**Условие**

*Направете система, която да дава възможност за комуникация чрез заявки между разработчици и техния ръководител на екип. Имплементирайте функционалности за изпращане на заявка за напускане и заявка за повишаване на заплатата.*

**Клас Developer**

*Разработчик.*

* Име (низ);
* Заплата (double);
* Ръководител на екип (указател към обект от клас TeamLead);
* Конструктор с 1 параметър const string& name: името е name, заплатата е 0, указателят сочи към nullptr;
* getName(): връща string - текущата стойност на името;
* getSalary(): връща double - текущата стойност на заплатата;
* getTeamLead(): връща TeamLead \* - текущата стойност на указателя към ръководителя;
* setInitialSalary(double amount): присвоява за заплатата подадената сума amount, само ако нейната текуща стойност е 0; (Ако не е да не присвоява стойност)
* void sendLeavingRequest();: подава към ръководителя LeavingRequest с името на разработчика;
* void sendPromotionRequest(double amount);: подава към ръководителя PromotionRequest с името на разработчика и количеството на заплатата, подадено в аргумента на функцията;

**Клас TeamLead**

*Ръководител на екип*.

Освен всичко изброено в Developer, съхранява и:

* вектор с указатели към обекти от клас Developer; (ръководителят не е част от екипа)
* заявки за напускане (LeavingRequest);
* заявки за повишаване на заплатата (PromotionRequest);
* Конструктор с 2 параметъра const string& name, double salary: името е name, заплатата е salary, a указателят към ръководителят на екипа сочи към текущият обект;
* getTeam(): връща vector<Developer \*> - текущият вектор, който представлява екипа;
* void addDeveloperToTeam(Developer \* developer, double salary);: добавя разработчик към екипа и задава заплатата му да бъде със стойност salary. Трябва за разработчикът да се промени, че вече има ръководител на екипа; (Възможно е да има разработчици в екипа с еднакви имена. Съобразете дали developer не сочи към nullptr.)
* void removeDeveloperFromTeam(const string& name);: премахва разработчик от екипа по подадено  име name. (Ако не се намери търсения разработчик, не последва действие. Ако намери разработчици с еднакво име премахва последния.)
* void increaseTeamSalariesBy(double amount);: повишава заплатите на разработчиците в екипа със стойност amount;
* void decreaseTeamSalariesBy(double amount);: понижава заплатите на разработчиците в екипа със стойност amount;
* void addLeavingRequest(const LeavingRequest& leavingRequest);: добавя на съхранение заявка за напускане;
* void addPromotionRequest(const PromotionRequest& promotionRequest);: добавя на съхранение заявка за повишаване на заплатата;
* void fulfillLeavingRequests();: изпълнява всички съхранени заявки за напускане, като премахва от екипа изпратилите заявките, променя техният ръководител (TeamLead \* да сочи към nullptr) и изчиства всички изпълнени заявки;
* void fulfillPromotionRequests();: изпълнява всички съхранени заявки за повишаване на заплатата, повишава заплатата на изпратилите заявките и изчиства всички изпълнени заявки;

**Клас Request**

*Заявка.*

* Съобщение (низ)
* Изпращач (низ) (името на този, който я изпраща)
* Брояч (цяло число) - започва от 0 и се увеличава с всеки създаден обект от тип Request или негови  наследници;
* ID (цяло число) - уникален идентификатор, който има стойността на брояча в момента на създаването му (ако сме създали два обекта от тип Request или негови наследници, първият ще има ID със стойност 1, а вторият ID със стойност 2);
* Конструктор с 2 параметъра const string& message, const string& sender: съобщението е message, изпращачът е sender, броячът се увеличава с 1, ID приема стойността на брояча;
* getMessage() : връща string - текущата стойност на съобщението;
* getSender(): връща string - текущата стойност на изпращача;
* getCount(): връща int - текущата стойност на брояча;
* getID(): връща int - текущата стойност на ID;

**Клас LeavingRequest**

*Заявка за напускане.*

Освен всичко изброено в Request:

* Конструктор с 1 параметъра const string& sender: съобщението е "I want to leave!", изпращачът е sender, броячът се увеличава с 1, ID приема стойността на брояча;

**Клас PromotionRequest**

*Заявка за повишаване на заплатата.*

Освен всичко изброено в Request:

* Kоличество (double)  - количество, с което да се увеличи заплатата;
* Конструктор с 2 параметъра const string& sender, double amount: съобщението е "I want a raise!", изпращачът е sender, количеството е amount , броячът се увеличава с 1, ID приема стойността на брояча;
* getAmount(): връща double - текущата стойност на количеството;