Progetto P2 - QtSalaries

Idea di progetto

Tool di automazione per il calcolo degli stipendi e per il tracking dei giorni feriali di un azienda di supermercati.

Meta-documentazione

Convezioni

Nomenclatura

Possibili convenzioni di nomenclatura:

- I nomi assegnati alle classi sono in lingua inglese con lettera maiuscola.
- I nomi assegnati alle variabili sono in lingua inglese composti unicamente da lettere minuscole. Se composti devono essere intervallati dal carattere
- I nomi dei metodi in lingua inglese. Se il metodo e' composto da più termini il primo ha lettera minuscola, tutte le iniziali dei successivi hanno lettera maiuscola
- Devono essere dichiarate le direttive d'uso che si vogliono usare per i metodi appartenenti a un namespace (es. se voglio non dover riscrivere std::endl ogni volta che lo utilizzo, devo notificare che viene dichiarata quella direttiva d'uso e in quale file);

Documentazione

- Vengono omessi nomi e descrizioni dei getter e setter per i campi dati (il nome del getter o del setter corrisponde perfettamente con il nome del campo dati di cui vengono recuperati i contenuti)

Realtà aziendale

Struttura gerarchica interna

Il punto vendita viene visto come una collezione di dipendenti che possono aderire alla proposta contrattuale part time o full time, unita alla presenza di un unico direttore per il punto vendita. E' possibile che un dipendente part time venga "upgradato" a dipendente full time.

Valori delle proposte contrattuali

Part Time

Nome	Valore
Ore lavoro giornaliero	4 ore
Compenso orario	7.5 euro
Giorni feriali accumulabili mensilmente	1 giorno
Bonus ore straordinario	0.5 euro

Full Time

Nome	Valore
Ore lavoro giornaliero	8 ore
Compenso orario	8 euro
Giorni feriali accumulabili mensilmente	1 giorno
Bonus ore straordinario	1.0 euro

Valori relativi al singolo ruolo

Ftemployee

No bonus

Ptemployee

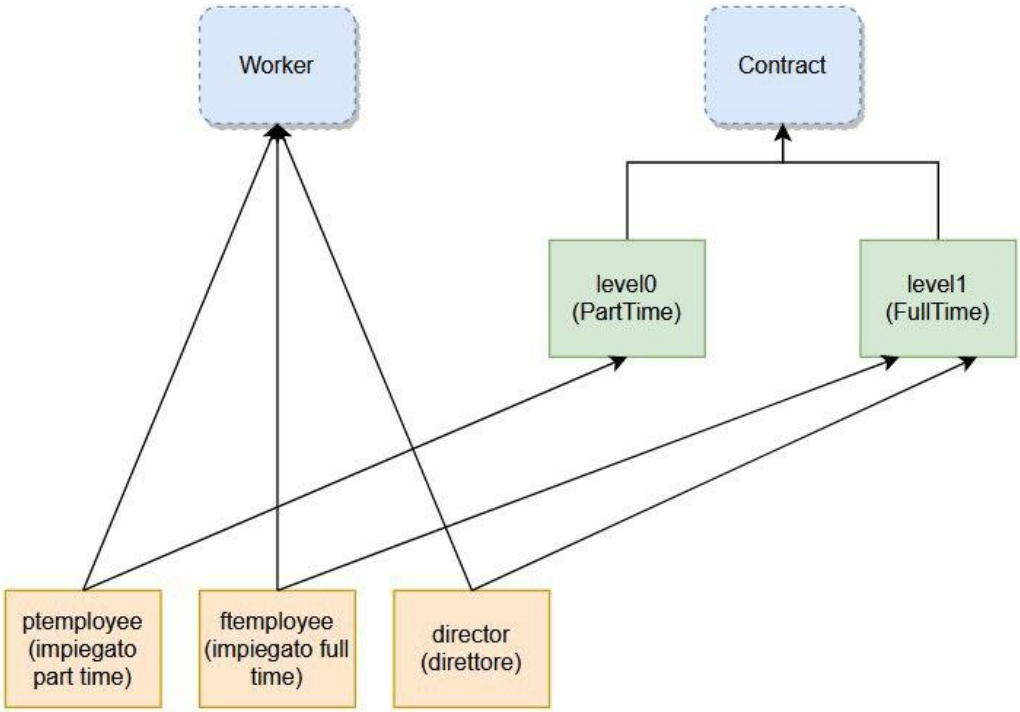
Nome	Valore
Salario base bonus	100
Giorni ferie bonus	1

Director

Nome	Valore
Salario base bonus	400
Giorni ferie bonus	1.5
Bonus aggiuntivo su ore straordinario	1

Gerarchia di classi

Visione d’insieme



La gerarchia si compone di 2 basi astratte:

- `Worker` : fornisce le funzionalità di base per lo store delle informazioni relative ad un dipendente aziendale qualsiasi e dichiara le funzioni necessarie al calcolo degli stipendi, ma non le implementa e le delega alle classi figlie.
- `Contract` : fornisce i dati essenziali relativi alle tipologie di contratto proposte dall'azienda.

La classe `Contract` deriva due classi concrete di contratto, ovvero:

- `level0` : un contratto part-time
- `level1` : un contratto full time

La classe `Worker` deriva tre classi concrete di lavoratori. Ciascuna classe di lavoratore deriva anche da una delle sottoclassi di `Contract` . Idealmente, un lavoratore (con le informazioni che esso porta con sè) unito alla sua tipologia di contratto permettono di calcolare stipendio e ferie accumulate.
Nel nostro caso, avremo:

- `ptemployee` : un impiegato part time
- `ftemployee` : un impiegato full time
- `director` : un direttore

Classi nello specifico

> ⚠ **N.B.**
>
> Leggi su meta-documentazione per vedere quali metodi sono omessi in questa sezione

Worker

```
class worker
{
private:
    std::string name;
    std::string sname;
    int last_month_worked_days;
    int last_month_worked_hours;
    double last_month_salary;
    int vac_acc;

public:
    worker(std::string, std::string);
    virtual ~worker();

    std::string getName() const;
    void setName(const std::string&);

    std::string getSname() const;
    void setSname(const std::string&);

    int getLastMonthWorkedDays() const;
    void setLastMonthWorkedDays(const int&);

    int getLastMonthWorkedHours() const;
    void setLastMonthWorkedHours(const int&);

    double getLastMonthSalary() const;
    void setLastMonthSalary(const double&);

    int getVacAcc() const;
    void setVacAcc(const int&);
    void resetVacAcc();

    void updateWorkData(const int&, const int&);

    virtual double calcBaseSal() const =0;
    virtual double calcBonus() const =0;
    virtual void updateVacAcc() =0;
    virtual double calcFullSal(const int&, const int&) =0;
};
```

Campi dati

Nome	Descrizione
name	Il nome del lavoratore
sname	Il cognome del lavoratore
last_month_worked_days	I giorni lavorati nell'ultimo mese

Nome	Descrizione
last_month_worked_hours	Le ore lavorate nell'ultimo mese
last_month_salary	L'ultimo stipendio percepito
vac_acc	I giorni di ferie accumulati dal lavoratore

Metodi

Nome	Descrizione
resetVacAcc()	riporta il valore dei giorni di ferie accumulati a 0
updateWorkData()	aggiorna il valore dei giorni lavorati e delle ore lavorate l'ultimo mese
calcBaseSal()	calcola lo stipendio base senza tenere conto di eventuali ore di straordinario
calcBonus()	calcola lo stipendio bonus accumulato nelle ore di straordinario lavorate
updateVacAcc()	aggiorna il valore relativo alle ferie accumulate
calcFullSal()	calcola lo stipendio completo (base+bonus) e aggiorna le ferie accumulate (è sostanzialmente un wrapper per gli altri metodi)

Contract

```
1  class contract
2  {
3  private:
4      int work_hours;
5      double salary;
6      double salary_bonus;
7      int vac_per_month;
8
9  public:
10     contract(int, double, double, int);
11     virtual ~contract();
12     virtual int getWorkHours() const;
13     virtual double getSalary() const;
14     virtual double getSalaryBonus() const;
15     virtual int getVacPerMonth() const;
16     virtual std::string getContractType() const=0;
17 };
```

Campi dati

Nome	Descrizione
work_hours	le ore di lavoro giornaliero previste
salary	guadagno orario

Nome	Descrizione
salary_bonus	il guadagno extra da aggiungere a salary per le ore di straordinario
vac_per_month	i giorni di ferie guadagnati dopo un mese di lavoro

Metodi

Nome	Descrizione
getContractType()	riporta una stringa associata al tipo di contratto che ne dichiara il nome (es. Part-time, Full-time). Si tratta di un getter virtuale

Level0

```
class level0: public contract {
private:
    static std::string pt_contr_name;
public:
    level0();
    virtual std::string getContractType() const override;
};
```

Campi dati

Nome	Descrizione
pt_contr_name	nome contratto impostato a “Part Time”

Level1

```
class level1: public contract {
private:
    static std::string ft_contr_name;
public:
    level1();
    virtual std::string getContractType() const override;
};
```

Campi dati

Nome	Descrizione
ft_contr_name	nome contratto impostato a “Full Time”

Ptemployee

```
class ptemployee: public worker, public level0 {
```

```
public:
    ptemployee(std::string, std::string);
    virtual double calcBaseSal() const override;
    virtual double calcBonus() const override;
    virtual double calcFullSal(const int&, const int&) override;
    virtual void updateVacAcc() override;
};
```

> **N.B.**
>
> viene solo fatto ovverride dei metodi della classe base worker, è un impiegato che non ha alcun bonus oltre a quelli contrattuali

Ftemployee

```
class ftemployee: public worker, public level1 {
private:
    static double ft_base_bonus_salary;
    static int ft_bonus_vac_day;
public:
    ftemployee(std::string, std::string);

    double getBaseBonusSalary() const;
    int getBonusVacDay() const;

    virtual double calcBaseSal() const override;
    virtual double calcBonus() const override;
    virtual double calcFullSal(const int&, const int&) override;
    virtual void updateVacAcc() override;
};
```

Campi dati

Nome	Descrizione
ft_base_bonus_salary	aumento sul salario di base di un impiegato
ft_bonus_vac_day	giorni di vacanza bonus guadagnati mensilmente

Director

```
class director: public worker, public level1 {
private:
    static double dir_base_bonus_salary;
    static double dir_bonus_bonus_salary;
    static int dir_bonus_vac_day;

public:
    director(std::string, std::string);
```

```
 9
10      double getBaseBonusSalary() const;
11      double getBonusBonusSalary() const;
12      int getBonusVacDay() const;
13
14      virtual double calcBaseSal() const override;
15      virtual double calcBonus() const override;
16      virtual double calcFullSal(const int&, const int&) override;
17      virtual void updateVacAcc() override;
18  };
```

Campi dati

Nome	Descrizione
dir_base_bonus_salary	aumento sul salario di base di un direttore
dir_bonus_bonus_salary	aumento sul bonus orario per le ore di straordinario
dir_bonus_vac_day	giorni di vacanza bonus guadagnati mensilmente