**Система за управление на БД със самолети за авиобаза**

Изготвено от Таня Желева

Проектът представлява умалена версия на база данни за самолети. Поддържа се основна реализация на REPL (read-evaluate-print-loop) със сложности *О(n)*. Предвидена е и Optimize команда, която намаля сложността до *O(log(n))*. Информацията за всички самолети се съхранява в постоянната памет във вид на записи в текстов файл. Всеки самолетен запис има вида *Id Plane Type Flights*.

**1 Архитектура**

* 1. **Plane**

Класът Plane описва отделен запис в базата. Основните му характеристики са:

* Private:
* id – Id на дадения самолет
* name – име на самолет
* type – типа на самолет
* flights – брой извършени полети
* Public:
* Plane() – конструктор по подразбиране
* Plane(int, const string&, const string&, int) – конструктор, приема атрибути Id – уникално число, Plane – име, Type – тип и Flights – брой извършени полети. В себе си вика методите *SetId(int), SetName(const string&), SetType(const string&), SetFlights(int).*
* void SetId(int) – сетър, задава стойност на *id* полето, като проверява дали подаденият параметър е валиден. В случай на невалидност хвърля *invalid\_argument exception*.
* void SetName(const string&) – сетър, задава стойност на полето *name*, като проверява дали подаденият параметър е валиден. В случай на невалидност хвърля *invalid\_argument exception*.
* void SetType(const string&) – сетър, задава стойност на полето *type*, като проверява дали подаденият параметър е валиден. В случай на невалидност хвърля *invalid\_argument exception*.
* void SetFlights(int) – сетър, задава стойност на полето *flights*, като проверява дали подаденият параметър е валиден. В случай на невалидност хвърля *invalid\_argument exception*.
* int GetId() const – гетър, връща стойността на полето *id*.
* const string GetName() const – гетър, връща стойността на полето *name*.
* const string GetType() const – гетър, връща стойността на полето *type*.
* int GetFlights() const – гетър, връща стойността на полето *flights*.
* const string ToString() – връща данните на самолета под формата на *string*.
* void Print() const – принтира данните на самолета.
* friend ostream& operator<<(ostream& out, const Plane&) – приятелска фукнция, предефинира се оператора << за класът *Plane*.
  1. **PlaneFactory**

Класът PlaneFactory описва основните операции поддържани от базата.

* Private:
* map<int, Plane> planesOptimized – пази във вида ключ-стойност самолетите след използване на Optimize командата. За ключ се използва id атрибута на всеки самолет.
* vector<Plane> planes – списък от самолетите в базата.
* string fileName – име на файла, който съдържа данните.
* bool isOptimizeUsed – показва дали е използвана Optimize командата.