AGH AGH University of Science and Technology

Zadanie projektowe

Proszę napisać własny program w języku C#, korzystając ze zdobytych w tym semestrze informacji na temat dobrych praktyk oraz zasad programowania obiektowego. Program może dotyczyć dowolnego tematu – osobom niezdecydowanym rekomendujemy tworzenie gry komputerowej (w szczególności typu gra przygodowa lub RPG) bądź też symulatora naśladującego działanie jakiegoś rzeczywistego obiektu lub systemu. W szczególności, dozwolone jest wykorzystanie tematu używanego dotychczas w zadaniach C1/C4.

Niezależnie od wybranego tematu, program powinien spełniać następujące wymagania:

- Musi się uruchamiać i być przesłany na UPEL (niespełnienie tego warunku oznacza otrzymanie 0 pkt za program).
- Powinien być dostatecznie duży – nie ma w tej kwestii twardego limitu, jednak można się spodziewać, że program zawierający mniej niż 20 klas będzie wymagał przedstawienia dodatkowych wyjaśnień podczas oddawania projektu.
- Powinien zostać napisany z użyciem interfejsów lub klas abstrakcyjnych, które będą implementowane lub po których będą dziedziczyły wybrane inne klasy.
- Istniejące hierarchie dziedziczenia powinny zawierać polimorficzne metody. Nie musi to dotyczyć absolutnie każdej klasy w hierarchii, powinno jednak występować tam, gdzie to możliwe. Zawarte metody powinny być prawdziwie polimorficzne, tj. nie powinny polegać na blokach *if* sprawdzających typ klasy.
- Należy przestrzegać zasad SOLID, a w szczególności zasady "Single-responsibility principle" – jedna klasa odpowiada za jedną rzecz.
- Program powinien zawierać przynajmniej jeden wzorzec projektowy, użyty do rozwiązania jakiegoś problemu. Wzorzec do zaimplementowania zostanie wskazany przez prowadzącego podczas zajęć poświęconych omówieniu planów projektu.

Projekt zostanie zrealizowany w trzech etapach, z których każdy będzie oceniony osobno:

Etap 1: wypełnienie i zaprezentowanie wstępnej deklaracji na temat zawartości programu. Podczas oddawania etapu 1 prowadzący zaproponuje lub ustali w rozmowie wzorzec projektowy, który będzie należało umieścić w programie. Ocena: do 10 punktów.

Etap 2: przygotowanie i zaprezentowanie częściowego diagramu UML. Diagram ten nie musi zawierać wszystkich klas, jakie docelowo znajdą się w projekcie, natomiast powinien obejmować klasy wymagane w deklaracji wstępnej z etapu 1 oraz wzorzec projektowy ustalony wcześniej z prowadzącym. Ocena: do 10 punktów, z czego 5 punktów jest za uwzględnienie na diagramie wzorca, 5 punktów za pozostałe klasy/metody.

Etap 3: oddanie działającego programu zgodnego z deklaracją i diagramem UML* oraz spełniającego warunki podane na liście powyżej. Ocena: do 40 punktów, z czego 10 punktów jest za zaimplementowanie wzorca, 30 punktów za resztę programu.
*nawet najlepszy plan czasami zawodzi – dlatego umawiamy się, że niewielkie i uzasadnione modyfikacje pierwotnych założeń nie będą stanowiły problemu.

Dodatek - wersja zadania dla chętnych: ASP.NET Core WebAPI

Osoby chętne i zainteresowane programowaniem mogą także zrealizować zadanie projektowe w formie aplikacji napisanej w technologii ASP.NET Core Web API. Jest to rodzaj aplikacji, która komunikuje się ze światem zewnętrznym za pośrednictwem ustandaryzowanych zapytań (typowo w formacie JSON) i po ich przetworzeniu zwraca określone odpowiedzi (typowo również w formacie JSON). Przykładowe API ze strony OpenWeather, w automatyczny sposób zwracający dane na temat bieżącej pogody w danym mieście (jeśli ktoś chce przetestować je samodzielnie, w celu uzyskania klucza konieczne jest założenie darmowego konta w serwisie):

https://openweathermap.org/current

W naszym przypadku nie jest konieczne udostępnianie API przez internet – wystarczy, że będzie ono działało lokalnie, na Państwa komputerze. W celu stworzenia takiej aplikacji należy samodzielnie zapoznać się z dokumentacją, przy czym szczególnie pomocny może być następujący tutorial ze strony Microsoftu (dostępne także tłumaczenie maszynowe – trzeba podmienić "en-us" na "pl-pl"):

https://docs.microsoft.com/en-us/learn/modules/build-web-api-aspnet-core/

Powyższy tutorial bazuje na środowisku Visual Studio Code, jednak osoby pracujące na Visual Studio również mogą go użyć – w momencie, w którym autorzy proszą o otworzenie terminala z poziomu VS Code, można zamiast tego otworzyć zwykły terminal (wiersz poleceń) systemu Windows lub Mac. Alternatywnie, można posłużyć się także innym tutorialem ze strony Microsoftu, który w swoich początkowych sekcjach opisuje tworzenie i uruchamianie Web API przy pomocy samego Visual Studio (kolejne sekcje zawierają już bardziej zaawansowany materiał, wykraczający poza poziom tego zadania):

https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-web-api?view=aspnetcore-5.0

Uwaga 1: pierwszy z podanych tutaj tutoriali jest łatwiejszy, jednak proszę się nie przerażać, jeżeli nie będą Państwo znali każdego z używanych w nim skrótów i terminów – wiele z nich ma charakter tylko poglądowy i nie jest potrzebne do napisania działającej aplikacji, wszystkie zaś dają się łatwo wyszukać w internecie. W razie trudności można także zwrócić się z pytaniami do prowadzącego zajęcia.

Uwaga 2: jak można zobaczyć w sekcji piątej tutorialu, dane na temat pizzy są w nim przechowywane w postaci zmiennych statycznych zawartych w klasie serwisu. Jest to oczywiście rodzaj placeholdera – w Państwa przypadku klasy serwisów mogą zamiast tego np. zapisywać i odczytywać dane z pliku.

Kryteria oceny dla zadania z Web API są takie same jak dla zwykłej wersji, z dwoma wyjątkami:

- W przypadku Web API nie ma konieczności umieszczania w aplikacji wzorca projektowego. W zamian za to, należy zadbać o prawidłowe wyodrębnienie warstw modeli, kontrolerów i serwisów, co samo w sobie jest rodzajem wzorca stosowanego przy tworzeniu aplikacji webowych. Mówiąc w uproszczeniu: modele odpowiadają za specyfikacje używanych przez nas klas z danymi, kontrolery odpowiadają za sposoby komunikacji ze światem zewnętrznym, serwisy odpowiadają za "logikę biznesową" i operacje wykonywane przez naszą aplikację w celu przetwarzania zapytań oraz zwracania wyników.
- Osoby oddające zadanie z Web API nie muszą rysować diagramu klas UML. Zamiast tego, mogą uzyskać do 10 punktów za przygotowanie dokumentacji dla swojego API – należy wyjaśnić jego ogólny cel oraz opisać każdą z metod kontrolera. Wymagane informacje dla metod: typ metody (get/post/put/delete), opis parametrów, przykładowe wywołanie, opis zwracanej odpowiedzi.
- Pozostałe kryteria z wersji podstawowej obowiązują bez zmian. Proszę przy tym zwrócić uwagę, że we wnętrzu swojego API mogą Państwo umieścić wiele różnych klas służących do innych rzeczy niż tylko zapisywanie i odczytywanie danych – mogą to być klasy, które pozwalają coś obliczać, symulować, generować, podejmować decyzje, wykonywać operacje prowadzące do końcowego wyniku.