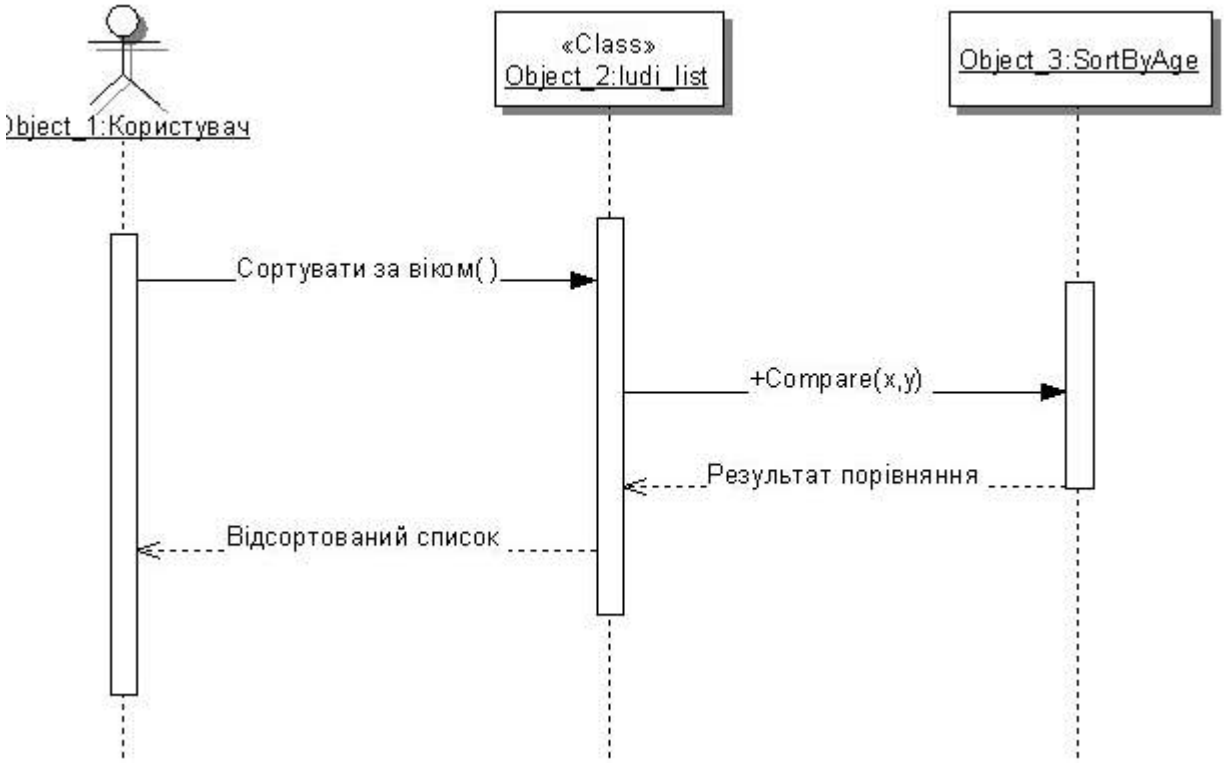

Рисунок 6. Операція сортування за віком

4. Робота з файлами (рис. 7).

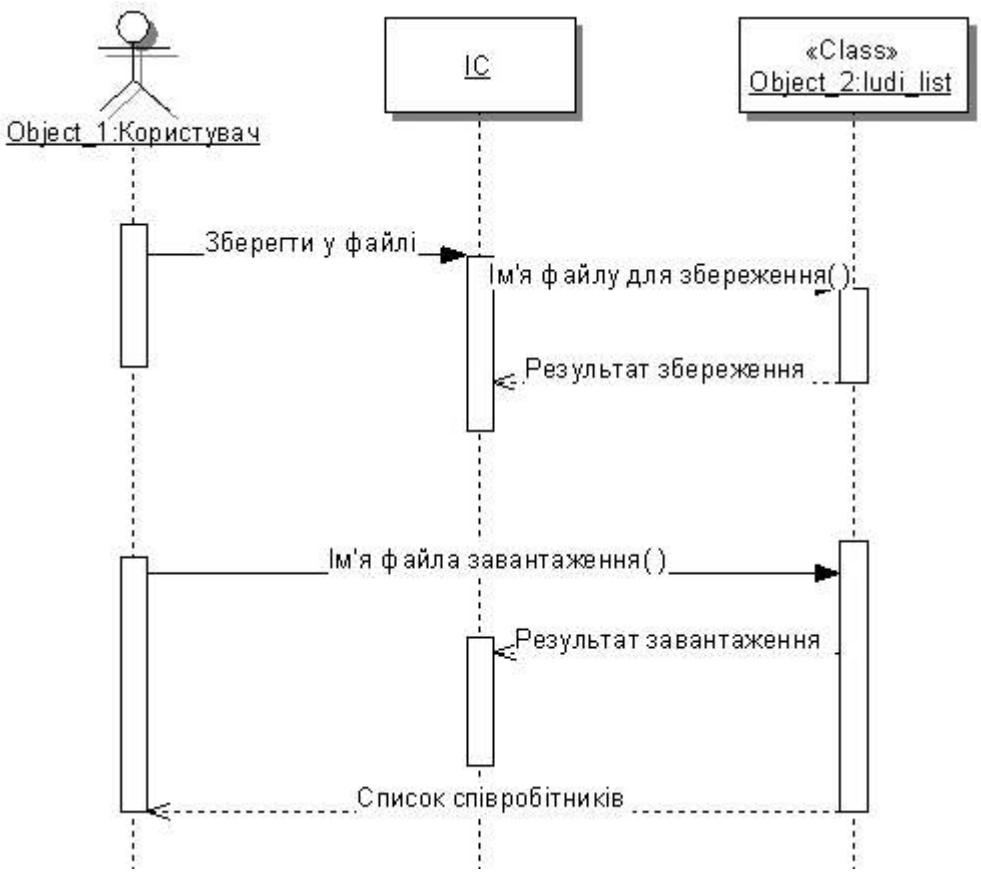

Рисунок 7. Операція робота з файлами

5. Опис структури програми у вигляді UML-діаграми компонентів

Опишемо структуру програми у вигляді діаграми компонентів. Розроблена програма включає три компонента:

- Головна форма;
- Форма для редагування даних;
- Класи.

Діаграма компонентів наведена на рис. 8.

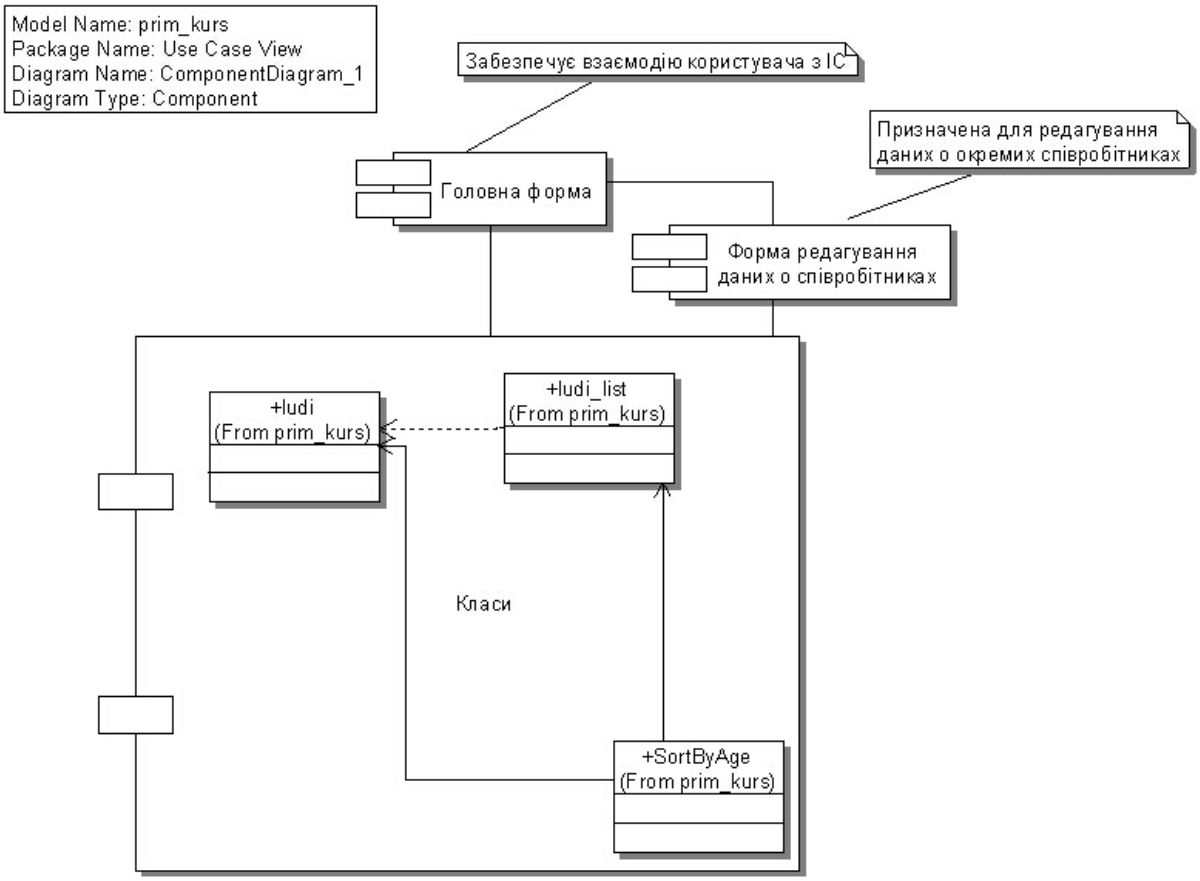

Рисунок 8. Діаграма компонентів

6. Опис тестових прикладів виконання програми

7. Список літератури

8. Додатки з роздруком тексту програми