Kolokwium z Programowania Obiektowego – zestaw B1

Wszystkie poniższe czynności wykonaj w ramach jednego projektu tak, aby spełniony był załączony diagram UML. Możesz dodawać w klasach pola, metody (o ile nie zaznaczono inaczej), choć staraj się by ich liczba była jak najmniejsza i nie powielała funkcji, które są już w poleceniach.

1. Stwórz projekt – aplikacja konsolowa lub WPF (przemyśl wybór, bo zmiana może być czasochłonna).
2. Stwórz klasę wyjątku BlednyNumerException. Dodaj w tej klasie domyślny konstruktor, tak aby w polu dodatkowe informacje dla tego wyjątku był przypisany komunikat "Błędny numer".
3. Stwórz klasę Pilkarz. Dodatkowe informacje:
   1. klasa ma być abstrakcyjna;
   2. przesłoń metodę ToString() tak, aby zwracała string z informacją o piłkarzu np. Imię i nazwisko: Jan Kowalski, Numer na koszulce: 12.
4. Stwórz klasy Napastnik i Obronca. Dodatkowe informacje:
   1. Konstruktory z pobranych parametrów mają inicjować kolejno pola: imie, nazwisko, numerNaKoszulce.
   2. przesłonięcie metody ToString() ma uzupełniać string z klasy wyżej odpowiednio o łańcuch "Napastnik:" lub "Obronca:".
   3. w konstruktorach klas wyrzuć wyjątek BlednyNumerException jeśli numer na koszulce napastnika ma być liczbą nieparzystą, a w przypadku obrońcy liczbą parzystą.
5. Stwórz interfejs IDruzyna.
6. Następnie stwórz klasę Druzyna. Dodatkowe informacje:
   1. pole druzyna ma być tablicą wymiarów 2x4;
   2. ta klasa ma nie zawierać żadnego jawnego konstruktora;
   3. do klasy podepnij interfejs IDruzyna i zgodnie z kolejnymi podpunktami dodaj odpowiednią implementację metod;
   4. metoda UstawNapastnika ma mieć działanie: w tablicy druzyna na miejscu o indeksie [0,pierwszy parametr metody] przypisuje odwołanie/referencję do obiektu typu Napastnik o parametrach pobranych z drugiego, trzeciego i czwartego parametru metody UstawNapastnika;
   5. metoda UstawObronce ma mieć działanie: w tablicy druzyna na miejscu o indeksie [1,pierwszy parametr metody] przypisuje odwołanie/referencję do obiektu typu Obronca o parametrach pobranych z drugiego, trzeciego i czwartego parametru metody UstawObronce;
   6. metoda CzyOK zwraca true, kiedy wszystkie elementy tablicy druzyna są jednocześnie różne od null, oraz zwraca false w przeciwnym wypadku;
   7. metoda UstawNazwe ustawia pole nazwa zgodnie z parametrem;
   8. metoda ToString() zwraca informację o zawartości pola druzyna np.  
      [0,0] Napastnik, Imię i nazwisko: Jan Nowak, Numer na koszulce: 22  
      [1,2] Obronca, Imię i nazwisko: Krzysztof Kowalski, Numer na koszulce: 17  
      Uwaga: metoda ma pomijać puste (null) elementy w tablicy. W stringu ma pojawić się też obowiązkowa informacja o pozycji w tablicy.
7. Dokończ program robiąc aplikację konsolową/wpf (zgodnie z początkowym wyborem) na operacje wejścia/wyjścia tak, aby przetestować metody z klasy Druzyna. Dodatkowe informacje:
   1. najpierw należy ustawić drużynę, po 4 napastników i obrońców,
   2. należy obsłużyć ew. wyjątki kiedy numer na koszulce jest błędny,
   3. należy obsłużyć ew. wyjątki jeśli pozycja w tablicy wychodzi poza zakres tablicy,
   4. możliwość ustawienia nazwy ma być dopiero możliwa, kiedy cała drużyna jest skompletowana (odpowiednia tablica ma wszystkie elementy różne od null), komunikat z potwierdzeniem ustawienia nazwy ma zostać wyświetlony na MessageBoxie lub konsoli.

Gotowy kod (niespakowany) prześlij jako oddzielne repozytorium do serwisu Github. Link wyślij mailem na adres piotr.jastrzebski@uwm.edu.pl lub piojas@matman.uwm.edu.pl w temacie wpisując KolokwiumPO.

Sprawdzenie będzie rozpoczęte tylko gdy kod kompiluje się. W przypadku braku kompilacji 0 pkt za całe kolokwium.

Punktacja:

* Polecenie 1 – 5 pkt
* Polecenie 2 – 5 pkt
* Polecenie 3 – 10 pkt
* Polecenie 4 – 5 pkt
* Polecenie 5 – 15 pkt
* Polecenie 6 – 10 pkt za logikę aplikację, 10 pkt za poprawne działanie kontrolek/właściwe działanie wejścia i wyjścia konsolowego, 10 pkt za obsługę wyjątków wymienionych w punkcie 5.

Przykładowe wyglądy aplikacji:



