Zadatak 1

Napomena: Klase implementirati tako da atributi imaju najniži mogući nivo pristupa (private ili protected).

Na programskom jeziku C# kreirati:

1. Apstraktnu klasu **Student** koja sadrži atribute:
   - broj indeksa,
   - ime,
   - prezime

   i sledeće javne metode:
   - konstruktor koji postavlja sve atribute,
   - apstraktni property koji vraća bodove studenta sa ispita,
   - predefinisanu metodu ToString(),
   - operatore < i > za poređenje dva objekta klase Student po broju bodova.

2. Klasu **StudentFIB** izvedenu iz klase **Student** kod kog se broj bodova na ispitu računa kao zbir bodova sa 1. kolokvijuma, 2. kolokvijuma i laboratorijskih vežbi i koja sadrži privatne članove:
   - atribute za predstavljanje broja bodova sa 1. kolokvijuma, broja bodova sa 2. kolokvijuma i broja bodova sa laboratorijskih vežbi,

   javne:
   - konstruktor koji postavlja vrednosti svih atributa.

3. Klasu **StudentSAF** izvedenu iz klase **Student** kod kog se broj bodova na ispitu računa kao zbir bodova pismenog ispita i seminarskog rada i koja sadrži privatne članove:
   - atribute za predstavljanje broja bodova sa pismenog ispita i broja bodova sa seminarskog rada,

   javne:
   - konstruktor koji postavlja vrednosti svih atributa.

4. Klasu **Kolekcija** koja sadrži atribute:
   - maksimalni broj elemenata u kolekciji,
   - trenutni broj elemenata u kolekciji,
   - niz elemenata tipa **Student**.

   U javnom delu klase definisati sledeće metode:
   - konstruktor koji postavlja maksimalni broj elemenata u kolekciji,
   - metodu za dodavanje elementa u kolekciju,
   - metodu za prikaz svih elemenata kolekcije na standardni izlaz.

5. Interfejs **Uredjivac** koji sadrži metodu za:
   - uređivanje elemenata u zadati redosled (rastući ili opadajući).

6. Klasu **UredjenaKolekcija** izvedenu iz klase **Kolekcija** koja implementira interfejs **Uredjivac** i ima atribut koji kaže u kakvom je stanju kolekcija (neuređena, uređena u rastući ili uređena u opadajući redosled) koja je inicijalno uvek postavljena na „neuređena”.

U metodi main kreirati objekat klase **UredjenaKolekcija** koja može da primi naksimalno 100 elemenata, upisati u nju 2 objekta tipa **StudentFIB** i 2 objekta tipa **StudentSAF**, urediti je u rastući redosled, a zatim izvršiti promenu uređenja. Štampati elemente kolekcije nakon svake transformacije.

Zadatak 2

Napomena: Klase implementirati tako da atributi imaju najniži mogući nivo pristupa (private ili protected).

Na programskom jeziku C# kreirati:

1. Interfejs **IVozilo** koji sadrži metode za:
   - učitavanje podataka o vozilu iz zadatog tekstualnog toka (koja prijavljuje izuzetke ukoliko učitani atributi nisu u dozvoljenim granicama),
   - upis podataka o vozilu u zadati tekstualni tok

   i property-je koji vraćaju:
   - naziv,
   - serijski broj,
   - maksimalnu brzinu vozila.

2. Klasu **Automobil** koja implementira interfejs **IVozilo** i koja ima atribut broj mesta za sedenje. Maksimalne brzine automobila su u granicama od 90 do 250 km/h.

3. Klasu **Kamion** koja implementira interfejs **IVozilo** i koja ima atribut nosivost kamiona u tonama. Maksimalne brzine kamiona su u granicama od 60 do 130 km/h.

4. Klasu **Put** koja sadrži privatne atribute: broj vozila, niz vozila pri čemu je svako vozilo tipa **IVozilo**, ograničenje brzine na tom putu (km/h) i dužinu puta (km). Javni članovi klase Put su:
   - konstruktor koji postavlja tip puta, ograničenje brzine i dužinu puta,
   - metod za dodavanje vozila na put,
   - metod za učitavanje niza vozila iz tekstualne datoteke zadatog imena,
   - metod za upis niza vozila u tekstualnu datoteku zadatog imena,
   - metod za izbacivanje vozila koje je u prekršaju. Vozilo je u prekršaju ako ne može da pređe put za zadato vreme (vreme se zadaje kao argument ove metode). Vozilo je u prekršaju i ako mu je prosečna brzina veća od ograničenja na putu. Ovaj metod za prosečnu brzinu vozila uzima slučajan broj u opsegu od 0 do maksimalne brzine vozila.

5. Delegat **UpozorenjeNosivost** koji ima kao argument maksimalnu nosivost puta u tonama i služi za upozoravanje kamiona na putu o tome. Klasa **Put** treba da ima događaj (event) ovog tipa i na taj događaj treba pretplatiti sve kamione koji se dodaju na put. U klasi **Kamion** treba implementirati metodu za obradu ovog događaja.

6. U metodi Main kreirati objekat klase Put, učitati vozila sa standardnog ulaza (ili iz datoteke), izbaciti vozila u prekršaju (maksimalno vreme na putu je 2 sata) i snimiti u datoteku "VozilaIzlaz.txt" samo vozila koja nisu u prekršaju. Ukoliko se u toku rada programa prijavi bilo kakav izuzetak treba prekinuti rad programa i na standardni izlaz prikazati odgovarajuću poruku o grešci.

Zadatak 3

Napomena: Klase implementirati tako da atributi imaju najniži mogući nivo pristupa (private ili protected).

Na programskom jeziku C# kreirati:

1. Generičku klasu **Matrica<T>** gde je parametar T struktura koja implementira interfejs **IBroj**. Klasa Matrica sadrži sledeće atribute:
   - broj vrsta, broj kolona i elemente matrice,

   i sledeće javne elemente:
   - javni konstruktor koji postavlja broj vrsta i broj kolona,
   - javni konstruktor za kopiranje,
   - javni indekser koji postavlja i vraća element matrice na zadatoji poziciji [i, j],
   - javnu metodu za prikaz elemenata matrice na standardni izlaz,
   - javni unarni operator **++** koji inkrementiera svaki od elemenata matrice,
   - javni binarni operator **+** koji radi sabiranje dve matrice i vraća novu matricu kao rezultat,
   - javni binarni operator ***** koji radi množenje dve matrice i vraća novu matricu kao rezultat.

2. Interfejs **IBroj** koji sadrži:
   - metodu za inkrementiranje broja,
   - metodu za sabiranje broja sa drugim zadatim brojem,
   - metodu za množenje broja sa drugim zadatim brojem,
   - property koji vraća nultu vrednost za odgovarajući format broja.

3. Strukturu **RacionalniBroj** koja implementira interfejs **IBroj** i predstavlja broj u formatu **brojilac/imenilac** gde su brojilac i imenilac celi brojevi. Nulom se smatra broj **0/1**.

4. Strukturu **KompleksniBroj** koja implementira interfejs **IBroj** i predstavlja broj u formatu **Re + j*Im** gde su realni i imaginarni deo realni brojevi. Nulom se smatra broj **0 + j*0**.

5. U metodi main isprobati rad ovih klasa i njihovih metoda.

Zadatak 4

Napomena: Klase implementirati tako da atributi imaju najniži mogući nivo pristupa (private ili protected).

Na programskom jeziku C# kreirati:

1. Apstraktnu klasu **Proizvod** za predstavljanje proizvoda u prodavnici čiji su atributi:
   - naziv,
   - celobrojna šifra,
   - status proizvoda (proizvod može biti u prodavnici, u magacinu ili naručen od proizvođača, definisati posebnu enumeraciju za status).

   U javnom delu klase definisati:
   - konstruktor koji inicijalizuje sve privatne atribute,
   - apstraktni property koji vraća cenu proizvoda,
   - virtualnu metodu koja u zadati tekstualni tok podataka upisuje vrednosti svih privatnih atributa klase,
   - virtualnu metodu koja iz zadatog tekstualnog toka podataka čita vrednosti svih privatnih atributa klase.

2. Klasu **ProizvodNaMerenje** izvedenu iz klase **Proizvod**.
   Atributi klase **ProizvodNaMerenje** su:
   - masa u kilogramima,
   - cena po kilogramu (cena proizvoda se dobija kao cena po kilogramu * masa).

   a javne metode su sledeće:
   - konstruktor koji postavlja sve atribute ove klase i osnovne klase **Proizvod**,
   - metoda koja u zadati tekstualni tok podataka upisuje vrednosti svih privatnih atributa ove klase i osnovne klase **Proizvod**,
   - metoda koja iz zadatog tekstualnog toka podataka čita vrednosti svih privatnih atributa ove klase i osnovne klase **Proizvod**.

3. Interfejs **INaplata** koji sadrži:
   - property koji vraća valutu (koristiti troslovni kod valute, npr. RSD, EUR, USD, itd.),
   - property koji vraća cenu za naplatu,
   - proprety koji vraća limit – najvišu dozvoljenu cenu za naplatu.

4. Klasu **Korpa** koja implementira interfejs **INaplata** i služi za čuvanje proizvoda koji se kupuju.

   Pored atributa koji se koriste za implementaciju interfejsa **INaplata** klasa **Korpa** sadrži i :
   - niz (ili generičku listu proizvoda u korpi).

   a javne metode su sledeće:
   - metoda koja u tekstualni fajl na zadatoj putanji upisuje vrednosti svih privatnih atributa klase (uključujući i sve proizvode u korpi),
   - metoda koja iz tekstualnog fajla na zadatoj putanji čita vrednosti svih privatnih atributa klase (uključujući i sve proizvode u korpi), **ako ukupna cena proizvoda u korpi pređe limit baca se izuzetak sa odgovarajućom porukom**,
   - metoda koja iz korpe izbacuje sve proizvode zadatog statusa.

U funkciji **Main** učitati korpu proizvoda iz zadatog tekstualnog fajla („ulaz.txt“), izbaciti proizvode sa statusom **Naručen** pa zatim sačuvati sadržaj korpe u drugom tekstualnom fajlu („izlaz.txt“). Ukoliko u **Main** metodi dođe do izuzetka na stadardnom izlazu se ispisuje poruka izuzetka i prekida se dalje izvršenje programa.

Zadatak 5

Napomena: Klase implementirati tako da atributi imaju najniži mogući nivo pristupa (private ili protected).

Na programskom jeziku C# kreirati:

1. Apstraktnu klasu **Utakmica** za predstavljanje utakmice u proizvoljnom sportu čiji su atributi:
   - naziv domaće ekipe,
   - naziv gostujuće ekipe.

   U javnom delu klase definisati:
   - konstruktor koji inicijalizuje sve privatne atribute,
   - apstraktni property koji vraća ishod utakmice (može biti pobeda domaćina, pobeda gostiju ili nerešeno, definisati posebnu enumeraciju za predstavljanje ishoda),
   - apstraktni property koji vraća ukupan broj isključenih igrača iz obe ekipe,
   - virtualnu metodu koja u zadati tekstualni tok podataka upisuje vrednosti svih privatnih atributa klase,
   - virtualnu metodu koja iz zadatog tekstualnog toka podataka čita vrednosti svih privatnih atributa klase.

2. Klasu **FudbalskaUtakmica** izvedenu iz klase **Utakmica**.

   Privatni atributi klase **FudbalskaUtakmica** su:
   - broj golova domaće ekipe,
   - broj golova gostujuće ekipe,
   - broj isključenih igrača domaće ekipe,
   - broj isključenih igrača gostujuće ekipe.

   a javne metode su sledeće:
   - konstruktor koji postavlja sve atribute ove klase i osnovne klase **Utakmica**,
   - metoda koja u zadati tekstualni tok podataka upisuje vrednosti svih privatnih atributa ove klase i osnovne klase **Utakmica**,
   - metoda koja iz zadatog tekstualnog toka podataka čita vrednosti svih privatnih atributa ove klase i osnovne klase **Utakmica** (**ako je neki od atributa negativan broj baca se izuzetak sa odgovarajućom porukom**).

3. Interfejs **IFairPlay** koji sadrži:
   - proprety koji vraća limit – najveći dozvoljeni broj isključenja po utakmici (ukupan broj, računajući obe ekipe).

4. Klasu **Turnir** koja implementira interfejs **IFairPlay** i služi za čuvanje podataka o utakmicama na turniru.

   Pored privatnih atributa koji se koriste za implementaciju interfejsa **IFairPlay** klasa **Turnir** sadrži i :
   - niz (ili generičku listu utakmica na turniru).

   a javne metode su sledeće:
   - metoda koja u tekstualni fajl na zadatoj putanji upisuje vrednosti svih privatnih atributa klase (uključujući i sve utakmice na turniru),
   - metoda koja iz tekstualnog fajla na zadatoj putanji čita vrednosti svih privatnih atributa klase (uključujući i sve utakmice na turniru),
   - metoda koja sa turnira izbacuje sve utakmice gde je ukupan broj isključenja veći ili jednak zadatom limitu.

U funkciji **Main** učitati podatke o turniru iz zadatog tekstualnog fajla („ulaz.txt“), izbaciti sve utakmice gde je ukupan broj isključenja veći ili jednak zadatom limitu pa zatim sačuvati podatke o turniru u drugom tekstualnom fajlu („izlaz.txt“). Ukoliko u **Main** metodi dođe do izuzetka na stadardnom izlazu se ispisuje poruka izuzetka i prekida se dalje izvršenje programa.