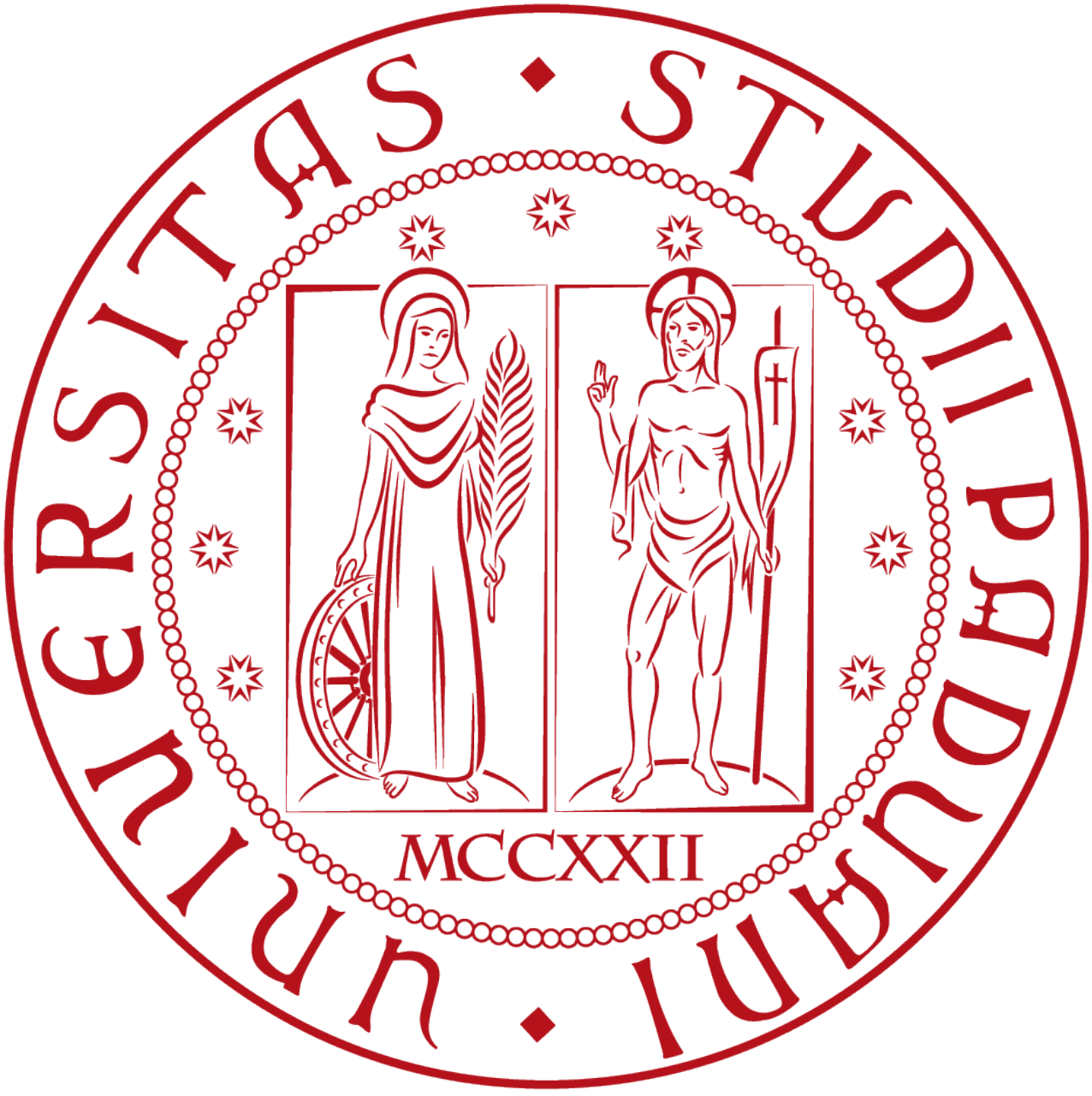
Relazione Progetto Programmazione ad Oggetti
a.s. 2020/2021



Tommaso Allegretti: 1201247 – tommaso.allegretti@studenti.unipd.it

Relazione Progetto di Programmazione ad Oggetti – QRental

Gerarchia

RentalObject: classe base astratta che indica un veicolo noleggiabile tramite il software Qrental.

Attributi:

- unsigned int rentTime: indica il numero di giorni per cui effettuare il noleggio
- unsigned int id: indica il la posizione che occupa all'interno del contenitore dei prodotti
- QString name: indica il nome del veicolo
- unsigned int km: indica i kilometri percorsi dal veicolo
- unsigned int rating: indica il voto dato dall'impresa di noleggio
- double dailyPrice: indica il costo base giornaliero del veicolo

ShowroomVehicle: classe derivata da RentalObject, classe istanziabile che indica un veicolo da esibizione.

Attributi:

- QString showName: indica il nome con cui riferirsi all'eventuale luogo di esposizione del veicolo
- bool rentRoom: indica se si vuole prenotare anche una stanza per esposizioni
- unsigned int specN: indica il numero di spettatori che ci aspetta prendano parte all'esposizione
- bool showSec: indica se si vuole assumere dello staff di security per l'esposizione

OffRoadVehicle: classe derivata da RentalObject, classe non istanziabile che indica un veicolo da fuoristrada.

Attributi:

- unsigned int trackCredits: i clienti che noleggiano veicoli da fuoristrada possono anche prenotare delle giornate nella pista interna dell'impresa, questo attributo segna il numero di giornate prenotate

RoadVehicle: classe derivata da RentalObject, classe non istanziabile che indica un veicolo da strada.

Attributi:

- double roadFee: tassa per i veicoli circolanti in strada, è associata al rischio che i clienti corrono guidando in strade trafficate, è assegnato in base alla pericolosità del veicolo
- unsigned int passengersN: indica il numero di passeggeri che il cliente pensa si trasportare col veicolo noleggiato
- bool highwayPass: indica se il cliente si avvale dell'offerta per le autostrade offerta dall'impresa

ORMotorcycle: classe derivata da OffRoadVehicle, istanziabile in quanto implementa i metodi virtuali dichiarati in precedenza. Rappresenta una moto da offroad.

Attributi:

- bool prevInjuries: il cliente è tenuto a specificare se ha subito infortuni seri in passato
- QString raceName: indica il nome con cui riferirsi all'eventuale gara a cui il veicolo prenderà parte

Atv: classe derivata da OffRoadVehicle, istanziabile in quanto implementa i metodi virtuali dichiarati in precedenza. Rappresenta un atv o quad.

Attributi:

- bool cleaningServ: indica se si vuole usufruire del servizio di lavaggio
- bool changeTires: indica se si vuole usufruire del servizio cambio gomme per avere pneumatici nuovi al momento del ritiro del veicolo

RoadMotorcycle: classe derivata da RoadVehicle, istanziabile in quanto implementa i metodi virtuali dichiarati in precedenza. Rappresenta una moto da strada.

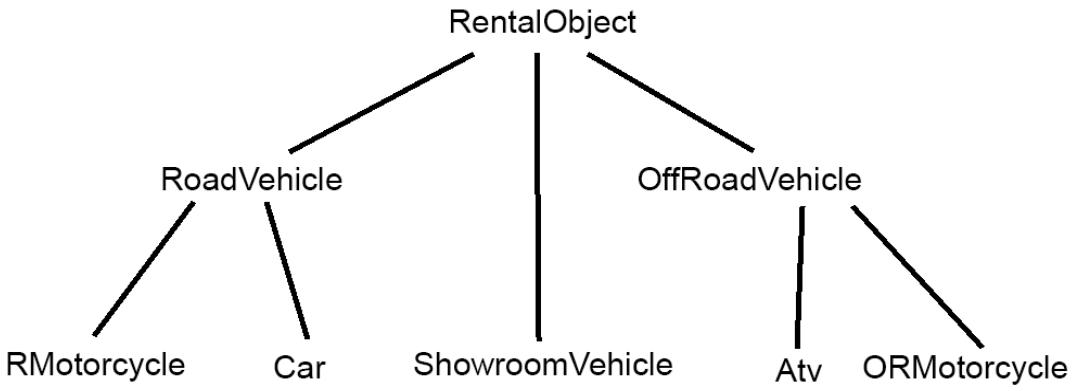
Attributi:

- unsigned int nStates: indica il numero di stati che si ha intenzione di viaggiare attraverso durante il periodo di noleggio, serve all'impresa di noleggio per tutelarsi da eventuali spese si trasporto per recuperare il veicolo

Car: classe derivata da RoadVehicle, istanziabile in quanto implementa i metodi virtuali dichiarati in precedenza. Rappresenta una macchina.

Attributi:

- bool ac: indica la presenza di aria condizionata all'interno del veicolo
- bool autoWind: indica la presenza di finestrini automatici
- bool gps: indica la presenza di un sistema di GPS



Chiamate Polimorfe

All'interno del codice sono presenti 3 metodi virtuali che utilizzano il polimorfismo:

- virtual double **getRentAmount()** const;
  Questo metodo serve per il calcolo del totale speso per il noleggio di un veicolo per un determinato periodo di tempo, è necessario che questo metodo sia polimorfo perchè ogni tipo di oggetto segue delle proprie regole su come calcolare il costo del noleggio.

- virtual double **getRefundAmount()** const;
  Questo metodo serve per il calcolo del reso che ci si aspetta di ottenere in caso si voglia cancellare il noleggio in anticipo, anche in questo caso ogni tipo di oggetto segue delle proprie regole sul calcolo del reso

- virtual RentalObject* **clone()** const;
  Questo è un metodo di clonazione utile per quando si aggiunge un noleggio alla lista dei noleggi effettuati

Classi Container e DeepPtr
Come da requisito del progetto sono state create 2 classi, una classe templatizzata di puntatori polimorfi smart e una classe templatizzata contenitrice per memorizzare i suddetti puntatori polimorfi.

Sviluppo
Per lo sviluppo è stato usato il sistema operativo Windows, Qt 5.9.5 e il compilatore MinGW 5.3.0 32bit.

Esecuzione del programma
Per compilare correttamente potrebbe essere necessario scaricare alcune librerie tramite il comando:

sudo apt-get install qt5-default

Inoltre è presente il file RentalSW.pro insieme al progetto, necessario per la compilazione tramite comando qmake, che dichiara QT += widgets.

Divisione Ore

- Analisi e ricerca 14 ore
- Sviluppo gerarchia 8 ore
- Sviluppo classe obbligatorie 7 ore
- Sviluppo GUI 18 ore
- Debugging 11 ore
- Relazione 4 ore
- **Totale** 62 ore