Termin 1 (13:30)

NAPOMENE: Atributi klasa treba da budu **privatni** ili **zaštićeni**. Ukoliko je neophodan javni pristup nekom od atributa, iskoristiti javna svojstva (properties).
Dozvoljeno je kreiranje dodatnih članova klasa (metode, konstruktori, svojstva,...) koji nisu navedeni u tekstu zadatka u odgovarajućim klasama, ukoliko su neophodni za implementaciju.
Prilikom rada sa fajlovima obezbediti zatvaranje tokova na odgovarajućim mestima u kodu.

Na programskom jeziku C# implementirati sistem za upravljanje botaničkom baštom. Kreirati apstraktnu klasu **Biljka** koja sadrži atribute: naziv biljke, da li je jednogodišnja ili višegodišnja i količinu vode koja joj je potrebna na dnevnom nivou. Ova klasa sadrži i sledeće metode:

- apstraktnu metodu *cenaOdržavanja* koja vraća dnevni trošak za održavanje te biljke;
- virtualnu metodu koja u zadati binarni tok podataka upisuje vrednosti svih atributa klase,
- virtualnu metodu koja iz zadatog binarnog toka podataka čita vrednosti svih atributa klase.

Iz klase **Biljka** izvesti klase **SaksijskaBiljka** i **StablastaBiljka**.
SaksijskaBiljka pamti još potrebnu količinu đubriva u kilogramima, cenu đubriva po kilogramu. Cena održavanja saksijske biljke se računa po formuli: *količinaVode * 10 + količinaĐubriva * cenaĐubriva*. Ova cena se povećava za 10% ukoliko se radi o višegodišnjoj biljci.
StablastaBiljka pamti još i datum sledećeg presađivanja. Cena održavanja se izračunava po formuli: *količinaVode * 100*. Ukoliko je sledeći datum presađivanja jednak današnjem, cena se povećava za 2.000.
Kreirati klasu **BotaničkaBašta** koja pamti naziv bašte, broj posetilaca za dan, cenu ulaznice i listu biljaka. Ova klasa ima sledeće metode:

- metodu koja dodaje postojeću biljku u baštu;
- metodu *izbaciVišakBiljaka* koja treba da uklopi da ukupni dnevni trošak botaničke bašte za biljke ne pređe dnevnu zaradu od karata. Metoda radi tako što biljke **koje su višegodišnje** i čija je cena održavanja najveća izbacuje iz botaničke bašte sve dok cena ne postane manja ili jednaka zaradi od karata;
- metodu koja sortira biljke u botaničkoj bašti u opadajućem redosledu po količini vode koju troše;

- metodu *proveriRaspored* koja proverava da li u bašti ima i saksijskih i stablastih biljaka, a ukoliko neka vrsta biljaka nedostaje izbacuje izuzetak **NedovoljnaRaznovrsnost**;
- metodu koja u binarni fajl na zadatoj putanji upisuje vrednosti svih privatnih atributa klase (uključujući i sve biljke);
- metodu koja iz binarnog fajla na zadatoj putanji čita vrednosti svih privatnih atributa klase (uključujući i sve biljke).

U glavnom programu kreirati botaničku baštu. Dodati joj 3 stablaste i 3 saksijske biljke. Upisati sve podatke o bašti u binarni fajl. Zatim kreirati novi objekat **BotanickaBasta** i u taj objekat učitati podatke iz binarnog fajla. Na kraju iz novog objekta izbaciti višak biljaka, sortirati preostale po količini vode i proveriti raspored.

Termin 2 (15:30)

NAPOMENE: Atributi klasa treba da budu **privatni** ili **zaštićeni**. Ukoliko je neophodan javni pristup nekom od atributa, iskoristiti javna svojstva (properties).
Dozvoljeno je kreiranje dodatnih članova klasa (metode, konstruktori, svojstva,...) koji nisu navedeni u tekstu zadatka u odgovarajućim klasama, ukoliko su neophodni za implementaciju.
Prilikom rada sa fajlovima obezbediti zatvaranje tokova na odgovarajućim mestima u kodu.

Na programskom jeziku C# implementirati sistem za upravljanje taksi službom. Kreirati apstraktnu klasu **Auto** koja sadrži atribute: naziv automobila i godinu proizvodnje. Ova klasa sadrži i sledeće metode:

- apstraktnu metodu *cenaKilometra* koja vraća trošak tog auta za pređeni kilometar;
- virtualnu metodu koja u zadati tekstualni tok podataka upisuje vrednosti svih atributa klase;
- virtualnu metodu koja iz zadatog tekstualnog toka podataka čita vrednosti svih atributa klase.

Iz klase **Auto** izvesti klase **EAuto** za predstavljanje električnog automobila i **AutoSUS** za predstavljanje automobila sa unutrašnjim sagorevanjem.
EAuto pamti još potrebnu količinu električne energije u kWh za prelaženje jednog kilometra, cenu električne energije po kWh i datum kad ističe rok trajanja baterije. Cena pređenog kilometra električnog automobila se računa po formuli: *potrebnoKwhZaKm* * *cenaKwh*. Ova cena se povećava za 5% za svaku godinu starosti automobila.
AutoSUS pamti još potrošnju u litrima goriva po pređenom kilometru i cenu goriva po litru. Cena pređenog kilometra se izračunava po formuli: *potrebnoLitaraZaKm* * *cenaLitra*. Ova cena se povećava za 10% za svaku godinu starosti automobila.
Kreirati klasu **TaksiSluzba** koja pamti naziv službe, ukupan broj pređenih kilometara, ukupno naplaćenu cenu usluga od korisnika i listu vozila. Ova klasa sadrži i sledeće metode:

- metodu koja dodaje postojeći automobil u taksi službu;
- metodu *izbaciPreskupaVozila* koja treba da izbaci vozila koja su preskupa po ceni kilometra. Metoda radi tako što izbacuje vozila koja potroše po kilometru više novca nego što je firma prosečno naplatila od korisnika po kilometru;
- metodu koja sortira automobile u opadajućem redosledu po godini proizvodnje;
- metodu *proveriBaterije* koja za svaki električni automobil proverava da li rok trajanja baterije istekao i ukoliko jeste baca izuzetak **PotencijalnaEksplozija**;

- metodu koja u tekstualni fajl na zadatoj putanji upisuje vrednosti svih privatnih atributa klase (uključujući i sve automobile);
- metodu koja iz tekstualnog fajla na zadatoj putanji čita vrednosti svih privatnih atributa klase (uključujući i sve automobile).

U glavnom programu kreirati taksi službu. Dodati joj 3 električna i 3 SUS automobila. Upisati sve podatke o taksi službi u tekstualni fajl. Zatim kreirati novi objekat **TaksiSluzba** i u taj objekat učitati podatke iz tekstualnog fajla. Na kraju iz novog objekta izbaciti preskupe automobile, sortirati preostale po godini proizvodnje i proveriti baterije.

Termin 3 (17:30)

NAPOMENE: Atributi klasa treba da budu **privatni** ili **zaštićeni**. Ukoliko je neophodan javni pristup nekom od atributa, iskoristiti javna svojstva (properties).
Dozvoljeno je kreiranje dodatnih članova klasa (metode, konstruktori, svojstva,...) koji nisu navedeni u tekstu zadatka u odgovarajućim klasama, ukoliko su neophodni za implementaciju.
Prilikom rada sa fajlovima obezbediti zatvaranje tokova na odgovarajućim mestima u kodu.

Na programskom jeziku C# implementirati sistem za rad sa restoranom. Kreirati apstraktnu klasu **Stavka** koja predstavlja stavku iz menija i sadrži atribute: naziv, datum isteka roka trajanja i da li je stavka veganska. Ova klasa sadrži i:

- apstraktno svojstvo koje vraća cenu stavke;
- virtualnu metodu koja u zadati binarni tok podataka upisuje vrednosti svih atributa klase;
- virtualnu metodu koja iz zadatog binarnog toka podataka čita vrednosti svih atributa klase.

Iz klase **Stavka** izvesti klase **Jelo** i **Piće**.
Klasa **Jelo** čuva još i nabavnu cenu, kvalitet (definisati enumeraciju sa vrednostima **Prihvatljiv** (1), **Dobar** (2), **Odlican** (3)). Cena jela se izračunava kao proizvod nabavne cene i kvaliteta. Ukoliko je istek roka trajanja uskoro (3 dana ili manje) ova cena se smanjuje za 20%.
Klasa **Piće** čuva još i količinu u litrima, cenu po litru i da li je piće domaće ili uvozno. Cena se izračunava po sledećoj formuli: *cenaPoLitru * količina*. Ukoliko je piće uvozno ova cena se dodatno uvećava za 30%.
Kreirati klasu **Restoran** koja pamti naziv restorana i listu stavki na meniju. Ova klasa sadrži i sledeće metode:

- metodu koja dodaje postojeću stavku u restoran;
- metodu koja sortira stavke u restoranu po ceni rastuće;
- metodu koja iz restorana izbacuje stavke kojima je rok trajanja istekao;
- metodu koja proverava da li u restoranu ima veganskih stavki, a ukoliko ih nema baca izuzetak **VeganUnfriendly**;
- metodu koja u binarni fajl na zadatoj putanji upisuje vrednosti svih privatnih atributa klase (uključujući i sve stavke);
- metodu koja iz binarnog fajla na zadatoj putanji čita vrednosti svih privatnih atributa klase (uključujući i sve stavke).

U glavnom programu kreirati restoran i dodati mu 4 jela i 4 pića. Upisati sve podatke o restoranu u binarni fajl. Zatim kreirati novi objekat **Restoran** i u taj objekat učitati podatke iz binarnog fajla. Na kraju iz novog objekta izbaciti stavke kojima je istekao

rok trajanja, sortirati preostale stavke po ceni rastuće i proveriti da li ima veganskih stavki.