Dziedziczenie, abstrakcja, polimorfizm

Zadanie wprowadzające

1. Napisz klasę `Osoba` zawierającą `Imię` oraz `Nazwisko`. Dodaj konstruktor ustawiający pola klasy.
2. Napisz klasę `Student` dziedziczącą z klasy `Osoba` i zawierającą dodatkowo `NrIndeksu`. Czy projekt się kompiluje?
3. Dodaj w klasie `Student` brakujący konstruktor, który ustawi wszystkie pola klasy. Aby ustawić pola nadklasy `Osoba` posłuż się jej konstruktorem.
4. Przeciąż metodę `ToString()` w obu utworzonych klasach w taki sposób, aby zwracała:
   - `Imię Nazwisko` – dla klasy `Osoba`,
   - `Imię Nazwisko [NrIndeksu]` – dla klasy `Student`.

   W klasie `Student` posłuż się metodą `ToString()` z nadklasy. Czy można to zrobić wykorzystując mechanizm rzutowania?
5. W metodzie `Main()` klasy głównej projektu stwórz 3 obiekty zgodnie z poniższym opisem:
   - obiekt klasy `Osoba` przypisany do zmiennej typu `Osoba`,
   - obiekt klasy `Student` przypisany do zmiennej typu `Osoba`,
   - obiekt klasy `Student` przypisany do zmiennej typu `Student`.

   Wywołaj na rzecz każdego obiektu metodę `ToString()` i wypisz wynik jej działania na ekran.
6. Stwórz tablicę obiektów typu `Osoba` i dodaj do niej wszystkie utworzone obiekty. Następnie w pętli wywołaj dla każdego elementu kolekcji metodę `ToString()` wypisując każdorazowo efekt jej działania na ekran.

Zadanie samodzielne

Celem zadania jest zasymulowanie prostego odtwarzacza plików muzycznych. Efektem pracy jest aplikacja konsolowa, która najpierw prosi użytkownika o podanie utworów, które mają znaleźć się na liście odtwarzania. Następnie tworzy listę i odtwarza wszystkie utwory po kolei, usuwając każdorazowo przesłuchany utwór.

1. Należy zdefiniować przynajmniej 5 gatunków muzycznych (klas), które tworzą hierarchię dziedziczenia.
2. Każdy gatunek posiada przynajmniej jedno pole nieodziedziczone, które dodaje jakieś brzmienie do danego gatunku.
3. Każda klasa posiada przeciążoną metodę `Play()`, która wypisuje na ekran tytuł i wykonawcę utworu oraz brzmienia charakterystyczne dla danego gatunku muzycznego. Metoda ta uzyskuje brzmienia pochodzące z odziedziczonych gatunków wywołując przeciążoną metodę `Play()` z nadklasy.
4. Każda klasa posiada dokładnie jeden konstruktor ustawiający wszystkie wartości pól tworzonego obiektu (pola odziedziczone mają zostać zainicjowane poprzez wywołanie konstruktora nadklasy).
5. Klasa `Player` (pełniąca funkcję odtwarzacza) zawiera metody `Add(Song song)` oraz `Remove(int songNumer)`, które służą odpowiednio do dodawania oraz usuwania utworów. Posiada również

funkcję `Play(int songNumer)`, która odtwarza utwór o danym numerze z playlisty (wywołując funkcję `Play()` tego utworu).

6. Po uruchomieniu, program w pętli prosi użytkownika o podanie gatunku utworu, tytułu oraz wykonawcy, po czym tworzy utwór podanego gatunku i dodaje go do playlisty. Po każdym wprowadzeniu utworu aplikacja pyta użytkownika, czy ten chce wprowadzić kolejny.
7. Po zakończeniu wprowadzania utworów przez użytkownika, program odtwarza wszystkie utwory w kolejności ich dodawania do listy.