Zadanie 1

Programovanie v jazyku C#, 2022/2023

V tomto zadaní je úlohou namodelovať správanie ľudí pri kúpe vysnívaných produktov v rôznych obchodoch. Model obsahuje reprezentáciu produktov, položiek v obchode, obchod, zľavové karty, ako aj kupujúcich a ich túžbu istý produkt kúpiť. Pre riešenie zadania bol pre vás vytvorený projekt s deklarovanými triedami, ich štruktúrou a metódami. Môžete vytvoriť aj vlastné riešenie, avšak musíte dodržiavať rozčlenenie projektu a verejné rozhranie tried, teda musíte implementovať minimálne metódy, ktoré sú popísané v tomto súbore.

Samotný nákup bude simulovaný cez kupujúceho človeka, ktorý postupne navštívi jednotlivé obchody a podľa svojich prianí a podľa svojho rozpočtu si kúpi niekoľko tovarov. Implementujete pritom rôzne stratégie nákupu. Kľúčovú rolu pri kúpe hrá cena, pričom okrem ceny samotného produktu budete brať do úvahy maržu predajcu, prípadné zľavy ako aj zľavovú kartu, ktorú nakupujúci má. Triedy a ich funkcionalita sú bližšie popísané nižšie.

Pri implementácii dodržujte nasledovné zásady:
- nepoužívajte žiadne knižnice okrem štandardných modulov jazyka C#,
- všetky členské premenné musia byť privátne alebo protected a pri práci s nimi dodržujte enkapsuláciu (žiadny priamy prístup k hodnotám mimo triedy),
- nepoužívajte rozhrania (interface),
- podľa potreby môžete zadefinovať metódy okrem tých uvedených v tomto dokumente.

Štruktúra projektu:

- `Buyers`
  - `Buyer.cs` – abstraktná trieda reprezentujúca všeobecné správanie kupujúceho
  - `Card.cs` – zľavová karta pre istý typ produktu
  - `EagerBuyer.cs` – kupujúci, ktorý kúpi produkt v prvom obchode, kde ho nájde
  - `CheapBuyer.cs` – kupujúci, ktorý hľadá najvýhodnejšiu cenu
  - `LazyBuyer.cs` – kupujúci, ktorý nenavštívi každý obchod
  - `Wish.cs` – reprezentuje želanie človeka kúpiť si niektorý tovar
- `Shop`
  - `Article.cs` – reprezentuje položku, teda tovar, ako ho vidí obchod
  - `Product.cs` – reprezentuje tovar
  - `ProductType.cs` – typ tovaru, na ktorý môže platiť zľavová karta
  - `Store.cs` – obchod, ktorý ponúka niekoľko položiek
- `Program.cs` – iba pre testovanie vášho riešenia, obsah hodnotený nebude

Na ďalších stranách nasleduje popis jednotlivých tried a metód, ktoré potrebujete implementovať. Poradie tried predstavuje odporúčané poradie implementácie, avšak v niektorých prípadoch budete potrebovať vopred implementovať niektorú funkcionalitu. Odporúčame si preto najprv prečítať celý popis, aby ste pochopili súvislosti medzi jednotlivými triedami, ako aj ich spoluprácu pri reprezentácii niektorých zložitejších procesov.

Približný počet riadkov formátovaného kódu riešenia je 500 (s pomocnými funkciami a gettermi a settermi). `Main` funkcia obsahuje ukážkové príklady použitia kódu.

ProductType

Definuje enumeračný typ, ktorý vymenuje rôzne typy produktov, ktoré budú v ponuke obchodov a môžu byť kúpené ľuďmi. Súbor nepotrebujete upravovať.

Product

Trieda definuje tovar, ktorý má názov (`name`), cenu (`price`) a kategóriu (hodnota z enumeračného typu `ProductType`). Všetky členské premenné triedy sú privátne, ich hodnoty sa nastavia v konštruktore. Trieda okrem toho prepisuje štandardnú metódu `ToString()`, ktorá vráti stringovú reprezentáciu tovaru, a môžete ju využívať pri testovaní riešenia.

Article

Trieda definuje položku a je rozšírením triedy `Product`, pričom položka z pohľadu obchodu popisuje produkt (`product`), maržu (`incomeRate` – o koľko percent drahšie predáva obchod produkt ako je samotná cena produktu), aktuálnu zľavu (`discount` – o koľko percent obchod predáva produkt lacnejšie ako je pôvodná cena produktu s maržou) a počet dostupných kusov v obchode (`quantity`).

Trieda obsahuje konštruktor, ktorý nastavuje hodnotu členských premenných na základe parametrov. Potrebujete umožniť vytvorenie inštancie s defaultnou nastavenou hodnotou zľavy na 0. Okrem toho potrebujete implementovať nasledujúce metódy:

- `GetPrice()` – vráti cenu jedného kusu položky, pričom k cene samotného produktu pripočíta maržu a odčíta zľavu. Napríklad ak samotný produkt stojí 100 eur a obchod si stanovil maržu 10% a žiadnu zľavu, položka bude stáť 110 eur (100 + 100 * 0,1). Keby obchod ponúkal zľavu 5%, pri výpočte finálnej ceny najprv pripočítate maržu a následne aplikujete zľavu: (100 + 100 * 0,1) * (1 – 0,05) = 104,5.
- `SetDiscount(double discount)` – nastaví aktuálnu zľavu pre položku.
- `SetIncomeRate(double discount)` – umožňuje zmeniť maržu pre danú položku.
- `GetCategory()` – vráti kategóriu produktu.
- `CanSell(int count)` – vráti hodnotu `true`, ak z položky je dostupný potrebný počet kusov (daný ako parameter `count`), v opačnom prípade vráti `false`.
- `Sell(int count)` – metóda aktualizuje položku pri predaji niekoľkých kusov (daný ako parameter `count`): ak je z položky dostupný potrebný počet kusov, aktualizuje počet (odčíta počet predaných kusov) a vráti hodnotu `true`, v opačnom prípade vráti `false`.

Card

Trieda `Card` reprezentuje zľavovú kartu, ktorá umožňuje zákazníkom kúpiť niektoré typy tovarov za zľavnenú cenu. Karta je definovaná zoznamom typov produktov, pre ktoré je platná (zoznam `validFor`) ako aj zľavou, ktorú môže zákazník uplatniť pri použití karty (`discount` – desatinné číslo medzi 0 a 1). Trieda už definuje konštruktor, potrebujete implementovať metódu `IsValidFor(ProductType category)`, ktorá vráti hodnotu `true`, ak je karta platná pre tovary typu `category`; inak vráti `false`.

Wish

Trieda `Wish` reprezentuje túžbu zákazníka kúpiť istý počet kusov niektorého produktu. Vnútorná štruktúra triedy je daná členskými premennými `product` (tovar, ktorý chce človek kúpiť) a `count` (koľko kusov chce kúpiť). Trieda definuje úplný konštruktor, ktorý na základe parametrov nastaví hodnotu členských premenných.

Za úlohu máte doplniť definíciu metódy `Update(int bought)`, ktorá aktualizuje objekt pri kúpe niekoľkých kusov tovaru. Návratová hodnota metódy naznačuje, či bola túžba splnená, teda či už zákazník kúpil vysnívaný počet kusov. Základná logika metódy je nasledovná:

- ak hodnota parametra `bought` je vyššia ako počet kusov, ktoré zákazník pôvodne chcel kúpiť, metóda vráti `false` (do mínusu nikdy nepôjdeme)
- v opačnom prípade aktualizujte členskú premennú `count`, ktorá bude reprezentovať počet kusov, ktoré zákazník ešte chce kúpiť
- ak túžba zákazníka bola splnená (už nechce kúpiť ďalšie kusy daného produktu), metóda vráti `true`, v opačnom prípade vráti `false`.

Store

Trieda `Store` predstavuje obchod, ktorý ponúka niekoľko rôznych produktov, resp. položiek. Obchod je reprezentovaný zoznamom ponúkaných produktov (zoznam objektov typu `Article` v členskej premennej `articles`), percentuálnym vyjadrením marže (`incomeRate` – využije sa pri vytváraní položiek) a celkovým príjmom (`income`). Zoznam položiek sa inicializuje na prázdny zoznam, konštruktor triedy nastavuje maržu podľa parametra a celkový príjem na 0.

Do triedy potrebujete implementovať nasledovné metódy:

- `GetIncome()` – metóda vráti celkový príjem obchodu.
- `StockProduct(Product product, int count, double discount)` – metóda aktualizuje ponuku obchodu, pričom ide o pridanie tovaru `product` do zoznamu `articles`. Nezabudnite na to, že v obchode bude každý tovar dostupný cez položku, teda potrebujete vytvoriť objekt typu `Article` na základe parametrov, ktoré metóda dostane. Ako maržu pre položku použite nastavenú maržu obchodu. Ak pre daný produkt už existuje položka v zozname `articles`, iba aktualizujte túto položku pridaním daného počtu kusov (parameter `count`), pričom sa môže zmeniť aj marža a zľava platná na položku.
- `ApplyDiscountForCategory(ProductType category, double discount)` – metóda nastaví novú hodnotu zľavy (parameter `discount`) pre všetky položky ponúkané obchodom, ktoré reprezentujú tovar zo zadanej kategórie (parameter `category`).
- `HasProduct(Product product, int count)` – metóda skontroluje, či z istého produktu (parameter `product`) je v ponuke potrebný počet kusov (parameter `count`). Vráti hodnotu `true`, ak obchod má minimálne daný počet kusov, a `false` v opačnom prípade.
- `GetPrice(Product product, Card card)` – metóda vypočíta a vráti cenu jedného kusu produktu pre zákazníka, ktorý má zľavovú kartu (parameter `card`). Ak daný produkt nie je v ponuke obchodu, metóda vráti 0. Ak produkt v ponuke je, zistite cenu položky, ktorá predstavuje tovar. Kartu môže zákazník aplikovať na základe typu tovaru: každá karta poskytuje zľavu pre isté kategórie produktov – ak žiadaný produkt (parameter `product`) patrí do niektorej z týchto kategórií, aplikujte zľavu stanovenú kartou, v opačnom prípade zľavu aplikovať nemôžete a zákazník zaplatí celú sumu.
- `SellProduct(Product product, int count, Buyer buyer)` – metóda reprezentuje predaj `count` kusov tovaru `product` zákazníkovi `buyer`, všetky tieto údaje metóda dostane ako parameter.

V metóde urobte nasledovné kroky:

1. zistite, či je vôbec dostupný potrebný počet kusov: ak nie, tak ukončite vykonávanie metódy a k žiadnej zmene nedôjde.
2. vypočítajte cenu jedného kusu pomocou metódy `GetPrice()`, od zákazníka potrebujete vyžiadať jeho kartu.
3. skontrolujte, či si zákazník môže dovoliť kúpiť daný počet kusov za vypočítanú cenu (v triede `Buyer` bude dostupná metóda) – ak nie, metóda končí

4. ak zákazník má finančné prostriedky pre zaplatenie daného počtu kusov, tovar mu predáte, pričom:
   4.1. aktualizujete položku (znížite počet dostupných kusov)
   4.2. aktualizujete celkový príjem obchodu
   4.3. zákazníkovi predáte tovar (zavoláte príslušnú metódu z triedy `Buyer`)

Buyer

`Buyer` je abstraktná trieda, ktorú využívame pre zabezpečenie spoločného rozhrania triedy pre rôzne typy nakupujúcich. Každý zákazník má kartu (`card` – môže byť aj `null`), rozpočet (`budget` – limituje produkty, ktoré človek dokáže kúpiť), zoznam vysnívaných produktov (`wishList`) a zoznam kúpených produktov (`hasBought`). Je dôležité uvedomiť si, že kým v zozname `wishLisht` bude túžba kúpiť tri kusy rovnakého produktu reprezentovaná jedným objektom, v zozname `hasBought` sa bude daný produkt vyskytovať trikrát.

Konštruktor triedy je už pripravený, nastavuje kartu a rozpočet, a zvyšné dva zoznamy inicializuje ako prázdne. Potrebujete doplniť implementáciu nasledovných metód:

- `AddToWishList(Product product, int count)` – pridá do zoznamu `wishList` zámer človeka kúpiť si `count` kusov tovaru `product`.
- `GetCard()` – vráti referenciu na zľavovú kartu nakupujúceho.
- `GetBudget()` – vráti dostupný rozpočet zákazníka.
- `GetBought()` – vráti zoznam nakúpených tovarov (zoznam môže obsahovať duplikáty).
- `CanAfford(double price, int count)` – metóda vráti hodnotu `true`, ak si zákazník môže dovoliť kúpiť `count` kusov istého tovaru, ak jeden kus stojí `price`. V opačnom prípade vráti `false`.
- `Inquire(Store store, Wish wish)` – metóda vráti `true`, ak si v obchode `store` človek dokáže kúpiť potrebný počet kusov istého tovaru tak, aby splnil svoju túžbu `wish`. V opačnom prípade metóda vráti `false`.
- `FindWish(Product product)` – metóda nájde túžbu zákazníka kúpiť istý tovar `product`, a vráti ju. Ak človek daný tovar kúpiť nechce, metóda vráti `null`.
- `SellTo(Product product, int count, double price)` – metóda reprezentuje kúpu `count` kusov tovaru `product`, kde cena jedného kusu je `price`. V metóde potrebujete:
  1. aktualizovať túžbu zákazníka kúpiť daný produkt, keďže istý počet kusov už nakúpil (môžete predpokladať, že daný tovar skutočne chcel kúpiť)
  2. aktualizovať rozpočet
  3. aktualizovať zoznam nakúpených produktov

Trieda definuje aj abstraktnú metódu `VisitStores(List<Store> stores)`, ktorú budú definovať podtriedy a bude reprezentovať stratégiu kupujúceho pri návšteve obchodov.

EagerBuyer

Trieda reprezentuje „greedy“ zákazníka, ktorý pri postupnej návšteve obchodov si kúpi tovar v prvom obchode, kde ho nájde v dostatočnom množstve bez ohľadu na cenu. Vnútorná reprezentácia je celá zdedená od nadtriedy `Buyer`. Implementujte metódu `VisitStores(List<Store> stores)`, ktorá definuje túto stratégiu:

- zákazník postupne navštívi jednotlivé obchody a v každom zistí, či by nemohol naplniť niektorú svoju túžbu;
- musíte skontrolovať, či daný produkt je dostupný v danom počte kusov v obchode a či si ho zákazník môže dovoliť pri cene, ktorú určuje obchod (aj po použití zľavovej karty);
- ak všetky kontroly prejdú, zákazník si tovar kúpi;

- nezabudnite na to, že túžba bude aktualizovaná a teda objekty v zozname `wishList` sa budú počas behu programu meniť – ošetrite, aby zákazník nenakúpil viac kusov istého produktu, ako pôvodne chcel.

Poznámka: V tejto aj ostatných podtriedach `Buyer` sa pre jednoduchosť nebudeme zaoberať prípadom, keď by zákazník svoju túžbu naplnil v rôznych obchodoch. Teda buď dokáže kúpiť potrebný počet kusov v jednom obchode, alebo ich vôbec nekúpi.

`CheapBuyer`

Trieda reprezentuje zákazníka, ktorý si chce vysnívaný tovar nakúpiť za čo najvýhodnejšiu cenu a práve preto pre každú svoju túžbu vyberie obchod, ktorý daný tovar ponúka za najnižšiu cenu a následne tovar kúpi u nich. Bude vyberať iba z obchodov, ktoré ponúkajú potrebný počet kusov.

Poznámka: Pri hľadaní najlacnejšieho obchodu nezabudnite na to, že ak metóda `GetPrice()` z triedy `Store` vráti 0, znamená to, že daný tovar nie je v ponuke a nie to, že je zadarmo!

`LazyBuzer`

`LazyBuyer` predstavuje nakupujúceho, ktorý nenavštívi všetky obchody, ale iba prvých niekoľko z nich, a práve preto je vnútorná reprezentácia rozšírená o členskú premennú `storeCount`, ktorá je celé číslo reprezentujúce maximálny počet obchodov, ktoré zákazník môže navštíviť. Okrem toho sa nakupujúci správa rovnako ako v prípade `EagerBuyer` triedy, t. j. tovar kúpi v prvom dostupnom obchode.

Poznámka: V metóde `VisitStores` ošetrite prípad, kde zoznam obchodov je kratší ako počet obchodov, ktoré je nakupujúci ochotný navštíviť (`storeCount`).