Budeme majstri (v kvíze)

Predstavte si, že sa chystáte na pub quiz, a chcete pozbierať niekoľkých vašich priateľov, aby ste sa prihlásili ako tím. Ale koho si vybrať? Našťastie máte známych v radoch organizátorov, ktorí vám dali informáciu o tom, približne z akých okruhov budú otázky na kvíze. Následne ste vašim kamošom dali cvičný test, na základe ktorého ste zistili, v akých okruhoch sa cítia doma a koľko toho vedia. Na základe týchto poznatkov potom viete implementovať model a simuláciu, ktorá vám navrhne tím, ktorý bude mať najväčšiu šancu na úspech.

Pri riešení úlohy budete znova vytvárať výpočtové modely, a precvičíte si prácu s dedičnosťou a abstraktnými triedami. Tiež získate skúsenosti s inými komponentmi ako triedy a rozhrania, a použijete rôzne údajové štruktúry v C#. Ako bonus sa naučíte aj to, ako funguje náhodnosť v tomto jazyku.

V projekte potrebujete implementovať niekoľko tried, pričom musíte dodržať štruktúru projektu ako aj namespacy:

`Program.cs` (namespace `Assignment2`) – hlavný súbor, implementujete tu dve metódy

`Player` (namespace `Assignment2.Player`) – adresár so súbormi modelujúcimi hráčov

- `QuizPlayer.cs` – základný hráč
- `KnowItAll.cs` – hráč, ktorý vie (skoro) všetko o (skoro) všetkom
- `LazyPlazer.cs` – hráč, ktorý (skoro) nič nevie o ničom

`Quiz` (namespace `Assignment2.Quiz`) – adresár so súbormi modelujúcimi kvíz a otázky

- `AbstractQuestion.cs` – abstraktná trieda so spoločnou funkcionalitou otázok
- `QuestionQuiz.cs` – trieda reprezentujúca kvíz s niekoľkými otázkami
- `QuestionType.cs` – enumerácia s kategóriami otázok
- `ShortAnswer.cs` – otázka s krátkou odpoveďou
- `TestQuestion.cs` – otázka s niekoľkými možnými odpoveďami
- `TFQuestion.cs` – otázka typu pravda/nepravda

Kostru riešenia nájdete v predpripravenom projekte. Keďže vaše riešenia prejdú automatizovanými testami, je dôležité, aby ste dodržali štruktúru projektu, najmä čo sa týka menných priestorov. Ak programujete v IDE, ktoré vytvára inú štruktúru ako vidíte v kostre riešenia, nezabudnite upraviť štruktúru pred finálnym odovzdaním.

Všetky členské premenné musia byť privátne, a nesmiete k nim pristupovať priamo mimo triedy. Členské premenné v nadtriedach môžu byť protected. Vaše riešenie môžete rozšíriť o premenné a metódy okrem tých uvedených v zadaní.

Uvedené poradie tried reprezentuje odporúčané poradie implementácie, avšak niektoré metódy sa spoliehajú na metódy iných tried, práve preto je dôležité, aby ste si na úvod prečítali celý popis úlohy.

`QuestionType`

Súbor obsahuje enumeráciu s možnými kategóriami otázok. Súbor nepotrebujete nijak upravovať. Ak niečo zmeníte, tak dbajte na to, aby naďalej boli dostupné všetky pôvodne zadefinované kategórie: `GENERAL`, `SPORTS`, `GEOGRAPHY`, `HISTORY`, `ARTS`, `LITERATURE`, `MATHS`, `PHYSICS`, `CHEMISTRY`, `ENTERTAINMENT`.

AbstractQuestion

Abstraktná trieda deklaruje niektoré spoločné vlastnosti všetkých typov otázok ako členské premenné:

- `question` (`string`) – otázka, na konkrétnej hodnote počas simulácie nezáleží;
- `category` (`QuestionType`) – kategória otázky, hodnota z `QuestionType`;
- `points` (`int`) – počet bodov, ktoré tím získa, ak na otázku odpovie správne.

Okrem toho trieda obsahuje prázdny konštruktor s parametrami zodpovedajúcimi členským premenným, je na vás, či v konštruktore implementujete nejakú funkcionalitu. Trieda tiež predpisuje abstraktnú metódu `EvaluateAnswer(string guess)`, ktorá slúži na vyhodnotenie odpovede tímu, a vráti `true` alebo `false` na základe jej správnosti. Metódu nechajte deklarovanú ako abstraktnú a bez tela, reálnu funkcionalitu neskôr implementujete v podtriedach. Trieda `AbstractQuestion` hodnotená nebude.

ShortAnswer – 0,5 bodov

Trieda `ShortAnswer` predstavuje otázku s krátkou odpoveďou. Trieda dedí od `AbstractQuestion` a je rozšírená o premennú `answer` typu `string`, ktorá reprezentuje správnu odpoveď. V triede potrebujete implementovať:

- konštruktor, ktorý správne nastaví všetky členské premenné (aj tie zdedené od nadtriedy) na základe hodnôt parametrov;
- metódu `EvaluateAnswer(string guess)`, ktorá vráti hodnotu `true`, ak odpoveď zadaná hráčom (parameter `guess`) je rovnaká ako správna odpoveď otázky. Na veľkosti písmen v odpovedi nezáleží.

TestQuestion – 1 bod

Trieda `TestQuestion` predstavuje otázku s niekoľkými možnými odpoveďami (ako pri štandardných testoch). Trieda dedí od `AbstractQuestion`, a je rozšírená o premenné `options`, čo je zoznam `string`ov a obsahuje jednotlivé možné odpovede, ako aj o celé číslo `correctId`, ktoré predstavuje index správnej odpovede z možností.

V triede implementujte:

- konštruktor, ktorý správne nastaví všetky hodnoty členských premenných (aj zdedených) na základe parametrov. Zoznam možných odpovedí inicializujte ako prázdny zoznam, index správnej odpovede nastavte defaultne na -1.
- metódu `EvaluateAnswer(string guess)`, ktorá vráti hodnotu `true`, ak odpoveď zadaná hráčom zodpovedá správnej odpovedi. Ak sa zadaná odpoveď nenachádza v zozname možností alebo je nesprávna, metóda vráti `false`.
- metódu `AddOption(string option)`, ktorá pridá možnú odpoveď do príslušného zoznamu, ustrážte pritom, aby sa možnosti neopakovali. Pri opätovnom pridaní tej istej odpovede zoznam neaktualizujte.
- metódu `SetCorrect(int id)`, ktorá nastaví index správnej odpovede, ak ten je platný (vzhľadom na pravidla indexovania prvkov zoznamu). V opačnom prípade príslušnú členskú premennú neaktualizujte.

TFQuestion – 0,5 bodov

Trieda `TFQuestion` predstavuje špeciálny typ otázky s niekoľkými možnosťami, kde na výber máme z dvoch možností: tvrdenie je pravdivé alebo nepravdivé. Trieda dedí od `TestQuestion`.

V triede implementujte:
- konštruktor, ktorý správne nastaví všetky hodnoty členských premenných (aj zdedených) na základe parametrov. Zoznam možných odpovedí inicializujte ako zoznam dvoch reťazcov *true* a *false* (v tomto poradí), index správnej odpovede nastavte na základe parametra `isTrue`, ktorý vyjadruje, či tvrdenie je pravdivé.
- v triede musí byť dostupná metóda `EvaluateAnswer(string guess)`, ktorá vráti hodnotu `true`, ak odpoveď zadaná hráčom zodpovedá správnej odpovedi a `false` naopak.

`QuestionQuiz` – 1 bod

Trieda `QuestionQuiz` reprezentuje kvíz s niekoľkými otázkami uvedených typov. Trieda je daná zoznamom otázok (s jednotným rozhraním cez triedu `AbstractQuestion`), dĺžkou (celkový počet otázok ako celé číslo) a indexom aktuálnej otázky (tiež celé číslo). Dĺžka kvízu vyjadruje konečný počet otázok, počas behu ju nepotrebujete aktualizovať.

V triede implementujte nasledovnú funkcionalitu:
- konštruktor nech inicializuje zoznam otázok ako prázdny, a nastavte zvyšné členské premenné vhodným spôsobom (podľa parametra, resp. na začiatok kvízu).
- metóda `AddQuestion(AbstractQuestion question)` má pridať do zoznamu otázok otázku, ktorú dostane ako parameter, ale iba v prípade, ak ste ešte nedosiahli celkový počet otázok v kvíze. V opačnom prípade sa zoznam neaktualizuje. Dbajte aj na to, aby zoznam neobsahoval duplikáty (stačí porovnávať na základe objektov).
- metóda `Reset()` nech pripraví kvíz na opätovné spustenie, aktualizujte teda index aktuálnej otázky.
- metóda `GetNextQuestion()` vráti ďalšiu otázku v kvíze. Metóda môže vrátiť `null` v dvoch prípadoch: kvíz ešte neobsahuje potrebný počet otázok a teda sa nedá odštartovať, alebo ste už na konci kvízu a všetky otázky boli položené. Volanie `GetNextQuestion()` priamo po volaní `Reset()` má vrátiť prvú otázku.

`QuizPlayer` – 2 body

Trieda predstavuje bežného hráča kvízov, ktorý má niekoľko svojich záujmov, v ktorých sa vyzná do istej miery. Táto skutočnosť je reprezentovaná členskou premennou `knowledgeLevel`, ktorá je slovník s kľúčmi typu `QuestionType` (teda rôzne kategórie otázok), a hodnotami typu `float`, teda percentuálne pravdepodobnosť toho, že hráč na otázku z daného okruhu odpovie správne.

Do triedy doplňte funkcionalitu:
- konštruktor nech nastaví vedomosti hráča na základe parametra.
- metóda `SetKnowledgeLevel(QuestionType category, float knowledgeLevel)` nastaví mieru vedomostí hráča pre istú kategóriu otázok. Ak sa táto kategória nachádza v slovníku, hodnotu aktualizujte. Ak kategória je nová, tak ju pridajte do slovníka aj s hodnotou. Kategóriu a mieru vedomostí dostanete ako parametre.
- metóda `GetGuess(AbstractQuestion question)` vráti hráčovu odpoveď a to nasledovne:
    - ak kategória otázky patrí medzi jeho záujmy, tak odpovie správne s pravdepodobnosťou podľa miery jeho vedomostí v danej kategórii.
    - ak kategória otázky nepatrí medzi jeho záujmy, tak odpovie náhodne. To znamená, že pre `ShortAnswer` a `TFQuestion` jeho odpoveď bude správna približne v 50%

prípadov. Pre `TestQuestion` si vyberie náhodnú možnú odpoveď (pravdepodobnosť správnej odpovede teda závisí od počtu možností).

Poznámka: Pri generovaní odpovede využívajte náhodnosť napríklad cez generovanie náhodného desatinného čísla v intervale 0—1 a porovnajte ho s pravdepodobnosťou správnej odpovede. Pri testovaní sa počíta s istou mierou tolerancie, napríklad pri 1000 opakovaní volania `GetGuess()` a pravdepodobnosťou správnej odpovede 0,8 sa očakáva cca. 800 správnych odpovedí, ale nemusí to byť presne toľko.

`KnowItAll` – 0,5 bodov

`KnowItAll` predstavuje hráča, ktorý vie správne odpovedať v kategóriách jeho záujmov. Trieda dedí od `QuizPlayer` a funkcionalitu potrebujete implementovať nasledovne:

- konštruktor ako parameter dostane zoznam záujmov hráča, mieru vedomostí nastavte na 1 pre každý z nich.
- metóda `GetGuess(AbstractQuestion question)` vracia odpoveď hráča, pričom ak sa jedná o otázku z jeho záujmov, tak vždy odpovie správne, v opačnom prípade dá náhodnú odpoveď tak, ako aj `QuizPlayer`.

`LazyPlayer` – 0,5 bodov

`LazyPlayer` je podobná `KnowItAll`, avšak predstavuje hráča, ktorý nedokáže správne odpovedať ani na otázky z jeho záujmov. Trieda dedí od `QuizPlayer` a funkcionalitu potrebujete implementovať nasledovne:

- konštruktor ako parameter dostane zoznam záujmov hráča, mieru vedomostí nastavte na 0 pre každý z nich.
- metóda `GetGuess(AbstractQuestion question)` vracia odpoveď hráča, pričom tá je vždy náhodná odpoveď vygenerovaná podobne ako pri `QuizPlayer`.

V súbore `Program.cs` potrebujete implementovať dve metódy, obe za 2 body:

- `FindBestTeam(QuestionQuiz quiz, List<QuizPlayer> allPlayers, int count)` – predstavuje situáciu, kde hľadáte `count` počet členov do vášho tímu zo zoznamu `allPlayers` pre kvíz `quiz` tak, aby ste získali čo najviac bodov. Svoje odpovede teraz zanedbajte, chcete nájsť tím, ktorý vám vyhrá titul s najväčšou pravdepodobnosťou aj keď vy nedáte ani jednu správnu odpoveď. Body za správnu odpoveď získate ak aspoň jeden člen tímu odpovie správne. Metóda vráti zoznam hráčov, kde ich počet je daný parametrom `count`. Ak `count` je väčšie číslo ako počet všetkých hráčov, tak metóda vráti `null`.
- `FindDuos(QuestionQuiz quiz, List<QuizPlayer> allPlayers, int count)` – predstavuje situáciu, kde ste organizátor kvízu, a prihlásilo sa niekoľko hráčov, ktorí ale nemajú partnera. Vašou úlohou je teda vygenerovať zoznam `count` dvojíc hráčov, ktorý následne metóda vráti (návratová hodnota je zoznam zoznamov hráčov). Ak sa z dostupných hráčov nemôžete vytvoriť `count` dvojíc, metóda vráti `null`. Dvojice chcete vytvoriť tak, aby jednotlivé dvojice získali celkovo čo najviac bodov.