Relazione di Programmazione ad Oggetti

McQt

Fast-food

Studente: Berselli Marco

Matricola: 1073582

Anno: 2015/2016

Sistema Operativo di sviluppo: Windows 10

Versione Qt: Desktop QT 5.3

Versione Compilatore: MinGW 4.8.2 32bit

1. Scopo del progetto

McQt e un progetto che riguarda la creazione di un programma che generi uno scontrino di un fast-food a partire dagli oggetti passati (Patatine, Bibita, Panino e Menù), composti o per ereditarietà o per inclusione dalla classe Prodotto.

Il programma creato e stato pensato per essere gestito da una persona all'interno del fast-food che genera lo scontrino in base alle richieste del cliente.

Ogni scontrino generato avrà un'intestazione, i prodotti che sono in vendita, che saranno stampati con la quantità, il prezzo a singolo prodotto, il prezzo totale e il loro relativo dettaglio se necessario. Dopo aver stampato tutti i prodotti all'interno di Scontrino stampo il prezzo totale, il numero totale di prodotti in vendita inseriti, il numero di scontrini effettuati nella giornata ed inoltre avrà una data e un orario in cui mi indica quando è stato emesso.

Patatine deriva da prodotto, cioè ha nome, un prezzo base e una quantità, in aggiunta rispetto al prodotto c'è una taglia(size) del prodotto che si può ordinare e che ne modifica il prezzo. Non ha per ora nessuna derivata.

Bibita deriva anche essa da prodotto e ha anche essa una taglia ma ha come derivate Acqua e Birra, che sono delle Bibite "speciali" perché hanno una taglia diversa è un calcolo del prezzo in base alla taglia diverso da Bibita.

Panino invece è composto da una lista non polimorfica di prodotti ed ha come derivate Hamburger, Toast e Vegetariano. La differenza principale tra panino e le sue derivate sta che Panino può aggiungere qualsiasi Prodotto disponibile alla lista. Derivate hanno invece un numero limitato di Prodotti che possono essere modificati e inseriti così non si va a "snaturare" la concezione del Panino creata. Inoltre le derivate hanno anche un prezzo scontato del 10% sulla somma totale dei prodotti all'interno del panino.

ES: Chiedo un panino con l'hamburger e ci tolgo il pane e l'hamburger.

Menù è una classe base astratta, perché non voglio che gli utenti decidano cosa inserirci dentro, ma voglio che scelgano un menù già impostato dalle derivate. Menù contiene un panino, una bibita e una patatina. Ha come derivate MacCarne che è composto: da una derivata di panino, cioè hamburger, da una bibita e da una patatina. MacVegetariano invece è composto da: un Panino Vegetariano, un acqua, ed una patatina.

I Menù, non permettono di modificare gli oggetti, tranne per MacCarne che può scegliere il nome della bibita. In compenso MacCarne ha uno sconto di un euro sul totale dei prodotti inseriti all'interno e di cinquanta centesi su MacVegetariano.

2. Descrizione della gerarchia polimorfica

Il progetto ha 3 gerarchie polimorfiche: Prodotto, Panino e Menù. Ognuna di queste gerarchie andrà inserita a pieno o parzialmente, come nel caso di Prodotto, dentro la classe contenitore Scontrino.

1. Prodotto:

Questa gerarchia di classi è composta da Prodotto, Patatina, Bibita, Acqua e Birra.

La classe Prodotto è la classe fondamentale per tutto il programma, perché è la base di altre classi che verranno aggiunte nello scontrino, infatti compone per incapsulamento panino e compone per ereditarietà bibita e patatina.

Prodotto è composto da:

- Un Nome di tipo: string, che mi dice come si chiama l'oggetto,
- Un prezzo base (tipo: double), che mi rappresenta il prezzo dell'oggetto,
- Una Quantità di tipo: int, che mi rappresenta la quantità del prodotto.

I Metodi pubblici sono:

- Costruttore, che nel caso in cui non si passano i parametri, vengono usati quelli di default. Inoltre se i parametri prezzo base e quantità vengo passiti come un valore negativo, il primo viene settato al prezzo minimo, cioè 0.01 Euro, mentre il secondo viene settato alla quantità minima, cioè 1.
- Distruttore virtuale lasciato di default standard;
- std::string getNome() const, ritorna il nome dell'oggetto,
- void setNome(std::string), modifica il nome dell'oggetto, in base al parametro passato,
- double getPrezzoBase() const, che ritorna il prezzo Base dell'oggetto,
- int getQuantita() const, ritorna la quantità dell'oggetto,
- void setQuantita(int), modifica la quantità in base al parametro intero passato.
- void sommaQuantita(const Prodotto&) controlla se gli oggetti sono uguali, se è così somma la quantità dell'oggetto in base alla quantità del parametro.
- void togliQuantita(const Prodotto&) controlla se gli oggetti sono uguali, se è così toglie la quantità dell'oggetto in base alla quantità del parametro, se la quantità del prodotto diventa inferiore a zero questa viene modificata a zero.
- double calcolaPrezzo()const è un metodo virtuale che mi ritorna il prezzo cioè la moltiplicazione tra prezzo base e quantità,
- bool operator==(const Prodotto &) const è un metodo virtuale e mi ritorna true se sia il prezzo base e il nome degli oggetti sono uguali, altrimenti ritorna false.
- bool operator!=(const Prodotto &)const è un metodo virtuale e mi ritorna true se o il prezzo base e/o il nome degli oggetti sono diversi, altrimenti ritorna false.

Patatina è derivata in modo pubblico da Prodotto, ed essa fa parte di scontrino. La classe permette di scegliere da parte del consumatore diverse taglie di patatine, ed è composto da:

- Una size(taglia) di tipo std::string, si riferisce alla grandezza del prodotto.

I metodi pubblici sono:

- Un costruttore che ha come parametri quelli di Prodotto e la size. Se la size è diversa da "S"(taglia piccola) o "M" (taglia media) o "L" (taglia grande) o "XL" (taglia extra grande) viene modificata a "S".
- std::string getSize() const ritorna la size dell'oggetto,
- void setSize(std::string) const , modifica la size in base alla stringa passata
- double prezzoSize() const, ritorna un prezzo in base alla moltiplicazione tra la size dell'oggetto ed il prezzo base. Il metodo è virtuale.
- double calcolaPrezzo() const, ritorna la moltiplicazione tra prezzoSize e la quantità.
- bool operator==(const Prodotto &) const ritorna true se gli oggetti sono uguali sia nel nome, nel prezzo base e nella size, altrimenti ritorna false;
- bool operator!=(const Prodotto &) const ritorna true se gli oggetti sono diversi o nel nome e/o nel prezzo base e/o nella size, altrimenti ritorna false;

Bibita è derivata in modo pubblico da Prodotto, ed essa fa parte di scontrino. La classe permette di scegliere da parte del consumatore diverse taglie della bibita, ed è composto da:

- Una size(taglia) tipo std::string si riferisce alla grandezza del prodotto .

I metodi pubblici sono:

- Un costruttore che ha come parametri quelli di Prodotto e la size. Se la size è diversa da “S”(taglia piccola) o “M” (taglia media) o “L” (taglia grande) viene modificata a “S”.
- std::string getSize() const ritorna la size dell’oggetto
- void setSize(std::string) const , modifica la size in base alla stringa passata
- double prezzoSize() const, ritorna un prezzo in base alla moltiplicazione tra la size dell’oggetto ed il prezzo base. Il metodo è virtuale.
- double calcolaPrezzo() const, ritorna la moltiplicazione tra prezzoSize e quantità.
- bool operator==(const Prodotto &) const ritorna true se gli oggetti sono uguali sia nel nome, nel prezzo base e nella size, altrimenti ritorna false;
- bool operator!=(const Prodotto &) const ritorna true se gli oggetti sono diversi o nel nome e/o nel prezzo base e/o nella size, altrimenti ritorna false;

Acqua deriva da Bibita ed ha una size diversa rispetto alla superclasse. Infatti la size può essere “0.5L” o “1L” che mi indica i litri disponibili. Vengono ridefiniti il costruttore in cui se la size è diversa da 0.5L o 1L allora viene modificata a 0.5L. Inoltre viene ridefinito il metodo prezzosize che in base alla size mi ritorna 1euro se l’acqua è da 0.5 litri o 2 euro se è da 1litro.

Birra anche essa ha una size diversa infatti la birra può essere da 0.33 litri (0.33L) oppure da 0.66 litri (0.66L) ed nel costruttore se viene passata diversa da quelle citate sopra verrà settata a 0.33litri.

Viene ridefinito il metodo prezzosize che in base al prezzo base e alla size ritornerà il prezzo. Se è 0.33 litri ritorna il prezzo del prezzo base. Se invece è 0.66 litri ritornerà il prezzo base moltiplicato per 1.5.

Panino contiene una lista di prodotti implementata con i puntatori smart, il nome del panino e la quantità. Inoltre ha una classe iteratore che “itera” il nodo attraverso lo smartp rendendo più semplice l’implementazione dei metodi. La classe ha 3 derivate, Hamburger, Toast e Vegetariano. Panino dà la possibilità di poter creare il proprio panino come si vuole partendo dai prodotti.

I metodi sono:

- bool lunghezzaUguale(const Panino&, const Panino&) è un metodo statico inserito nella parte privata, che mi dice se due Panini hanno la lunghezza uguale. Ritorna true se è vero altrimenti mi ritorna false,
- Costruttore che costruisce il panino dandogli un nome ed una quantità, quindi all’inizio l’oggetto non ha prodotti inseriti al suo interno,
- std::string getNome() const ritorna il nome del panino,
- void setNome(std::string) modifica il nome del panino,
- int getQuantita() const , ritorna la quantità del panino,
- void setQuantità(int), modifica la quantità del prodotto,
- void stampadettaglio() const, stampa il dettaglio dei prodotti all’interno del panino,
- void sommaQuantità(const Panino&), prende come parametro un altro panino e se sono uguali ne modifica la quantità sommandola,
- void togliQuantita(const Panino&), prende come parametro un altro panino e se sono uguali ne modifica la quantità togliendola,
- Distruttore virtuale di panino lasciato di default,
- Prodotto& getProdotto(const Prodotto&) ritorna il prodotto del panino in base in base al prodotto passato,
- void aggiungiProdotto(const Prodotto&) aggiunge il prodotto passato per parametro all’interno di panino. Se il prodotto di Panino era già stato inserito precedentemente ne modifica la quantità sommandola,
- void rimuoviProdotto(const Prodotto&) toglie il prodotto di panino se il prodotto passato ha la quantità uguale o maggiore alla quantità del primo. Altrimenti sottrae la quantità del parametro passato al prodotto inserito,

- double prezzoAlPezzo() const ritorna il prezzo del panino in base alla somma totale di tutti i prodotti passati,
- void calcolaPrezzo() const ritorna il prezzoAlPezzo moltiplicato per la quantità,
- bool operator==(const Panino&) ritorna true se sono uguali i nomi e tutti i prodotti tra il panino passato e l'oggetto, altrimenti ritorna false,
- bool operator!=(const Panino&) const ritorna true se sono diversi o i nomi o i prodotti inseriti o entrambi tra panino passato come parametro e l'oggetto
- iteratore begin() const, ritorna un oggetto iteratore che punta al primo nodo della lista del panino,
- iteratore end() const, ritorna un oggetto iteratore che punta allo smartp che ha il puntatore che punta a 0, cioè nessun nodo,
- Prodotto& operator[](const iteratore&) const, prende come parametro un oggetto iteratore che punta ad un nodo, operator[] fa in modo che ritorni un prodotto.

Hamburger, classe derivata da panino, che a differenza di panino si presenta come una classe più "rigida" perché inserisce un numero limitato di prodotti e dà la possibilità di modificare il panino in maniera limitata:

- Costruttore inserisce all'interno del panino un nome ("Hamburger") e quantità passata per parametro, successivamente inserisce i prodotti all' interno del panino: Pane, Hamburger, Pomodoro, Insalata e sotto aceti.
- void aggiungiProdotto(const prodotto&) aggiunge i prodotti se sono pomodoro, insalata o sotto aceti, con una quantità massima di 1. Quindi questi prodotti possono arrivare fino a 2 di quantità massima.
- void rimuoviProdotto(const Prodotto&), può rimuovere solo pomodoro insalata e sotto aceti, li può togliere completamente dal panino oppure può lasciarne una sola quantità, quindi come minimo si può giungere ad un panino composto da pane ed hamburger.
- double prezzoAlPezzo() const, ritorna il prezzoAlPezzo del panino scontato del 10 %.

Vegetariano ha gli stessi metodi di Hamburger solo che ho come prodotti: pane, pomodoro, insalata e mozzarella. Di questi oggetti posso modificare pomodoro mozzarella e insalata aggiungendo una unità per ciascuno di loro ma non posso rimuovere dal panino pane, mozzarella e pomodoro. Il prezzo al pezzo viene calcolato come in Hamburger, cioè il prezzoAlPezzo di panino meno il 10%.

Toast come Hamburger e Vegetariano nel corpo del costruttore inserisce i prodotti che sono: pane tostato, prosciutto cotto e sottilette. Posso aggiungere il 1 quantità del prosciutto cotto e sottile e posso aggiungere come prodotto opzionale al massimo 2 quantità di salsa ai funghi. Per la rimozione non posso togliere più di una quantità di prosciutto cotto e sottilette, mentre posso rimuovere completamente la salsa ai funghi. Come per Hamburger e Panino il prezzoAlPezzo viene calcolato attraverso il prezzoAlPezzo di Panino meno il 10%.

Menù:

Menù è un contenitore che contiene dei puntatori ad degli oggetti che sono Panino, Bibita e Patatina. Il Menù offre ai clienti del fast-food la possibilità di ottenere dei prodotti in vendita già preselezionati ad un prezzo inferiore però non offre la possibilità di cambiarli o modificarli, tranne nella derivata MacCarne che si dà la possibilità di scegliere la bibita. E' composto da una stringa che contiene il nome, un intero che contiene la quantità e 3 puntatori: uno a Panino, uno a Bibita e uno a Patatina. Questa classe è astratta quindi non può essere dichiarata dalle altre classi. Le sue classi derivate sono MacCarne e MacVegetariano.

Menù ha come metodi:

- Costruttore che inserisce il nome i puntatori e la quantità. Se la quantità passata è inferiore a 1, la quantità di menù viene modificata ad uno.
- Costruttore di copia, che fa puntare ai puntatori degli oggetti costruiti di copia dal Menù passato. Quindi non copia i puntatori.

- Menu& operator= (const Menu&) prima distrugge gli oggetti puntati poi ne crea di nuovi di copia dal menu passato.
- Distruttore virtuale che chiama la delete sui puntatori, così si evita Garbage.
- std::string getNome() const ritorna il nome del menù.
- const Bibita& getBibita() const, ritorna la bibita puntata, l'oggetto ritornato è costante così non può essere modificato.
- const Panino& getPanino() const ritorna il panino puntato, ritorna l'oggetto costante così non può essere modificata.
- const Patatina& getPatatina() const ritorna l'oggetto patatina, ritorna come costante così non può essere modificato.
- int getQuantita() const ritorna la quantità del menù.
- void setQuantita() modifica la quantità del menù.
- void sommaQuantita(const Menu&), se l'oggetto del metodo è uguale al parametro passato ne viene sommata la quantità.
- void togliQuantita(const Menu&), se l'oggetto del metodo è uguale al parametro passato ne viene tolta la quantità, se la quantità è minore uguale a zero viene impostato uguale a zero.
- double *prezzoAlPezzo*() const, metodo virtuale puro.
- double calcolaPrezzo() const, ritorna la moltiplicazione di prezzoAlPezzzo per la quantità
- bool operator ==(const Menu&) const, ritorna true se il menù sono uguali cioè sono uguali: il nome, il panino, la bibita, le patatine, altrimenti false.
- bool operator !=(const Menu&) const, ritorna la true se il menù sono diversi cioè sono diversi: il nome o/e il panino o/e la bibita o/e le patatine, altrimenti ritorna false.

MacCarne prende come parametro il nome della bibita e la quantità. Viene costruito un menù che ha come nome "MaCcarne", la bibita di taglia media con il nome del parametro passato, un Hamburger, una patatina di taglia piccola di nome" Fritte". Prezzo al pezzo di Menù viene "sovrascritto" (ovverride) e ritorna somma del prezzo totale degli oggetti meno 0.5 euro;

MacVegetraiano prende come parametro la quantità. Viene costruito un menù che ha come nome "MacVegertariano" e ha come bibita viene assegnata la classe Acqua di nome "Naturale", come panino un Vegetariano e una patatina di taglia piccola di nome "Lesse". prezzoAlPezzo di Menù viene sovrascritto(ovverride) e ritorna la somma del prezzo totale degli oggetti meno un euro.

3.Descrizione dell'codice polimorfo

Scontrino è una classe contenitore che contiene una lista polimorfica, cioè una lista di smartp composta da un puntatore a nodo.

La classe nodo ha delle classi derivate che sono nodoMenu, nodoPanino, nodoBibita e nodoPatatina. Ogni classe derivata da nodo rispettivamente contiene un puntatore cioè Menu per nodoMenu, Panino per nodoPanino, Bibita per nodBibita e Patatina per nodoPatatina.

La classe scontrino ha un metodo, operator[], che ritorna un puntatore al nodo puntato, grazie alla classe iterator, che mi itera la lista.

Essendo che è stato ritornato un puntatore a nodo allora si può usare il polimorfismo per ottenere le classi derivate che contengono i rispettivi puntatori, che a loro volta possono usare il polimorfismo. Quindi se il tipo dinamico è diverso da quello statico allora chiama i metodi della classe che sono stati marcati virtuali e ritorna gli oggetti puntati che corrispondono al tipo dinamico.ES: Se durante la costruzione di nodoPanino viene passato un oggetto ti tipo Hamburger, il puntatore per sottotipaggio combacia con l'oggetto passato. Durante una chiamata a un metodo virtuale a runtime viene controllato il tipo dinamico e viene chiamato il metodo virtuale della classe con tipo dinamico.

Quindi quando ritorno un puntatore a nodo, grazie ai dynamic_cast so' che tipo di nodo derivato è e posso sfruttarlo per chiamare all' suo interno il puntatore che chiamerà i metodi virtuali e non.

4.Descrizione GUI

Alla apertura della GUI sulla destra si visualizza una textarea in cui c'è lo scontrino vuoto. Man mano che si aggiungono oggetti all' interno di scontrino si visualizzeranno eventualmente se necessari anche i dettagli all'interno dell'oggetto inserito. Inoltre, sempre sulla destra, c'è un bottone, pagamento, che una volta cliccato creerà un pdf contente lo scontrino(Salvato nella cartella in cui è compilato il programma). Successivamente lo scontrino verrà eliminato e verrà creato uno nuovo scontrino vuoto.

Sulla sinistra invece abbiamo un tab con quattro pagine in cui ci sono Panino, Menù, Bibite e Patatine che a loro volta sono divise in due parti la parte superiore riguarda l'inserimento mentre la parte inferiore riguarda la rimozione le modifiche, se sono permesse.

La parte di inserimento in panino avviene attraverso una comboBox in cui è possibile scegliere il panino.

Se il panino è un panino a scelta (ovvero un panino standard, non derivato) sarà vuoto, dentro al groupbox Prodotto verranno visualizzate delle checkBox con il nome del prodotto che si vuole inserire. Inoltre di fianco a ciascuna checkbox ci sarà uno spinbox che mi indica la quantità da inserire di quel determinato prodotto. L'inserimento avverrà quando verrà cliccata la checkbox relativa al prodotto con una quantità pari a uno successivamente basterà aggiungere o togliere la quantità desiderata con lo spinbox. Se si vuole togliere un prodotto sul panino basterà cliccare di nuovo il checkbox.

Nel groupbox affianco a quello chiamato Prodotto c'è ne' uno chiamato Aggiungi in cui sarà possibile dare un nome al panino scelto e modificare la quantità del panino che si vorrà inserire. Infine c'è un bottone chiamato aggiungi che mi permette di aggiungere il panino all'interno dello scontrino.

Per i panini come Toast, Hamburger e Vegetariano sarà possibile inserire il panino senza che si selezioni un prodotto perché vengono costruiti di default. Eventualmente se si vuole aggiungerne una quantità superiore di prodotti all'interno del Panino allora lo sì può fare selezionando il prodotto e aumentandone la quantità attraverso lo spinbox di fianco. Ovviamente non si possono aggiungere ulteriori quantità su prodotti per cui non sono modificabili come pane o Hamburger per il panino Hamburger.

Per la rimozione c'è un comboBox in cui vengono raccolti gli oggetti panino inseriti all'interno di scontrino, dove è possibile modificare i prodotti all'interno dei panini.

All'interno del gruopbox Rimuovi Prodotto si potrà rimuovere un prodotto, se l'oggetto lo permette, cliccando sul checkbox relativo. Si può anche modificare la quantità del prodotto interagendo con lo spinbox affianco al checkbox relativo. Come per Inserimento ci sono delle restrizioni che non mi permettono di rimuovere o aggiungere o modificare la quantità oltre una certa soglia per alcuni Prodotti.

Di fianco al groupbox Rimuovi Prodotto c'è ne' un altro che si chiama Aggiungi Prodotto che permette attraverso delle checkbox di aggiungere i prodotti non presenti all'interno di panino.

Di fianco alla combobox della rimozione abbiamo uno spinbox che mi permette di modificare la quantità del panino e un bottone con scritto rimuovi che mi permette di togliere il prodotto da scontrino.

Per Menù o Patatine o Bibite invece delle chekbox e dei spinbox all' interno dei groupbox abbiamo dei radio button che ci permettono di modificare l'oggetto.

In MacCarne e Bibita sarà possibile modificare il nome della bibita che si vuole aggiungere. In aggiunta Bibita avrà la possibilità di modificarne la size. Stessa cosa accade per Acqua e Birra.

Patatine invece ci dà la possibilità di cambiare solamente la size.

Ovviamente restano in alterate le funzioni dei spinbox che modificano la quantità e quelle dei bottoni che rimuovono o inseriscono gli oggetti.