Visoka škola za menadžment u turizmu i informatici u Virovitici

**Milijunaš**

Kolegij: Osnove programiranja

Projektni Zadatak

Mentori: Ivan Heđi, dipl.ing. Student: Mario Šomođi

Mario Pecimotika, mag.educ.phys.et inf.

Marko Hajba, mag.math.

Sadržaj

[Opis zadatka 2](#_Toc31614837)

[Strukture 3](#_Toc31614838)

[Odgovori 3](#_Toc31614839)

[Rezultat 3](#_Toc31614840)

[Pitanje 3](#_Toc31614841)

[Funkcije 3](#_Toc31614842)

[static void Main(string[] args) 3](#_Toc31614843)

[static void OcistiLogDatoteku() 3](#_Toc31614844)

[static void LogajTrenutnuAkciju(string akcija) 3](#_Toc31614845)

[static void OcistiLinijuUKonzoli() 3](#_Toc31614846)

[static void OcistiMemoriju() 4](#_Toc31614847)

[static void OcistiOdredeniBrojLinijaUKonzoli(int brojLinija) 4](#_Toc31614848)

[static void DohvatiIzbornik() 4](#_Toc31614849)

[static void PonovniPrikazIzbornikaIliIzlaz() 4](#_Toc31614850)

[static void IspisiIzbornik() 5](#_Toc31614851)

[static int OdabirOpcijeSaIzbornika() 5](#_Toc31614852)

[static string UnosTeProvjeraDaUnosNijePrazan(string polje) 5](#_Toc31614853)

[static void IspisUnosa(string polje) 5](#_Toc31614854)

[static void IspisiError(string polje, int brojLinija) 5](#_Toc31614855)

[static void PauzirajProgram(int trajanje) 6](#_Toc31614856)

[static void DohvatiPitanja() 6](#_Toc31614857)

[static int DohvatiUkupniBrojPitanja() 6](#_Toc31614858)

[static Odgovori RandomizirajIIspisiOdgovore(Pitanje pitanje) 6](#_Toc31614859)

[static void UnosOdg(Pitanje pitanje, Odgovori odgovori) 7](#_Toc31614860)

[static void IspisPitanja() 7](#_Toc31614861)

[static int ProvjeriPredeniPrag() 8](#_Toc31614862)

[static void UpisRezultataUXML(string ime, string prezime, int dobitak) 8](#_Toc31614863)

[static void DohvatiRezultate() 9](#_Toc31614864)

[static void IspisiRezultate() 9](#_Toc31614865)

# Opis zadatka

Napisati program koji simulira igru “Milijunaš” . Igra se sastoji od 15 pitanja koja se učitavaju iz baze pitanja zapisanih u datoteci pitanja.xml. Svako pitanje treba imati čvor sa nazivom pitanja unutar kojega se treba nalaziti novi čvor sa 4 moguća odgovora. Pragovi se nalaze na petom i desetom pitanju te osiguravaju natjecatelju minimalni iznos koji mogu osvojiti. Shema pitanja je sljedeća:

|  |  |  |
| --- | --- | --- |
| Redni broj pitanja | Novčana vrijednost pitanja | Težina pitanja |
| 1. | 100kn | 1 |
| 2. | 200kn | 1 |
| 3. | 300kn | 1 |
| 4. | 500kn | 1 |
| 5. | 1.000kn | 2 |
| 6. | 2.000kn | 2 |
| 7. | 4.000kn | 3 |
| 8. | 8.000kn | 3 |
| 9. | 16.000kn | 4 |
| 10. | 32.000kn | 4 |
| 11. | 64.000kn | 5 |
| 12. | 125.000kn | 5 |
| 13. | 250.000kn | 6 |
| 14. | 500.000kn | 6 |
| 15. | 1 MILIJUN | 7 |

Svako pitanje zapisano u datoteci pitanja.xml treba imati određenu težinu (npr. Atribut tezina) te se postavlja na odgovarajuću poziciju u igri.

Nakon što se pokrene program prikazuje se početni izbornik.

**1-Nova Igra**

Kandidat upisuje svoje ime i prezime nakon čega mu se sljedno prikazuje pitanje po pitanje (u slučaju točnih odgovora). Ispod pitanja se treba prikazati forma za unos odgovora, te se nakon pritiska tipke ENTER ispisuje ishod: TOČNO ili NETOČNO. U slučaju netočnog odgovara kandidat osvaja iznos minimalnog prijeđenog praga. Nakon završene igre potrebno je zapisati u datoteku rezultati\_kandidata.xml rezultat kandidata sa atributima ime, prezime, datum\_igranja te osvojeni\_iznos.

**2-Rezultati kandidata**

Prikaz svih kandidata koji su sudjelovali u igri, sortirani od kandidata koji su osvojili najveći iznos.

**3-Izlaz**

Odaberi sve opcije, program se završava.

**Popis datoteka:**

* pitanja.xml
* rezultati\_kandidata.xml
* logs.log

# Strukture

## Odgovori

Ima 4 vrijednosti koje su odgA, odgB, odgC, odgD koje sadrže sadržaj odgovora. Potrebna je kako bi pri randomiziranju odgovora spremili njihove sadržaje, kod ispisa pitanja tako da uvijek znamo pod kojim slovom je koji odgovor.

## Rezultat

Ima 4 vrijednosti koje su prvoImeIgraca, prezimeIgraca, datumIgranja, osvojeniIznos koje sadrže te podatke. Koristim ju za spremanje svih podataka iz xml-a rezultati\_kandidata tako što napravim listu tipa Rezultat.

## Pitanje

Ima 8 vrijednosti koje su idPitanja, tezinaPitanja, upit, odgA, odgB, odgC, odgD, tocanOdg koje sadrže te podatke. Koristim ju za spremanje svih podataka iz xml-a pitanja tako što napravim listu tipa Pitanje.

# Funkcije

## static void Main(string[] args)

Parametri : string[] args

Nove inicijalizirane varijable : nema.

Funkcija koja ce biti prva izvedena bez obzira na redosljed ostalih funkcija. Poziva druge funkcije koje pokreću daljnji rad programa.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void OcistiLogDatoteku()

Parametri : nema.

Nove inicijalizirane varijable : nema.

Otvaram StreamWriter sa nazivom log koji pokazuje na logs.log datoteku te šaljem parametar false StreamWriter-u što znači da neće dodati na datoteku nego će ju očistiti pri pozivu funkcije, te na kraju zatvaram log StreamWriter.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void LogajTrenutnuAkciju(string akcija)

Parametri : varijabla akcija tipa string.

Nove inicijalizirane varijable :

* Varijabla now tipa DateTime u koju spremam vrijeme kada je pozvana funkcija.

Otvaram StreamWriter sa nazivom log koji pokazuje na logs.log datoteku te šaljem parametar true StreamWriter-u što znači da ce dodavati na tu datoteku svaki put kad je funkcija pozvana tj. nece brisati prijašnje upisani sadržaj. Onda upisujem u logs.log datoteku vrijeme poziva funkcije te pokraj akciju koju sam poslao funkciji i na kraju sačuvam i zatvaram log StreamWriter.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void OcistiLinijuUKonzoli()

Parametri : nema.

Nove inicijalizirane varijable :

* Varijabla currentLineCursor tipa int koja ce imati vrijednost na kojoj se kursor trenutno nalazi u konzoli.

Namješta kursor na nulti stupac I trenutni red gdje je kursor bio, onda piše prazan string po cijeloj dužini konzole i na kraju vraća kursor na poziciju na kojoj je bio prije upisa praznog stringa.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void OcistiMemoriju()

Parametri : nema

Nove inicijalizirane varijable : nema.

Očistimo sve liste koje zauzimaju memoriju. Funkcija se poziva uvjek prije terminiranja programa.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void OcistiOdredeniBrojLinijaUKonzoli(int brojLinija)

Parametri : varijabla brojLinija tipa int

Nove inicijalizirane varijable :

* Varijabla brojac tipa int na početku je 0 služi da znamo koliko puta smo prošli kroz while petlju.

Imamo while petlju koja će se izvršavati sve dok je brojac manji od brojLinija, namještamo kursor na jednu liniju iznad linije na kojoj je bio te pozivamo funkciju OcistiLinijuUKonzoli koja će ju očistiti te inkrementiramo brojac. To ponavljamo sve dok brojac nepostane jednak brojuLinija ili veći.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void DohvatiIzbornik()

Parametri : nema.

Nove inicijalizirane varijable :

* Varijabla izlaz tipa bool koja će u početku biti false. Koristi se za izlaz iz do-while petlje kada korisnik odabere izlaz iz programa tu ćemo varijablu postaviti na true te će se program terminirati.
* Varijabla prvoPokretanjeIzbornika u početku će biti true.Čim korisnik pokrene izbornik drugi put ova varijabla će se promjeniti u false. Koristi se da znamo kada je korisnik pokrenuo izbornik više od jedanput kako bi mogli resetirati sve varijable koje su potrebne za početak nove igre.
* Varijabla odabir tipa int. U ovu varijablu spremamo korisnikom odabir sa izbornika.
* Varijabla prvoImeIgraca tipa string, sprema ime koje je unjeo korisnik.
* Varijabla prezimeIgraca tipa string, sprema prezime koje je unjeo korisnik.
* Varijabla novcaniDobitak tipa int, sprema dobiveni iznos koji je korisnik dostigao nakon završene igre.

Imam do-while petlju koja će otvarati izbornik sve dok je varijabla izlaz jednaka false. Ovo je funkcija koja daje korisniku odabir jednoga od ponuđenih izbora na izborniku te onda pomoću switch-casea izvršava taj izbor tj. poziva sve druge funckije koje su potrebne za izvršavanje tog izbora.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void PonovniPrikazIzbornikaIliIzlaz()

Parametri : nema.

Nove inicijalizirane varijable :

* Varijabla krajUnosa tipa bool koja će na početku biti false, koristi se za izlaz iz while petlje.
* Varijabla sadrzajUnosa tipa ConsoleKeyInfo koja će sadržavati naš unos.

Imam while petlju koja će se izvršavati sve dok krajUnosa nije true u varijablu sadrzajUnosa spremamo unos korisnika i onda imamo if else u kojem prvo provjeravamo dali je korisnik pritisnuo ENTER tipku i ako je postaviti ćemo krajUnosa na true kako bi izašli iz petlje te će program nastaviti i ponovno prikazati izbornik pomoću do-while petlje koju imamo u funkciji DohvatiIzbornik.Nakon provjere za ENTER tipku ako korisnik nije pritisnuo enter tipku ide provjera sa ESC tipku ako je pritisnuo ESC tipku onda će se program terminitati, inače ako korisnik ne pristisne niti jednu od te dvije tipke dobiti će error i ponavlja se unos dok neizabere jednu od te dvije tipke.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void IspisiIzbornik()

Parametri : nema.

Nove inicijalizirane varijable : nema.

Jednostavna funkcija koja će samo ispisati izbornik u konzolu.

Funkcija je tipa void sto znači da ne vraća nikakvu informaciju.

## static int OdabirOpcijeSaIzbornika()

Parametri : nema.

Nove inicijalizirane varijable :

* Varijabla krajUnosa tipa bool na početku je false koju koristimo za izlaz iz while petlje.
* Varijabla sadrzajUnosa tipa ConsoleKeyInfo koja sadrži ono sto je korisnik unjeo.
* Varijabla unos tipa char koja ce sadržavati ASCII vrijednost onoga sto je korisnik unjeo.

Imam while petlju koja će se izvršavati sve dok krajUnosa nije true. Čitamo korisnikov unos i spremamo ga u sadrzajUnosa te onda uzimamo njegovu ASCII vrijednost i spremamo ju u unos kako bi u if-u mogli provjeriti dali je korisnik unjeo broj ili je unjeo nešto što nije broj, ako je unjeo broj funkcija će vratiti vrijednost tog broja u tipu integera tako što ćemo oduzeti ASCII vrijednost broja ‘0’ i dobiti integera. Inače ako je korisnik pritisnuo bilo koju tipku osim broja ispisujemo error i dajemo mu ponovni odabir.

## static string UnosTeProvjeraDaUnosNijePrazan(string polje)

Parametri : varijabla polje tipa string koja sadrži ime polja koje trenutno provjeravamo.

Nove inicijalizirane varijable :

* Varijabla sadrzajUnosa tipa string koja je u početku jednaka praznom stringu “”.

Pomocu while petlje korisniku dajemo unos informacije sve dok nešto neupiše ako je ostavio unos praznim dobiti će error i unos će mu se ponovno prikazati. Kada korisnik upiše informaciju koja je tražena izlazimo iz while petlje te funkcija vraća sadrzajUnosa.

## static void IspisUnosa(string polje)

Parametri : varijabla polje tipa string koja sadrži ime polja koje trenutno provjeravamo.

Nove inicijalizirane varijable : nema.

Jednostavna funkcija koja će pomoću varijable polje samo ispisati korisniku šta da unese kada se koristi funckija UnosTeProvjeraDaUnosNijePrazan.

Funckija je tipa void sto znači da ne vraća nikakvu informaciju.

## static void IspisiError(string polje, int brojLinija)

Parametri : varijabla polje tipa string koja sadrži informaciju o polju koje trenutno provjeravamo, varijabla brojLinija tipa int koja sadrži broj koliko linija koda se mora izbrisati kako bi nam konzola izgledala prikadno.

Nove inicijalizirane varijable : nema.

Funkcija će ispisati error na konzolu za određenu informaciju koju trenutno unosimo a to ćemo znati preko parametra polje.

Funkcija je tipa void što znači da ne vraća nikakvu informaciju.

## static void PauzirajProgram(int trajanje)

Parametri : varijabla trajanje tipa int

Nove inicijalizirane varijable : nema.

Funkcija koja će kod poziva pauzirati program na određeni broj milisekundi koje šaljemo preko parametra trajanje kod poziva funkcije.

Funkcija je tipa void što znači da ne vraća nikakvu informaciju.

## static void DohvatiPitanja()

Parametri : nema.

Nove inicijalizirane varijable :

* Varijabla sadrzaj\_xml tipa string koja će sadržavati sve sto je u xml-u pitanja.
* Varijabla xml\_datoteka tipa XmlDocument
* Varijabla atributi tipa XmlNodeList

Prvo ocistimo globalnu listu koja se zove listaPitanja tako da slučajno nebih duplicirali pitanja ako se igra bude dva puta igrala. Otvaramo StreamReader oSr koji pokazuje na pitanja.xml te pomoču njega pročitamo cijeli taj xml file i spremimo ga u sadrzaj\_xml. Pomoću kreirane varijable xml\_datoteka učitamo xml koji se nalazi u varijabli sadrzaj\_xml te onda spremimo sve attribute svakog pitanja u varijablu atributi. Sada možemo pomoću foreach petlje prolaziti kroz svako pitanje i njegove atribute. U foreach petlji dodajemo novi element u listu listaPitanja koja je tipa Pitanje(struktura prije objašnjena) element je isto tipa Pitanje I sadrži sve atribute toga trenutnog pitanja i to napravimo za svako pitanje iz xml-a i sada sva pitanja imamo u listi i možemo pristupiti svim atributima svakog pitanja pomoću strukture.

Funkcija je tipa void što znači da ne vraća nikakvu informaciju.

## static int DohvatiUkupniBrojPitanja()

Parametri : nema.

Nove inicijalizirane varijable :

* Varijabla ukupnoPitanja tipa int koja je na početku 0.

Pomocu foreach petlje prolazimo kroz sva pitanja u listi listaPitanja te za svako pitanje inkrementiramo ukupnoPitanja za jedan i tako izbrojimo koliko ukupno imamo pitanja. Na kraju kada izbrojimo koliko ima pitanja funkcija vraća ukupnoPitanja uvečana za jos jedan zato što to koristimo za generaciju random broja do maksimalnog broja pitanja a random funkcija nam daje broj jedan manje od max i zato ga moramo povečati za jedan.

## static Odgovori RandomizirajIIspisiOdgovore(Pitanje pitanje)

Parametri : varijabla pitanje tipa Pitanje(struktura)

Nove inicijalizirane varijable i liste:

* Varijabla sadrzajOdgovoraA tipa string.
* Varijabla sadrzajOdgovoraB tipa string.
* Varijabla sadrzajOdgovoraC tipa string.
* Varijabla sadrzajOdgovoraD tipa string.
* Varijabla abcd tipa string koja je na početku jednaka “a”. Varijabla koju koristim za ispis odgovora.
* Varijabla sadrzajOdgovora tipa string.
* Varijabla brojOdg tipa int koja je na početku jednaka 1.
* Nova lista iskoristeniOdgovori tipa int.
* Varijabla rnd tipa Random.
* Varijabla odg tipa int.
* Varijabla sadrzajiSvihOdgovora tipa Odgovori.

Pomoću while petlje koja će se izvršavati sve dok je brojOdg manji od 5, kreiramo novi random broj od 1 do 5 i spremamo ga u varijablu odg onda pomoću switch-casea u varijablu sadrzajOdgovora spremimo odgovarajući odgovor toga pitanja recimo da je random broj bio 3 spremili bi odgC pitanja u sadrzajOdgovora i onda imamo jedan if koji provjerava dali se odabrani odgovor već ispisao tj. nalazi li se u listi iskoristeniOdgovori ako se nalazi onda se generira novi broj I trazi se odgovor koji se nenalazi u listi ako se nenalazi u listi onda ulazimo u if i ispisujemo prvi odgovor te ga dodajemo u listu da znamo da je iskorišten te onda imamo switch-case koji će po tome koji je odgovor po redu tj. varijabla brojOdg odrediti sljedeći ispis postaviti će abcd varijablu na “b” te će sadrzajOdgovoraA staviti da je jednak varijabli sadrzajOdgovora i nakon toga će povečati brojOdg i ponavljati proces dok neispiše 4 odgovora koji se neponavljaju. Kada ispiše 4 odgovora i spremi sadržaj svakoga u odgovarajuću varijablu onda ćemo u varijablu sadrzajiSvihOdgovora spremiti sva 4 odgovora. Funkcija će na kraju vratiti varijablu sadrzajiSvihOdgovora.

## static void UnosOdg(Pitanje pitanje, Odgovori odgovori)

Parametri : varijabla pitanje tipa Pitanje(struktura), varijabla odgovori tipa Odgovori(struktura)

Nove inicijalizirane varijable :

* Varijabla odgovor tipa string u koju ćemo spremati odgovor korisnika.
* Varijabla siguran tipa string u koji spremamo odluku korisnika o odustajanju.
* Varijabla konacanOdgovor tipa string u koju spremamo odluku korisnika o potvrdi svoga odgovora.

Imam while petlju koja će se izvršavati sve dok korisnik neunese a, b, c, d ili da za odustajanje ako korisnik unese bilo što drugo izbaciti će mu error i ponuditi ponovnu mogučnost unosa odgovora. Nakon sto korisnik unese jedan od odgovora provjeravamo koji je unjeo pomocu if-ova te onaj koji je unjeo stavljamo da je jednak sadrzaju toga odgovora koji imamo u varijabli pitanje a ako mu je odgovor bio da onda cemo ga jos jednom pitati dali je siguran da zeli odustati ako odgovori bilo sto osim da ili ne program ce izbaciti error i dati mu ponovni upit dali je siguran sve dok ne odluci ako izabere da onda cemo postaviti globalnu varijablu odustaje na true koju onda koristimo u funkciji IspisPitanja da zavrsimo igru, ako je odgovorio ne onda cemo samo nastaviti sa igrom I dati mu ponovni upis odgovora na pitanje koje je bilo prije odustajanja. Nakon toga imamo if koji ce se izvrsiti ako je varijabla odustaje jednaka false I ako odgovor nije da unutar if-a imamo upit za konacni odgovor koji je u while-u sve dok korisnik neodgovori sa da ili ne. Kada odgovori sa da imamo if koji ce provjeriti jeli njegov odgovor jednak tocnomOdg pitanja pomocu varijable pitanje ako je ispisati ce TOCNO! Te postaviti globalnu varijablu tocno na true I povecati globalnu varijablu tocniOdgovori za jedan, inace ako je netocan odg postaviti ce globalnu varijablu tocno na false sto cemo onda poslje u funkciji IspisPitanja koristiti za zavrsetak igre ako na konacni odgovor odgovori sa ne pozivamo funkciju UnosOdg opet i dajemo mu ponovni unos odg-a sve dok neodgovori ili neodustane.

## static void IspisPitanja()

Parametri : nema.

Nove inicijalizirane varijable i liste :

* Varijabla brojPitanja tipa int koja je na pocetku 1, s kojom cemo pratiti koliko smo pitanja ispisali na konzolu.
* Varijabla odBrojaPitanja tipa int koja je na pocetku 1, od ovog broja pitanja do doBrojaPitanja
* Varijabla doBrojaPitanja tipa int koja je na pocetku 4, do ovog broja pitanja
* Varijabla trenutnaTezina tipa string koja je na pocetku “1”, tezina ce biti ovoliko dok je brojPitanja 1-4.
* Nova lista ids tipa string koja ce sadrzavati sve id-e pitanja koja smo vec ispisali.
* Varijabla rnd tipa Random.
* Varijabla ukupnoPitanja tipa int.
* Varijabla id tipa string.

Prvo u varijablu ukupnoPitanja spremimo ukupni broj pitanja koji cemo dobiti pomocu funckcije DohvatiUkupniBrojPitanja. Te nakon toga imamo while petlju koja ce se izvrsavati sve dok je brojPitanja manji od 16 unutar te while petlje imamo foreach petlju koja prolazi kroz sva pitanja u listi listaPitanja. Sada imamo if pomocu kojega provjeravamo dali je trenutni broj pitanja koji ispisujemo veci ili jednak varijabli odBrojaPitanja I da je brojPitanja manji ili jednak varijabli doBrojaPitanja to znaci da smo u ovom slucaju trazimo prva 4 pitanja a prva 4 pitanja moraju imati tezinu jedan I zato imamo jos jednu stvar u if-u a to je da je tezina trenutnog pitanja koje smo pristupili pomocu foreach-a jednaka trenutnoTezini koja je “1”, ako je to sve istina onda ulazimo u if I generiramo random broj koji cemo spremiti u varijablu id onda provjeravamo dali je id trenutnog pitanja jednak random generiranom id-u ako je imamo jos jednu odluku gledamo dali lista ids sadrzi id toga pitanja znaci gledamo dali smo to pitanje vec ispisali ovo je potrebno tako da se pitanja nebi ponavljala ako lista nesadrzi taj id onda cemo ispisati upit toga pitanja te cemo pomocu funkcije RandomizirajIIspisiOdgovore(prijasnje objasnjena) onda pozivamo funkciju UnosOdg(Prijasnje objasnjena) pomocu koje provjeravamo dali je korisnik unjeo tocan ili netocan odgovor I dali je korisnik odustao ako je korisnik odusato onda cemo otici na kraj ove funkcije I prikaziti njegov dobitak ako ga je imao te dati mu izbor ponovnog prikaza izbornika ili izlaza iz programa ako nije odustao onda cemo dodati id toga pitanja u listu ids I inkrementirati brojPitanja da znamo da smo sad ana drugom pitanju, ako je korisnik netocno odgovorio desiti ce se ista stvar kao kad je odlucio odustati. Znaci kada varijabla brojPitanja dode do 5 pitanja onda cemo pomocu switch-casea promjeniti vrijednosti trenutneTezine na “2” odBrojaPitanja na 5 te doBrojaPitanja na 6 I tako stalno mjenjamo tezinu koja odredena pitanja na tom mjestu trebaju imati I samo pitanja sa tom tezinom ispsujemo sve dok korisnik neodustane, odgovori netocno ili dode do kraja igre I postane Milijunas.

Funkcija je tipa void sto znaci da ne vraca nikakvu informaciju.

## static int ProvjeriPredeniPrag()

Parametri : nema.

Nove inicijalizirane varijable :

* Varijabla novcaniDobitak tipa int.

Provjeravamo koliko je korisnik imao tocnih odgovora ako je imao izmedu 5 i 10 onda cemo postaviti varijablu novcaniDobitak na 1000 ako je imao izmedu 10 i 15 onda na 32000 a ako je imao 15 tocnih odgovora postaviti cemo ju na 1000000 inace ako nista od toga nije istina onda cemo ju postaviti na 0 i javiti korisniku da nije niti jedan prag presao. Funkcija ce na kraju vratiti varijablu novcaniDobitak.

## static void UpisRezultataUXML(string ime, string prezime, int dobitak)

Parametri : varijabla ime tipa string, varijabla prezime tipa string, varijabla dobitak tipa int.

Nove inicijalizirane varijable :

* Varijabla postoji tipa bool.
* Varijabla now tipa DateTime u koju cemo spremiti trenutno vrijeme.
* Varijabla doc tipa XDocument.

Prvo provjeravamo postoji li datoteka rezultati\_kandidata.xml u slucaju da je ovo prvi rezultat datoteka nece postojati imamo if u koji udemo ako je varijabla postoji false napraviti cemo datoteku te spremiti prvi rezultat u nju pomocu System.Xml.Linq biblioteke. Te na kraju spremiti datoteku. U slucaju da vec postoji samo cemo dodati novi element na postojecu datoteku kojic e biti trenutni rezultat sa atributima ime, prezime, datumIgranja te osvojeni iznos igraca, te isto spremiti na kraju.

Funkcija je tipa void sto znaci da ne vraca nikakvu informaciju.

## static void DohvatiRezultate()

Parametri : nema.

Nove inicijalizirane varijable :

* Varijabla sadrzaj\_xml tipa string koja ce sadrzavati sve sto je u xml-u rezultati\_kandidata.
* Varijabla xml\_datoteka tipa XmlDocument
* Varijabla atributi tipa XmlNodeList

Prvo ocistimo globalnu listu koja se zove rezultati tako da ako korisnik ponovno pokrene izbornik i ispisuje rezultate drugi put kako se nebih dva puta ispisali. Otvaramo StreamReader oSr koji pokazuje na rezultati\_kandidata.xml te pomocu njega procitamo cijeli taj xml file i spremimo ga u sadrzaj\_xml. Pomocu kreirane varijable xml\_datoteka ucitamo xml koji se nalazi u varijabli sadrzaj\_xml te onda spremimo sve attribute svakog rezultata u varijablu atributi. Sada mozemo pomocu foreach petlje prolaziti kroz svaki rezultat I njegove atribute. U foreach petlji dodajemo novi element u listu rezultati koja je tipa Rezultat(struktura prije objasnjena) element je isto tipa Rezultat I sadrzi sve atribute toga trenutnog pitanja I to napravimo za svaki rezultat iz xml-a i sada sve rezultate imamo u listi i mozemo pristupiti svi atributima svakog rezultata pomocu strukture.

Funkcija je tipa void sto znaci da ne vraca nikakvu informaciju.

## static void IspisiRezultate()

Parametri : nema

Nove inicijalizirane varijable :

* Varijabla redniBroj tipa int na pocetku 0.
* Varijabla table tipa ConsoleTable

Sortiramo listu rezultati po osvojenom iznosu tako da ide on veceg do manjeg pomocu System.Linq biblioteke. Inicijaliziramo novu tablicu pomocu ConsoleTables biblioteke koja ce imati 5 stupaca koji su Redni broj, Ime, Prezime, Datum pristupa, Osvojeni iznos. Onda imam foreach petlju koja prolazi kroz sve rezultate u listi rezultati I dodaje vrijednosti svaku vrijednost u svoj stupac i tako za svaki rezultat dok neprode sve. Kada prode sve ispise tablicu.

Funkcija je tipa void sto znaci da ne vraca nikakvu informaciju.