Adatbázis kezelő program

Ez a C++ program egy adatbázis rendszert képes működtetni file műveletek segítségével. SQL adatbázis kezelést a program nem tartalmaz. A program képes személy neveket tárolni a database.txt file -ban. A rekord tartalmazza a vezeték és kereszt nevet szünettel elválasztva. A keresztnév után vesszővel elválasztva a rekord azonosító számát találjuk meg. A program segítségével lehet új rekordot hozzáadni, listázni és törölni is. A Windows operációs rendszer konzol ablakában futtatható a program.

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
class database
{
private:
	int id = 0;
	string field1;
public:
	void menu();
	void get_database_field_from_user();
	void add_user();
	void delete_user();
	void dinamikus_tombbe_olvasas(int count);
	void list_users();
	int get_number_of_records();
};

int main()
{
	setlocale(LC_ALL, "hun");
	database p1;
	p1.menu();
	system("pause");
	return 0;
}
```

```
void database::menu()
{
	int menu = 0;
	cout << "Üdvözlöm az adatbázis kezelő programban!" << endl;
	do
	{
		cout << endl;
		cout << "\t" << "Menü:" << endl;
		cout << "---" << endl;
		cout << "1. ADD record" << endl;
		cout << "---" << endl;
		cout << "2. DELETE record, sorszám alapján" << endl;
		cout << "---" << endl;
		cout << "3. LIST records" << endl;
		cout << "---" << endl;
		cout << "4. EXIT" << endl;
		cin >> menu;
		if (menu == 1)
		{
			add_user();
		}
		if (menu == 2)
		{
			delete_user();
		}
		if (menu == 3)
		{
			list_users();
		}
	} while (menu != 4);
	cout << "Viszontlátásra!" << endl;
}
void database::get_database_field_from_user()
{
	cout << "Kérem adja meg a felhasználó nevét!" << endl;
	getline(cin >> ws, field1);//a vezeték és keresztnév közé space is     kerülhet
}
void database::add_user()
{
	ofstream output;//kimeneti adatfolyam példánya
	string outPutFileName = "database.txt";
	output.open(outPutFileName, ios_base::app);
	//file megnyitás hozzáfűzés üzemmódban
	get_database_field_from_user();//a rekord egy mezőjének bekérése
	id = get_number_of_records();//korábbi rekordok számának meghatározása
	id++;
	output << field1 << ", " << id << endl;//a rekord file -ba írása
	cout << endl;
	cout << "  --- Hozzáadás sikeres! ---" << endl;
	cout << endl;
	output.close();//file bezárás
}
```

```
void database::list_users()
{
	ifstream input;
	string inputFileName = "database.txt";
	input.open(inputFileName);
	string cel;
	//segéd string változó, ami a getline() függvény híváshoz szükséges
	while (getline(input, cel))
	{
		cout << cel << endl;
	}
	cout << "---" << endl;
	input.close();//file bezárás
}
int database::get_number_of_records()
{
	ifstream input;
	string inputFileName = "database.txt";
	input.open(inputFileName);
	string cel;
	int rekordok_szama = 0;
	while (getline(input, cel))
	{
		rekordok_szama++;
	}
	input.close();
	return rekordok_szama;
}


void database::delete_user()
//a törölni kívánt elemet kihagyom, nem pedig törlöm
{
	int count = 0;
	fstream input;
	string inputFileName = "database.txt";
	input.open(inputFileName);
	string cel;
	while (getline(input, cel))
	{
		count++;//a rekordok száma kell a dinamikus tömb foglalásához
	}
	dinamikus_tombbe_olvasas(count)
	input.close();
}
```

```
void database::dinamikus_tombbe_olvasas(int count){
	fstream input;
	string inputFileName = "database.txt";
	input.open(inputFileName);
	string cel;
	string* pStringTomb = new string[count];//dinamikus tömb foglalás
	int i = 0;
	while (getline(input, cel)){
		pStringTomb[i] = cel;
		cout << i + 1 << ". record: " << pStringTomb[i] << endl;
		i++;
	}
	input.close();//file bezárás
	//FILE -ba írás új sorszámmal
	ofstream output;
	string outputFileName = "database.txt";
	output.open(outputFileName);
	cout << "Kérem a sorszámot: " << endl;
	int sorszam = 0;
	cin >> sorszam;
	cout << "---" << endl;
	i = 0;
	while (i < sorszam - 1){
		output << pStringTomb[i] << endl;
		i++;
	}
	i++;
	cout << "  --- Törlés sikeres! ---" << endl;
	cout << "Törölt rekord: " << pStringTomb[i - 1] << endl;
	int uj_ID = i;//innentől kell cserélni az ID -t
	//módszer:
	//1. elmegyek rekord végére és megszámolom az id hosszát
	//2. replace-el kicserelem az id -t
	string uj_ID_string;
	int x1 = 0;
	int x2 = 0;
	int i_edik_id_mezo_hossz = 0;
	while (i < count){
		x2 = pStringTomb[i].length();//a rekord vége
		x1 = pStringTomb[i].length() - 1;
		while (pStringTomb[i].at(x1) != ','){
			x1--;
		}
		i_edik_id_mezo_hossz = x2 - x1;
		//az id mérete a nagyobbik távolság minusz a kisebbik táv: deltaX=x2-x1
		uj_ID++;
		uj_ID_string = to_string(uj_ID - 1);
		pStringTomb[i].replace(x1 + 2, i_edik_id_mezo_hossz, uj_ID_string);
		//x1+2 a vessző és a szünet miatt van
		output << pStringTomb[i] << endl;
		i++;
	}
	output.close();//file bezárás
	delete[] pStringTomb;
}
```