Programmazione di applicazioni software
CdL in Ingegneria dei Sistemi Informativi
a.a. 2020/2021
Docente: Prof. Andrea Prati

Prova pratica del 27 luglio 2021

Ricordarsi di consegnare il codice sorgente di tutte le classi e della classe main salvando il progetto Code::Blocks in Z:

Simulazione negozi di vestiti

Si scriva un programma per simulare la vendita di Vestiti. I Vestiti vengono venduti in Negozi. Un **Negozio** è definito da un nome, un indirizzo e la lista dei Vestiti attualmente presente nel Negozio. Per la classe Negozio si devono implementare due metodi:

- Un metodo cercaVestito che riceve in input il nome del Vestito (si veda dopo) e restituisce la quantità di Vestiti di quel tipo presenti nel Negozio (0 se il Negozio non ha quel Vestito);
- Un metodo faiConto che riceve in input il nome del Vestito, la quantità che un Cliente vuole comprare e il Cliente stesso e restituisce il prezzo totale in euro come preventivo di spesa (-1 se il Negozio non ha la merce).

Nota: Un Negozio può avere nella lista anche più volte lo stesso Vestito.

Un **Vestito** è caratterizzato da un nome (nome del modello, ad esempio "Camicia casual blu"), una taglia (memorizzata come stringa, es: "XL", "XXL", "L", ecc.), un prezzo in euro e un colore principale (stringa). Si devono gestire tre tipologie di Vestiti:

- **Pantalone** che è caratterizzato anche da un campo booleano ad indicare se è fallato (cioè difettoso). Se è fallato, il Pantalone viene venduto con il 30% di sconto
- **Camicia** che ha anche un campo booleano ad indicare se è in saldo ed in tal caso viene venduto al 15% di sconto
- **Gonna** che è caratterizzata anche da un campo float di sconto (es: 0.18 ad indicare uno sconto del 18%) che però viene applicato solo ai Clienti femmine

Sia la classe base Vestito che le tre derivate, devono prevedere un metodo calcolaPrezzo che riceve come input il Cliente (per gestire il caso delle Gonne) e ritorna il prezzo a valle di eventuali sconti. Si supponga che Vestiti di tipo diverso (Pantalone, Camicia o Gonna) non possano avere lo stesso nome.

I Negozi sono frequentati da Clienti. Un **Cliente** è caratterizzato da un nome e cognome e un sesso (memorizzato come singolo carattere). La classe Cliente deve prevedere due versioni (in overloading) di un metodo compra:

- Una versione prevede come input il nome del Vestito e la quantità che si intende comprare e restituisce il Negozio il cui prezzo totale è inferiore e il prezzo totale
- Una versione prevede come input il nome del Vestito, la quantità da comprare e il nome del Negozio da cui si vuole comprare, e restituisce il prezzo totale o -1 se il Negozio non ha disponibile quella quantità di quel Vestito

Supponendo che qualunque sia il prezzo totale il Cliente compri i Vestiti nel Negozio in cui li ha trovati, i Vestiti comprati (nella quantità indicata come parametro di compra) devono essere rimossi dalla lista di quel Negozio.

Tempo MASSIMO a disposizione: 180 minuti

Ogni classe, dove opportuno, dovrebbe avere, oltre a getter e setter, un metodo per la visualizzazione dei suoi dati sullo schermo.

Il main deve inizializzare una serie (non ne servono molti) di oggetti delle varie classi. Si suggerisce di organizzare i vari oggetti della classe Negozio nel main in una qualche struttura dati (non necessariamente dinamica) per comodità.

Si considerano punti aggiuntivi per chi dovesse utilizzare i thread che gestiscano più Clienti contemporaneamente ed in particolare il metodo compra della classe Cliente.

Sviluppare le classi che permettono una soluzione basata sulla programmazione Object Oriented al problema.

Tempo MASSIMO a disposizione: 180 minuti