Лабораторна робота №24

Тема: ООП. Потоки

Мета: Навчитися використовувати ООП. Потоки.

Індивідуальне завдання

Загальне завдання.

Поширити попередню лабораторну роботу таким чином:

- використання функцій printf/scanf замінити на використання cin/cout;
- усі конкатенації рядків замінити на використання stringstream;
- замінити метод виводу інформації про об'єкт на метод, що повертає рядок-інформацію про об'єкт, який далі можна виводити на екран;

```
std::string MyObject::toString();
```

- замінити метод вводу інформації про об'єкт на метод, що приймає рядок з інформацією про об'єкт, обробляє його та створює об'єкт на базі цієї інформації;
- поширити клас-список, шляхом реалізації методів роботи з файлами за допомогою файлових потоків (fstream) (якщо використовувалися функції fprintf/fscanf – замінити їх на класи ifsteam/ofstream), при цьому сигнатури методів повинні виглядати таким чином:
  - читання (List – клас-список об'єктів, при цьому слід пам'ятати, що при повторному читанні з файлу, попередні дані списку повинні бути очищені):

```
void List::readFromFile(string fileName);
```

  - запис:

```
void List::writeToFile(string fileName);
```

- продемонструвати відсутність витоків пам'яті;
- продемонструвати роботу розроблених методів за допомогою модульних тестів;
- не використовувати конструкцію "using namespace std;", замість цього слід робити "using" кожного необхідного класу: `using std::string`, `using std::cout`;

Хід роботи

1. Доповнення коду для базового класу, класу списку та класу помічника.

```
void Helper::menu(List& list) {
    cout << "\n\nA list of employees.Select an action from the following.\n\n";
    cout << "1.Write list to screen.\n";
    cout << "2.Generate and add employee.\n";
    cout << "3.Delete employee by index.\n";
    cout << "4.Search for employees with insurance.\n";
    cout << "5.Reading employees from a file.\n";
    cout << "6.Writing employees in file.\n";
    cout << "0.Exit.\n\n";
    cout << "Your choice: ";

    int choice = -1;

    while (choice < 0 || choice>7) {
        cin >> choice;
        if (choice < 0 || choice>7)
            cout << "You entered an invalid value, please retry\n\nYour choice: ";
    }
    cout << endl << endl;
```

Рисунок 1.1 - Приклад використання потоків за допомогою cin та cout.

```
string Employee::printEmployee() {
    std::stringstream ss;
    ss << "\nCompany:" << company;
    ss << "\nMail:" << mail;
    ss << "\nFullname:" << fullName;
    ss << "\nCharacteristic:" << characteristic;
    ss << "\nWork experience in years:" << workExperience;
    ss << "\nInsurance:" << insurance ? "Yes" : "No";

    return ss.str();
}
```

Рисунок 1.2 - Приклад використання stringstream для конкатенації у методі, який повертає рядок для виводу на екран.

```
friend std::istream& operator>> (std::istream& in, Employee& obj) {
    in >> obj.company;
    in >> obj.mail;
    in >> obj.fullName;
    in >> obj.characteristic;
    in >> obj.workExperience;
    in >> obj.insurance;

    return in;
}
```

Рисунок 1.3 - Приклад створення об’єкта за допомогою потоку.

```
void List::readFromFile() {
    string company;
    string mail;
    string fullName;
    string characteristic;
    int workExperience;
    bool insurance;

    std::ifstream in(filename);
    if (in.is_open())
    {
        list.clear();
        while (in >> company >> mail >> fullName >> characteristic >> workExperience >> insurance)
            list.push_back(Employee(company, mail, fullName, characteristic, workExperience, insurance));
    }
    in.close();
}
```

Рисунок 1.4 - Приклад методу зчитування з файлу.

```
void List::writeToFile() {
    std::ofstream out(filename);
    if (out.is_open()) {
        for (int i = 0; i < list.size(); i++)
            out << list[i].getCompany() << " " << list[i].getMail() << " " << list[i].getFullName() << " " << list[i].getCharac() << " " << list[i].getWorkExp() <<
            " " << list[i].getInsurance() << endl;
    }
    out.close();
}
```

Рисунок 1.5 - Приклад методу запису до файлу.

Висновок: Навчився використовувати ООП. Потоки.