Документация към проект на тема Система за управление на БД със самолети за авиобаза

към курса по Структури от данни и програмиране за специалност Информатика зимен семестър

2019/2020г.

1.Описание на заданието:

Проектът представлява реализация на самолетна база от данни,която се съхранява под формата на текстов файл. Методите,които трябва да бъдат имплементирани са създаване(createPlane),

изтриване(deletePlane),редактиране(updatePlane),преглед(showPlane) и търсене(searchPlane).

2.Стъпки за реализация:

Реализирана е структура от данни,която съдържа информацията за даден самолет и клас,който съдържа динамичен масив от структурите с информацията и булева променлива(проверка дали масивът е сортиран).При създаване,изтриване и редактиране на конкретен самолет информацията се подава на текстовия файл с извършване на съответното действие.За търсене с време О(n) е използван метод за линейно търсене, а за оптимизиране е използван алгоритъм метод на сливането с време за изпълнение О(n logn) и двоично търсене с време за изпълнение O(log n).

3.1 Структурата PlaneDetails има следните член-данни и член-методи:

Член-данни:

* long int id - цяло положително число с максимална стойност 2^50(уникален идентификатор).
* std::string plane - низ с максимална дължина 2^8(име на самолет).
* Std::sting type - низ, с максимална дължина 2^8(тип самолет).
* long int flights - цяло положително число с максимална стойност 2^50(брой осъществени полети).

Член-функции:

* PlaneDetails() - контруктор по подразбиране.
* PlaneDetails(long long int,std::string,std::string, long long int) – конструктор с параметри.
* void print() const – функция за извеждане на информацията,която е константна(не може да се променят данните).

3.2 Класът Plane има следните член-данни и член-функции:

Член-данни:

* bool isSorted – булева променлива,показваща дали масивът е сортиран;
* std::vector<PlaneDetails> planes – динамичен масив от самолети с информация PlaneDetails.

Помощни функции(private):

* bool validateData(long long int, std::string, std::string, long long int) – функция,която проверява дали данните са коректни.
* void createPlaneInFile(long long int, std::string, std::string,long long int) – функция,която при създаване на самолет го записва във файл.
* void deletePlaneInFile(long long int) – функция, която при изтриване на самолет го премахва от файл.
* void updatePlaneInFile(long long int,int) – функция, която при промяна на самолет го променя и във файл.
* int searchPlaneByID(long long int) – функция,която връща индекс на търсен самолет по индетификатор (линейно търсене).
* void merge(std::vector<PlaneDetails>&, std::vector<PlaneDetails>&, std::vector<PlaneDetails>&) – функция, която приема 2 масива, сравнява им елементите и записва по-малкият в редицата и го прескача. След изчерпване на елементите на някоя от редиците другите се записват в редицата.
* void mergeSort(std::vector<PlaneDetails>&) – функция,която разделя редицата на 2 и сортира рекурсивно 2те редици, след което с помощта на merge ги слива в една редица.Новата редица е сортирана.
* int binarySearch(int,int,long long int) – функция,която намира елемент с двоично търсене.
* void fillVector() - функция,която при стартиране на програмата зарежда всички записи във вектора, ако файлът не е празен.

Член-функции:

* Plane() – конструктор по подразбиране,който вика функцията fillVector().
* void createPlane(long long int, std::string, std::string, long long int) – създава запис на самолет.Вика в себе си createPlaneInFile и validateData.
* void deletePlane(long long int) – изтрива запис на самолет.Вика в себе си deletePlaneInFile.
* void updatePlane(long long int, std::string, std::string) – променя атрибут на самолет.Вика в себе си updatePlaneInFile.
* void showPlanes(long int, long int) – показва поискан брой самолети от даден индекс.
* void searchPlane(long long int) – търси по идентификатор на самолет, в зависимост от това дали е извикана optimizePlane или не търси линейно или двоично.
* void optimizePlane() – сортива масива с помощта на метод на сливането.

4.Идеи за бъдещи подобрения:

Реализиране на програмата с червено-черно дърво.Няма да има нужда от сортировка и търсенето ще отнема време O(log n).

5.Референция към git хранилище:

https://github.com/galyap38/PlaneDatabase.git

Изготвена от Галя Петрова,ф.н:45263,гр:2,сп:Информатика