Práctica IV

Apartado 1 [2,5 puntos]:

Implementar las siguientes clases que representan los distintos roles de los trabajadores de una empresa cualquiera:

- Responsable área
- Empleado
- Becario

```
Becario
                                                  Empleado
                                                  private:
private:
                                                        std::string nombre;
      std::string nombre;
                                                        int IDempleado;
      int IDempleado;
                                                        std::string nombreResponsable;
      std::string nombreResponsable;
                                                        shared_ptr<Becario> BecarioAsignado;
public:
                                                  public:
      Becario();
                                                        Empleado();
      Becario(string nom,int id,string jefe);
                                                        Empleado(string nom,int id,string jefe);
      std::string getResponsable() const
                                                        void setBecario(shared_ptr<Becario>);
                                                        std::string getResponsable() const
```

```
Trabajador
Responsable
private:
      std::string nombre;
                                                                         private:
      int IDempleado;
                                                                                ????
      vector<hared_ptr<Empleado>> Equipo;
      string NombreProyecto;
       int HorasProyecto:
                                                                         public:
public:
                                                                                ????
      Responsable();
      Responsable(string nom,int id,string nomProyecto);
      void setProyecto(string);
      std::string getProyecto() const;
      void setEquipo(vector<hared_ptr<Empleado>>);
      vector<hared_ptr<Empleado>> getEquipo();
      int getHorasProyecto();
```

Las 3 clases deben heredar de la clase base trabajador. Deberás "recolocar" los miembros (variables y atributos) de las clases derivadas que creas oportuno para hacer un correcto uso del mecanismo de herencia.

Apartado 2 [2,5 puntos]:

Añadir e implementar los siguientes miembros en la clase que consideréis más oportuna:

- int HorasRealizadas; (se trata de una variable entera donde cada uno de los trabajadores irá apuntando las horas realizadas)
- void setHorasRealizadas(int h); (se trata de un método setter que escribe el tiempo dedicado por el trabajador)
- void CalculaHorasProyecto(void), dicho método calcula el número de horas realizadas en un proyecto por el equipo asignado al responsable. En el caso de ser superior a HorasProyecto (el número de horas que el departamento comercial ha presupuestado al cliente) el método debe lanzar una excepción que notifique esta situación.

Apartado 3 [1 punto]:

Crear 3 funciones que creen respectivamente un Empleado, un Responsable y un Becario. Dichas funciones deben devolver un puntero inteligente y utilizar los parámetros de entrada que consideres oportunos.

- std::shared ptr<Becario> CrearBecario(......)
- std::shared ptr<Empleado> CrearEmpleado(......)
- std::shared ptr<Responsable> CrearResponsable(......)

Apartado 4 [4 puntos]:

Crea un programa en el que se pregunte de forma cíclica por las siguientes opciones:

- Dar de alta un trabajador:
 - Preguntar por el tipo de trabajador (Becario/Empleado/Responsable)
 - En función del tipo de rol preguntar por la información necesaria para su registro en el sistema
 - Añadirlo en la variable contenedor que consideréis oportuna, utilizando la función que corresponda del apartado 3.
- Crear un equipo:
 - Preguntar por número de integrantes
 - Preguntar por nombre de los integrantes
 - Comprobar si dichos nombres están en la variable contenedor correspondiente
 - En caso afirmativo añadir Empleado a contenedor Equipo
 - En caso negativo continuar preguntando por otros integrantes hasta completar el número indicado.
 - Una vez completado el equipo preguntar por el nombre del responsable que va a gestionar su trabajo
 - En caso de existir en el sistema asignarle el equipo.
- Buscar una persona en el sistema:
 - Preguntar por el nombre de la persona a buscar
 - Buscar en las variables contenedor disponibles

- En caso de encontrar a dicha persona mostrar la información que almacena haciendo uso de la sobrecarga del operador <<.
- Salir del programa