**Министерство образования Российской Федерации**

**МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**им. Н.Э. БАУМАНА**

Факультет: Информатика и системы управления

Кафедра: Информационная безопасность (ИУ8)

#### **ДОМАШНЕЕ ЗАДАНИЕ № 1**

“Объектно-ориентированное программирование”

**Вариант № 3**

**Руководитель практики:**

Мироненко Е.О.

**Студент**: Курганов. Н.С.

Москва 2016

**ЗАДАНИЕ**

**Сотрудники, подразделения -- отдел кадров**

Подразделения в предприятии имеют иерархическую структуру.

Процессы:

* найм, увольнение сотрудника
* создание, переформирование, расформирование, перемещение, объединение подразделений
* поиск сотрудника, определение структуры его подчиненности
* переход сотрудников между подразделениями
* работа сотрудника на несколько ставок в различных подразделениях
* Необходимо реализовать функции сохранения и восстановления данных.

По каждому сотруднику должна вестись история его перемещений внутри компании в рамках сессии работы программы.

**Основная структура программы «Три кита»**

Программа реализована с помощью классов и векторов.

В программе присутствуют три класса :

* Worker – класс сотрудник, в котором хранится поле имена сотрудников и методы , необходимые для работы с этим полем
* Division – класс подразделение, работает аналогично классу Worker
* Shell – класс оболочка. Это самый главный класса в программе, в котором в качестве полей выступают два вектора , хранящие классы сотрудников и подразделения. В качестве методов здесь выступает весь функционал программы, т.е функции создания , удаление перемещения и т.д.

Описание кс

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Класс сотрудники - Worker** | | | | | | | |
|  | | | | | | | |
| **Смысловое описание класса**: хранение поле имени сотрудника и методы , необходимые для работы с этим полем | | | | | | | |
| **№ п/п** | **Содержание свойства** | | **Тип данных и название** | | | **Примечание** | |
|  | Имя сотрудника | | string **name**; | | |  | |
|  | История сотрудника | | vector <string> **history**; | | |  | |
|  | | | | | | | |
| Методы класса **Worker** | | | | | | | |
| **№ п/п** | **Содержание метода** | **Тип метода** | | **Прототип метода** | | | **Примечание** |
|  | Создание сотрудника без параметров | конструктор (конст.) | | **Worker**(); | | | Создание сотрудника |
|  | Добавление в историю имени сотрудника | конструктор | | **Worker**(string n): name(n) ; | | | Построение |
|  | Удаление сотрудника | Деструктор | | ~**Worker**() | | |  |
|  | Получить имя сторудника | Метод | | string **get\_name**(); | | |  |
|  | Получить истории | Метод | | string **get\_history**(int i); | | |  |
|  | Получить размера вектора истории | Метод | | int **get\_history\_size**(); | | |  |
|  | | | | | | | |
| Класс подразделение **Division** | | | | | | | |
| **№ п/п** | **Содержание свойства** | **Тип данных и название** | | | **Примечание** | | |
|  | Имя подразделения | **name\_div**; | | |  | | |
|  | Вектор указателей на класс | **vector <Worker\*> workers**; | | | Нужен для работы с именами сотрудников | | |
|  | | | | | | | |
| Методы класса **Division** | | | | | | | |
| **№ п/п** | **Содержание метода** | **Тип метода** | | **Прототип метода** | | | **Примечание** |
|  | Создание подразделения без параметров | конструктор (конст.) | | **Division**(); | | |  |
|  | Создание подразделения c параметром | конструктор | | **Division**(string n) : name\_div(n) | | | Параметр имя подразделения |
|  | Удаление подразделения | Деструктор | | ~**Division**() | | |  |
|  | Получить имя подразделения | Метод | | string **get\_div**(); | | |  |
|  | Установка имени подразделения | Метод | | void set\_div(string n) | | |  |
|  | Добавление сотрудника | Метод | | void add\_worker(Worker\* w) | | |  |
|  | Получение имени сотрудника | Метод | | Worker\* get\_worker(int i) | | |  |
|  | Удаление сотрудника | Метод | | void del\_worker(int i) | | |  |
|  | Получение размера подразделения | Метод | | int get\_div\_size() | | | Кол-во рабочих в подразделение |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | | | |
| Класс оболочка **Shell** | | | | | |
| **№ п/п** | **Содержание свойства** | **Тип данных и название** | | **Примечание** | |
|  | Вектор класса рабочий | vector <Worker> workers | | Нужен для работы с сотрудниками | |
|  | Вектор класса подразделение | vector <Division> divis; | | Нужен для работы с подразделениями | |
|  | | | | | |
| Методы класса **Shell** | | | | | |
| **№ п/п** | **Содержание метода** | **Тип метода** | **Прототип метода** | | **Примечание** |
|  | Создание подразделения без параметров | конструктор (конст.) | Shell() | |  |
|  | Создание подразделения c параметром | конструктор | Shell(const Shell& o) | |  |
|  | Добавление сотрудника | Метод | void addemp() | |  |
|  | Вывод списка сотрудников | Метод | void listemp() | | Список сотрудников в компании |
|  | Увольнение сотрудников | Метод | void delemp() | |  |
|  | Добавление подразделения | Метод | void add\_division() | |  |
|  | Удаление подразделения | Метод | void del\_division() | |  |
|  | Список подразделений | Метод | void list\_division() | | Печатает список подразделений в компании |
|  | Добавление сотрудника в подразделение | Метод | void add\_worker\_div() | | Добавление сотрудника в уже заранее свозданное подразделение с помощью команды add\_division() |
|  | Просмотр списка сотрудников в подразделении | Метод | void list\_div\_worker() | | Печатает список сотрудников в конкретном подразделении |
|  | Поиск сотрудника по компании и подразделениям в компании | Метод | void search\_worker\_div() | |  |
|  | Переформирование уже созданного подразделения | Метод | void rename\_div() | | Меняет имя уже созданного подразделения |
|  | Перемещение сотрудника из одного подразделения в другое | Метод | void move\_worker\_div() | |  |

В int main() с помощью команды help печатается весь функционал программы, как правильно вводить команды и что та или иная команда делает . Сделано для простоты обращения с программой.

if (command == "help") {

cout << "---EMPLOYEE---" << endl;

cout << "addemp - add new employee" << endl;

cout << "listemp - list of employers" << endl;

cout << "delemp - delete employee" << endl;

cout << "---DIVISION---" << endl;

cout << "add\_div - add new division" << endl;

cout << "list\_div - list of divisions" << endl;

cout << "del\_div - delete division" << endl;

cout << "add\_div\_emp - add employee to division" << endl;

cout << "list\_div\_emp - list employees of division" << endl;

cout << "search\_emp\_div - search employee in divisions" << endl;

cout << "rename\_div - reform division" << endl;

cout << "move\_emp\_div - move employee to new division" << endl;

}

**ИТОГ :** реализовано почти все задание , за исключением объединения подразделений и истории сотрудника по компании.

Старался писать код максимально понятно и разборчиво.

С уважением, Курганов Никита. ИУ8-24.