**Лабораторна робота №23.**

**Вступ до ООП.**

* **Вимоги**
* **Розробник**
* Гладков Костянтинт Сергiович
* Студент групи КІТ-320;
* 01-march-2021.
* **Основне завдання**



* **Опис роботи**

Програма працює за тим же принципом, що і колишні, однак використовує ООП, полегшуючи роботу. Також там присутній читання з файлу і запис в нього (ну і інші слабкі зміни)

* **Функціональне призначення**

Дана програма може бути використана для маніпуляцій з нашим динамічним масивом, який має в собі класи. Таким чином ми отримуємо великий функціонал на увазі більше кількості можливих методів для роботи з нашим масивом. Також там присутній читання з файлу і запис в нього (ну і інші слабкі зміни)

* **Опис логічної структури**

Данна програма працює за принципом створення класу, в якому зберігається динамічний масив іншого класу, а також за допомогою методів, які контролюється цей масив. Також там присутній читання з файлу і запис в нього (ну і інші слабкі зміни)

Функція main містить в собі виклик функцій, а також корисних властивостей для функцій

class List - клас, який зберігає в собі динамічний масив іншого класу, а також які всередині себе має всі методи роботи з даним класом.

class Agency - базовий клас.

void List::expand() - метод, який використовується для розширення обсягу нашого динамічного масиву

void List::add\_ag - метод, який використовується для розширення нашого масиву, а точніше додавання в даній оновлений масив нового елемента за рахунок спеціального коструктора

Agency& List::get\_ag - метод, який дозволяє отримати елемент масиву за індексом.

void List::remove\_last\_ag() - метод, який прибирає останнім агенство з масиву.

int List::get\_oldest\_ag() - метод, який дозволяє знайти самий старший елемент в масиві (iндекс).

void List::read\_from\_file - метод для читання з файлу

void List::write\_to\_file - метод для запис у файл



рис. 1 - void List::expand()



рис. 2 - void List::add\_ag



рис. 3 - Agency& List::get\_ag

рис. 4 - void List::remove\_last\_ag()



рис. 5 - int List::get\_oldest\_ag()



рис. 6 - List::read\_from\_file



рис. 7 - List::write\_to\_file

* **Важливі елементи програми**

List my\_list;

my\_list.add\_ag(Agency("Kostya", "Gladkov", "Email", "help", "London", 10));

my\_list.add\_ag(Agency("Someone", "Noone", "Email", "nope", "London", 15));

for (int i = 0; i < my\_list.get\_num(); i++){

cout << my\_list.get\_ag(i).get\_info() << "\n";

}

this->cap \*= 2;

Agency \*\*temp\_arr = new Agency\*[this->cap];

for (int i = 0; i < this->num\_of\_elem; i++)

{

temp\_arr[i] = new Agency(\*this->pAgency[i]);

}

for (int i = 0; i < this->num\_of\_elem; i++)

{

delete this->pAgency[i];

}

delete [] this->pAgency;

this->pAgency = temp\_arr;

this->initialize(this->num\_of\_elem);

* **Варіанти використання**

Дана програма може бути використана для роботи з базовим класом, а також класом-списком, який буде зберігати в собі динамічний масив базового класу. Також дані можуть братися з файлу і в кінці туди записуватися.

**Висновки**

У даній лабораторній роботі були придбані знання роботи з ООП і отримав досвід роботи з файлами в с ++