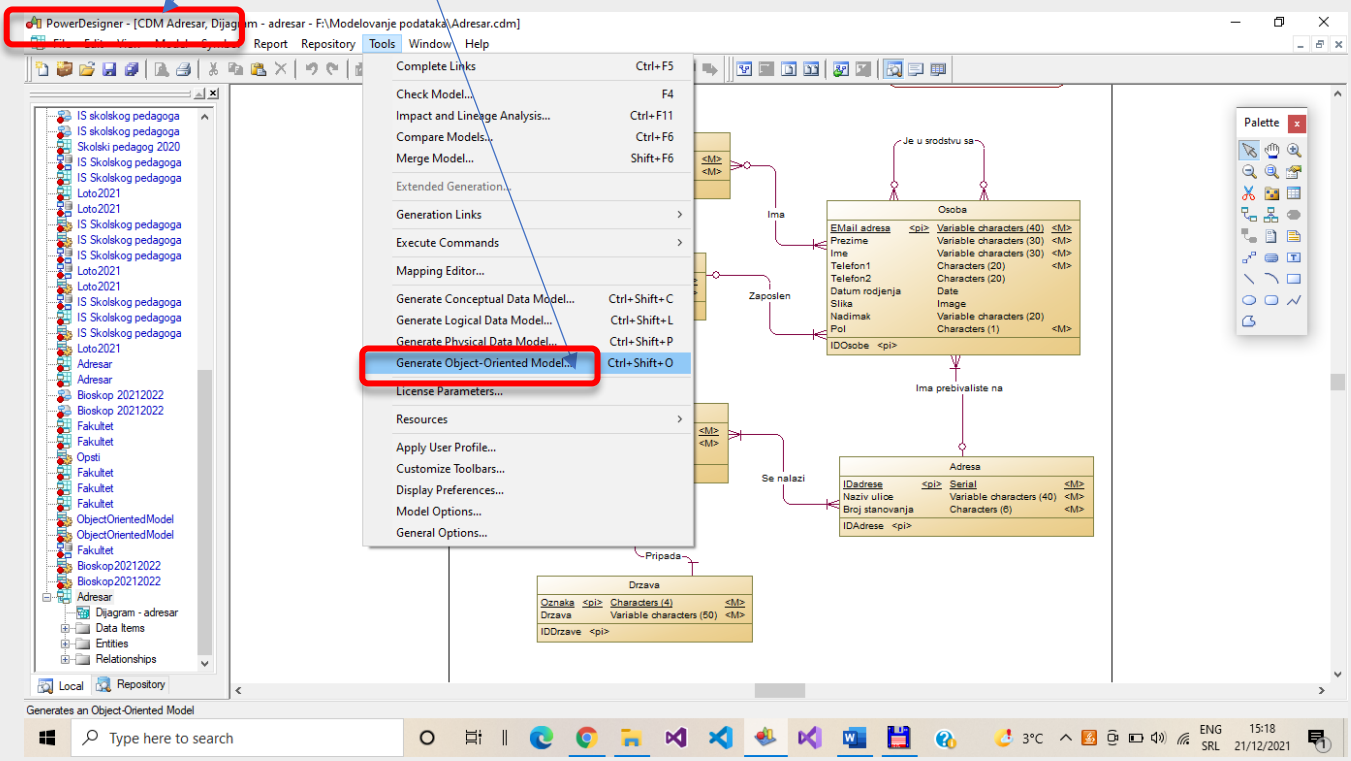
TUTORIJAL ZA RAD SA SYBASE POWER DESIGNER CASE ALATOM - OOM

MODELOVANJE OBJEKATA (Class Diagram objektnog modela, na osnovu CDM modela podataka)

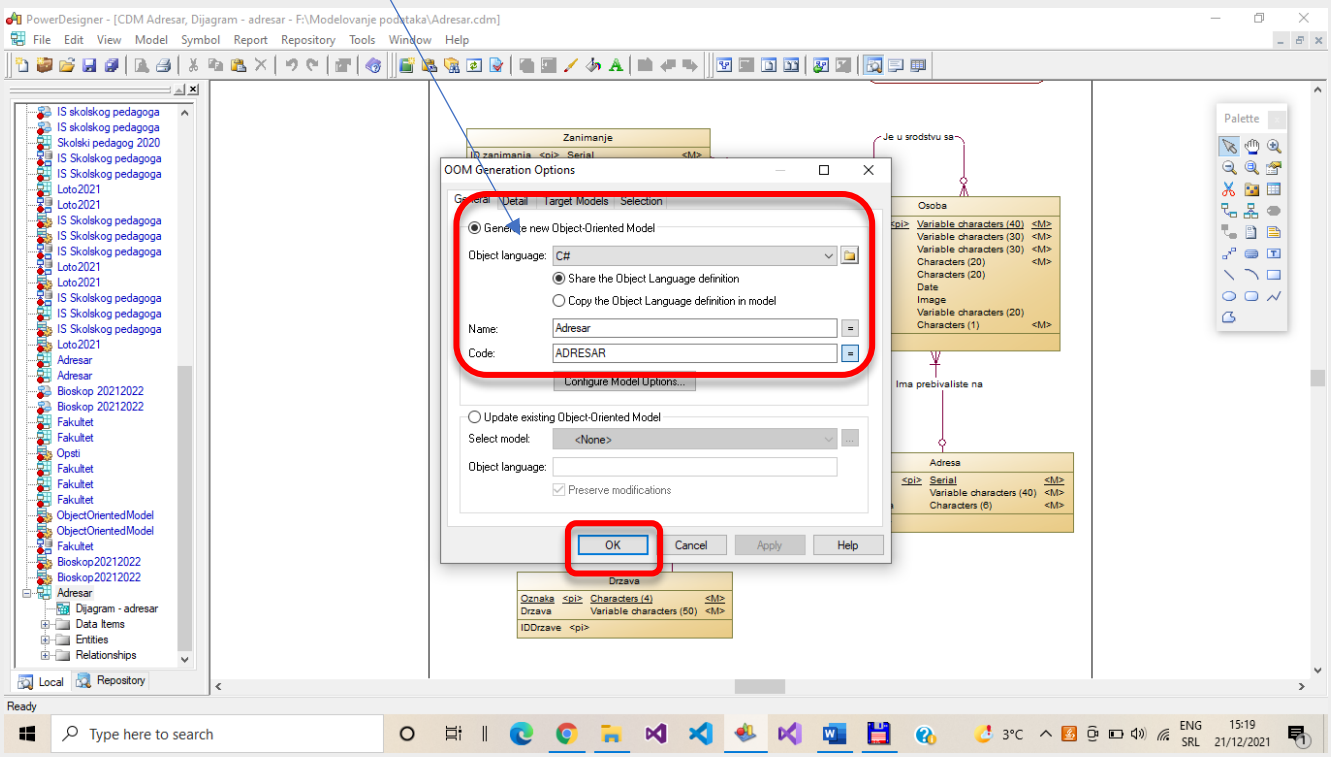
Apstraktna predstava objekta se zove klasa (Class) i svaki objekat mora pripadati nekoj klasi. Objekat je instanca klase. Klasa se opisuje svojim imenom/nazivom, karakteristikama u vidu atributa (Attributes) i zadacima/operacijama koje se mogu izvršiti nad klasom, tj. metodama (Methods). Stanje objekta se definiše kao skup vrednosti svih njegovih atributa i vezama sa ostalim objektima u jednom trenutku vremena. Metode implementiraju ponašanje objekta. Mogu biti javni i privatni/tajni.

Metode su procedure i funkcije u programskom jeziku. Atributi se mogu formirati na osnovu relacionog modela podataka i za sve atribute se mora definisati neki tip podatka. U savremenim CASE alatima je mogude direktno mapirati konceptualni ili fizički model podataka u objektno orijentisani model. Metode se definišu na osnovu aktivnosti biznis proces modela, proces dijagrama tokova podataka i dijagrama slučajeva korišdenja. **Dijagram klasa** predstavlja sredstvo prikaza statičkih odnosno strukturalnih osobina sistema. Sastoji se od simbola klasa, interfejsa i relacija. Asocijacije su relacije između klasa koje prikazuju sa koliko objekata druge klase je neki objekat povezan. Generalizacija prikazuju kako je neka podklasa povezana sa nadklasom od koje nasleđuje atribute i operacije. Agregacija i kompozicija su odnosi deo-celina.

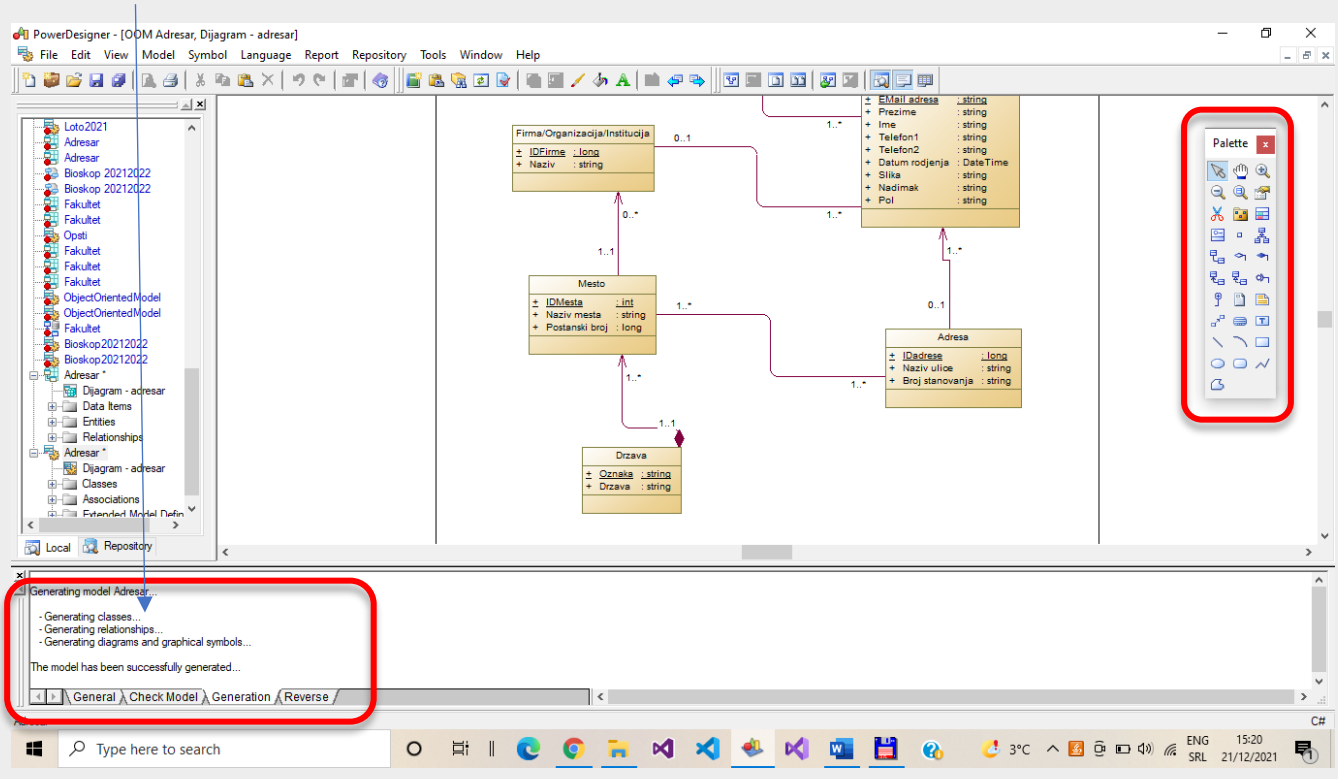
Formiranje dijagrama klasa - Nakon pokretanja Power Designer CASE alata, potrebno je učitati CDM, tj. konceptualni model podataka. Direktno mapiranje CDM modela u objektni model se vrši preko stavke menija "Tools – Generate Object-Oriented Model".



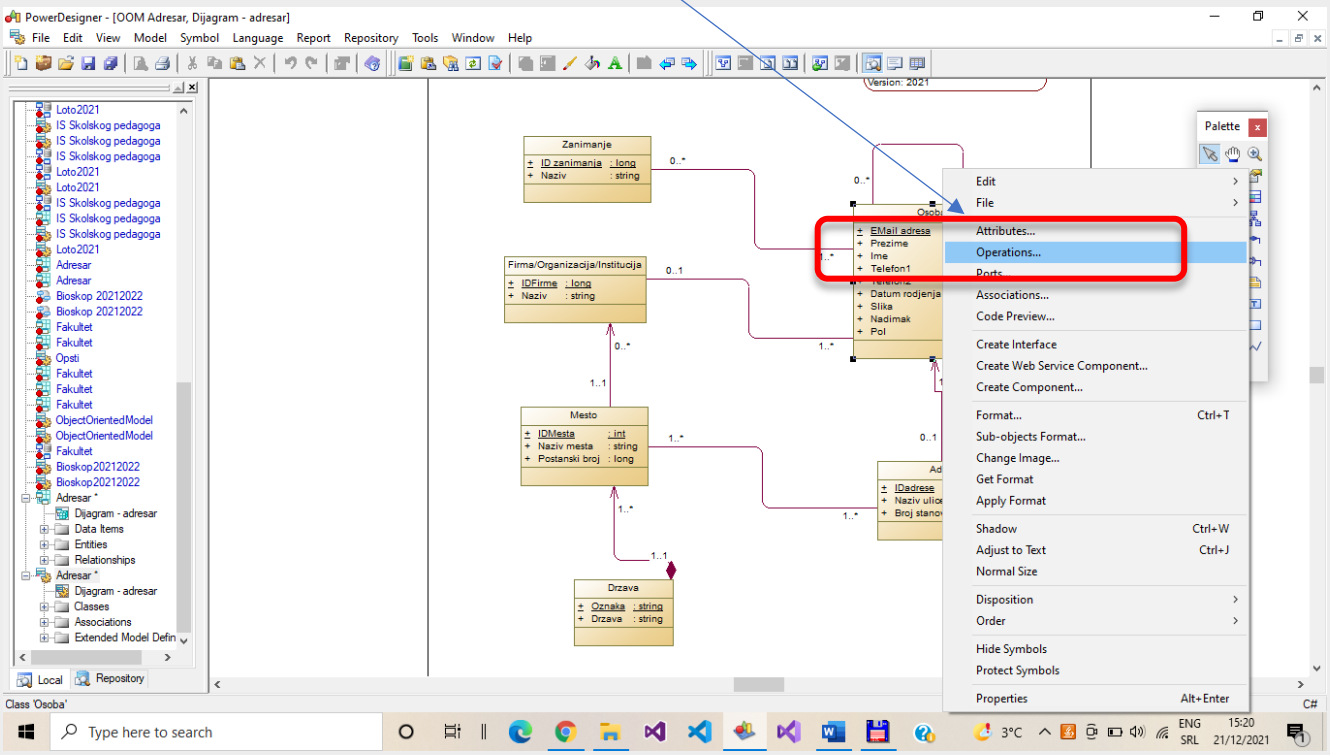
Otvara se dijalog prozor za izbor osobina novog modela kao na sledećoj slici. Bira se implementacioni programski jezik (“Object language”). U primeru izabran Microsoft C#. Može se upisati naziv modela “Model name”, iako je ponuđen isti naziv kao i za CDM model podataka. Kod modela se automatski popunjava od strane CASE alata. Ukoliko postoji već kreirani objektni model podataka, alternativno je moguće izabrati opciju: “Update existing Object-Oriented Model” za kreiranje dijagrama klasa unutar postojećeg OOM modela. Potvrda izbora je “OK”.



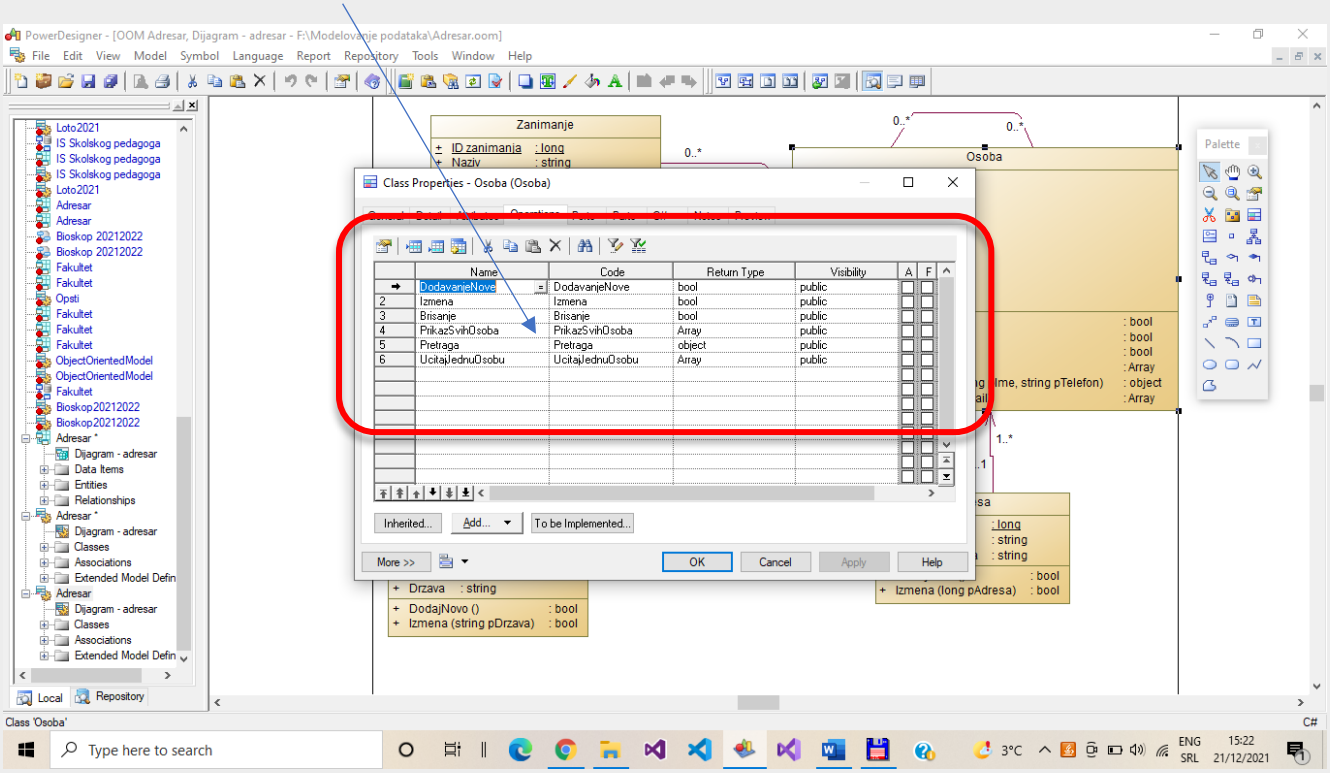
Kreira se novi, popunjeni radni list sa dijagramom klasa i paletom alata za crtanje dijagrama klasa. Informacije o uspešnosti postupka generisanja modela je prikazana u prozoru “Result List” i u kartici “Generation”.



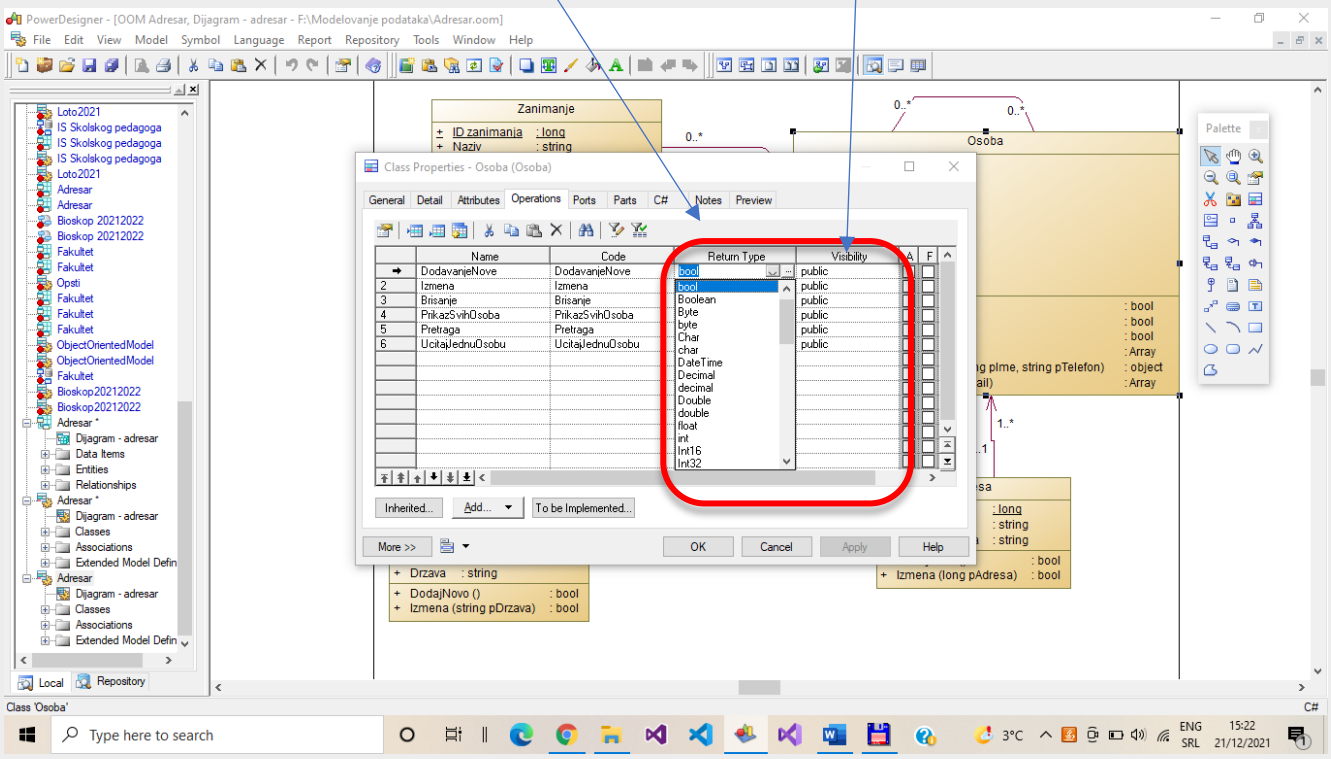
U okviru OOM modela kreiranim na ovakav način je potrebno definisati operacije koje se izvršavaju nad klasom, tj. metodama, procedurama ili funkcijama. Klase se mogu menjati ili definisati nove, ali u ovom tutorijalu taj postupak neće biti opisan, pošto se na časovima nije ni radilo na taj način. Bira se željena klasa, te se pokreće otvaranje prozora za određivanje karakteristika operacija (desni klik mišem kada je klasa u fokusu):



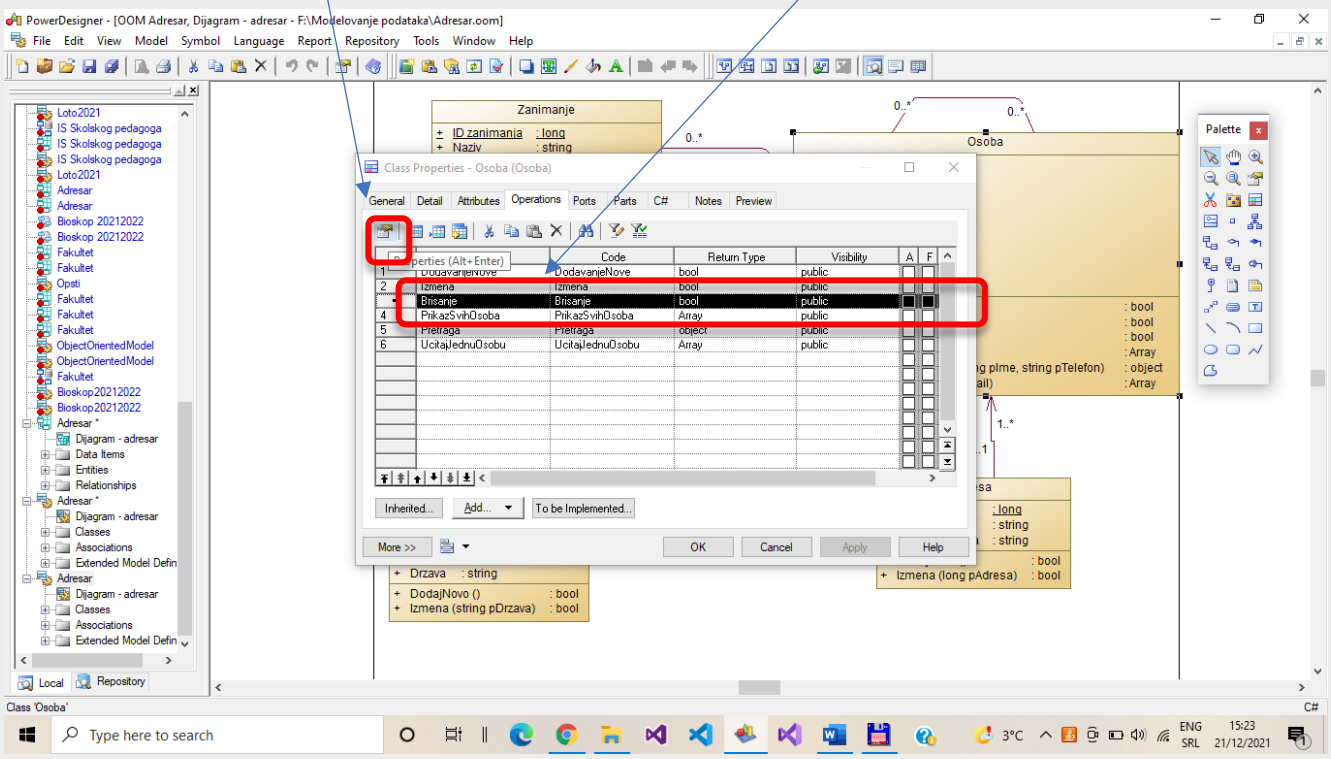
Otvara se dodatni dijalog prozor “Class Properties” sa aktivnom trećom karticom - “Operations”. U tabelu se unose nazivi metoda. Kod se, kao i u svim situacijama, popunjava automatski.



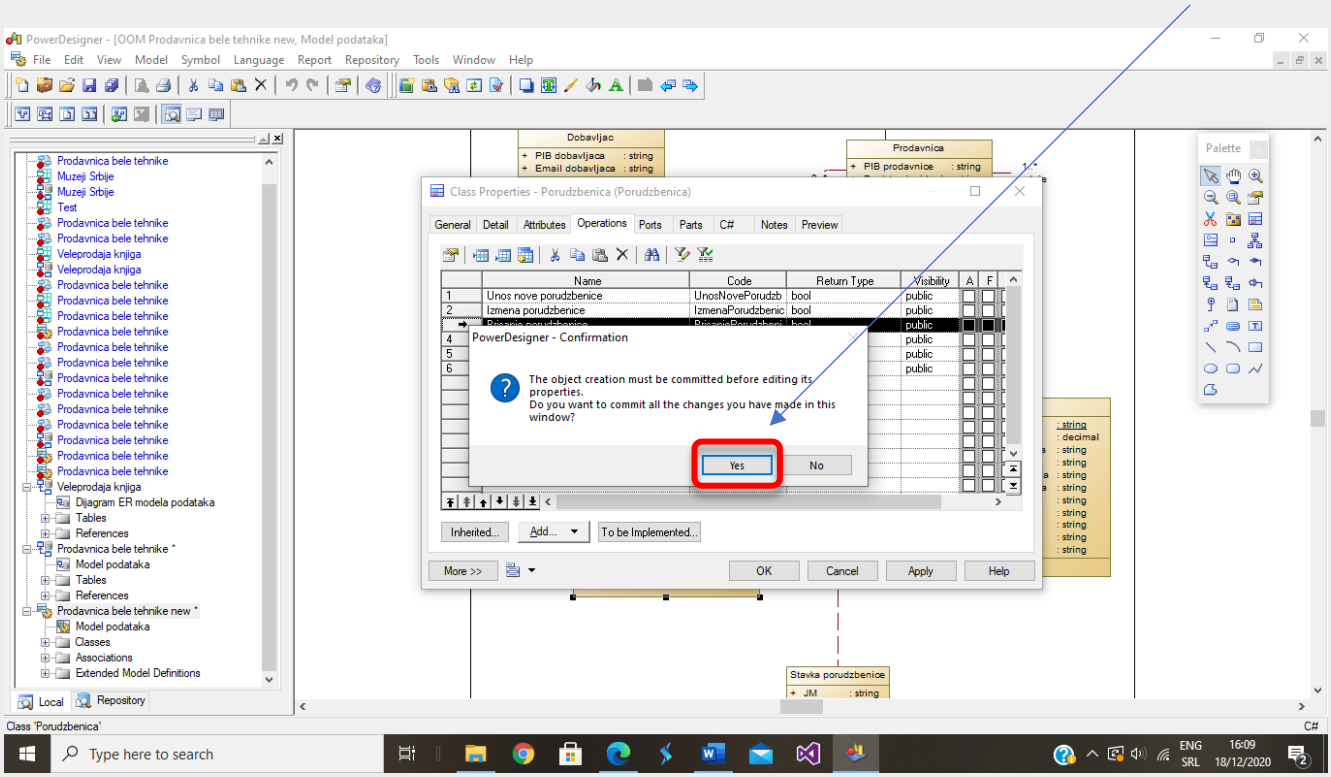
Projektant bira vrstu metode preko “Return Type” kolone, kako bi odredio da li je upitanju funkcija koja vraća objekat, neki konkretan tip podatka(int, bool, char, string, decimal, double, float…) i sl., ili procedura koja ne vraća vrednost (u C# jeziku je to “void”). Vidljivost (public, private, protected) je u koloni “Visibility”.



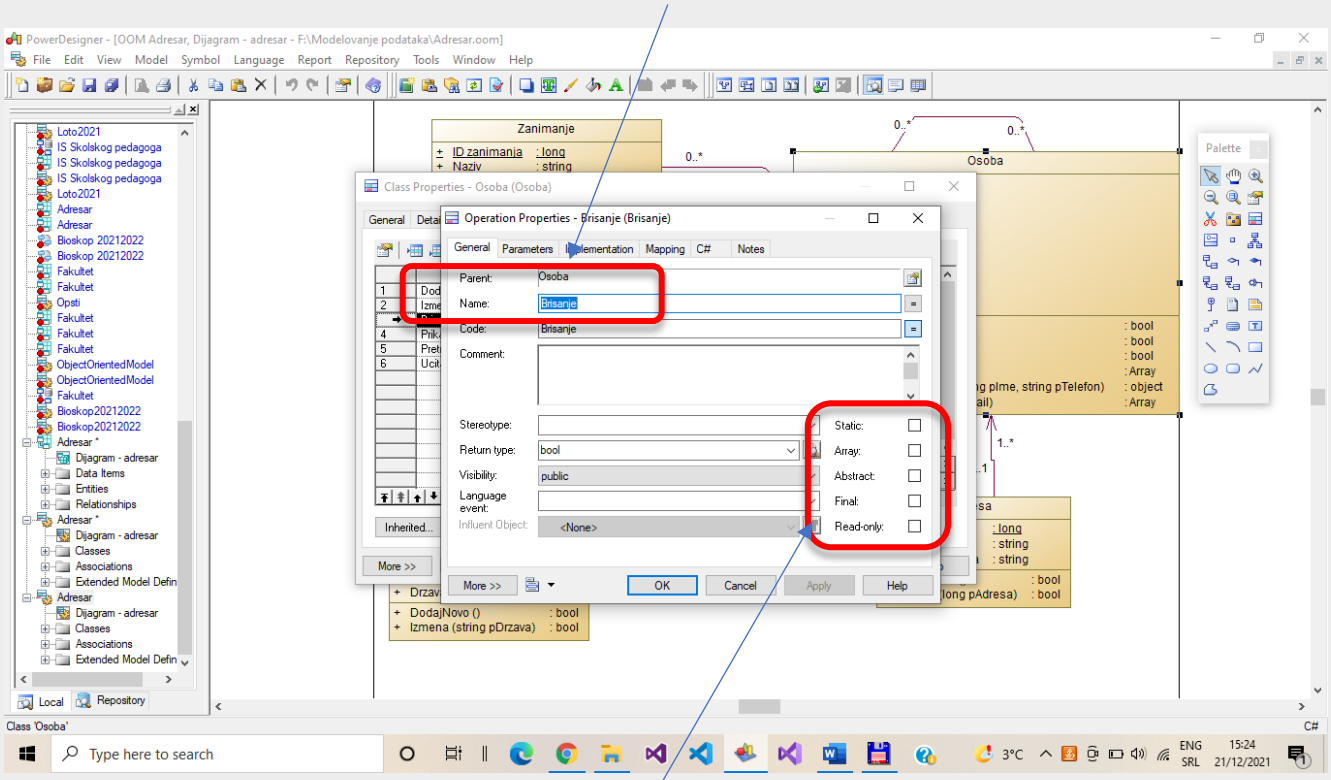
Definisanje parametara metode se vrši preko osobina metoda “Properties”. Aktivira se prvim tasterom sa leve strane na paleti alata, ali isključivo za metodu koja je selektovana.



Sledi dijalog za potvrdu promena osobina objekta “Object creation commit”. Bira se “Yes” za potvrdu.

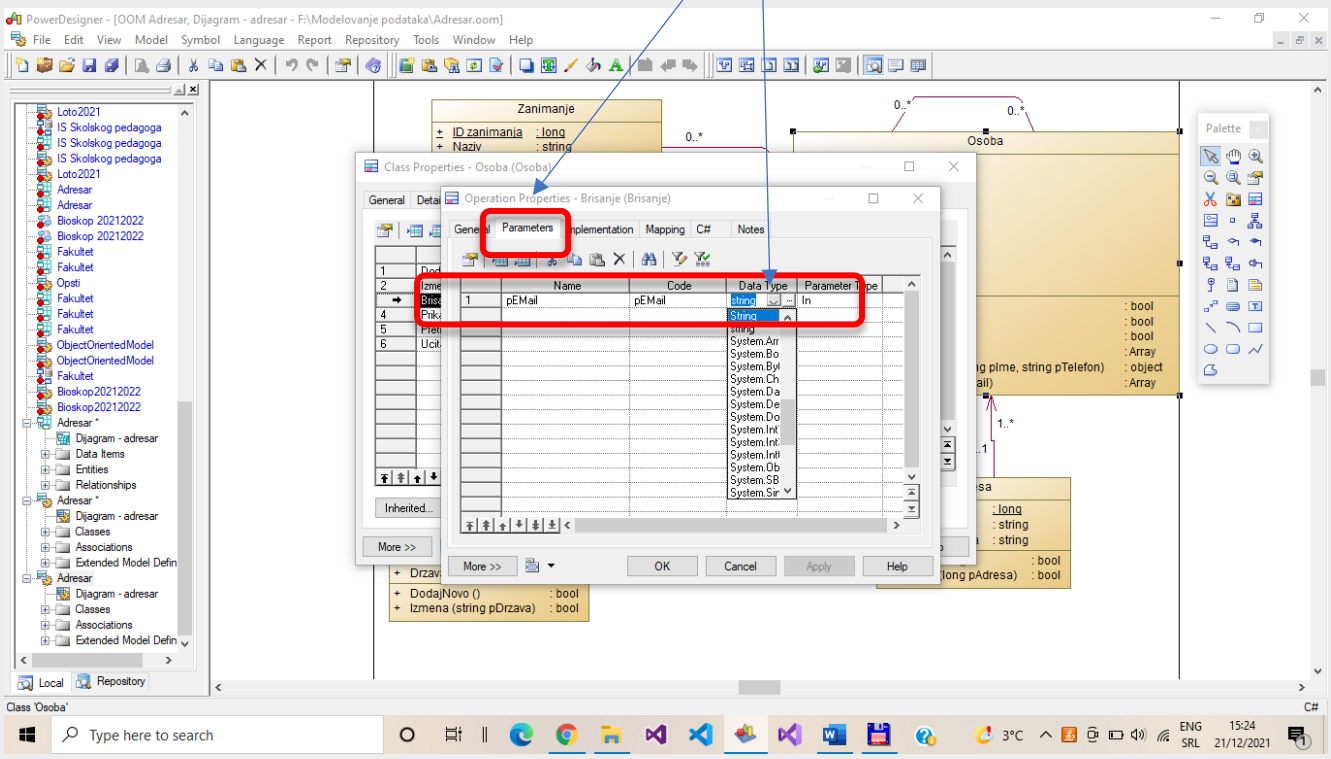


Otvara se prva kartica osobina operacija klase. Ime (“Name”) je već uneto u prethodnom prozoru.

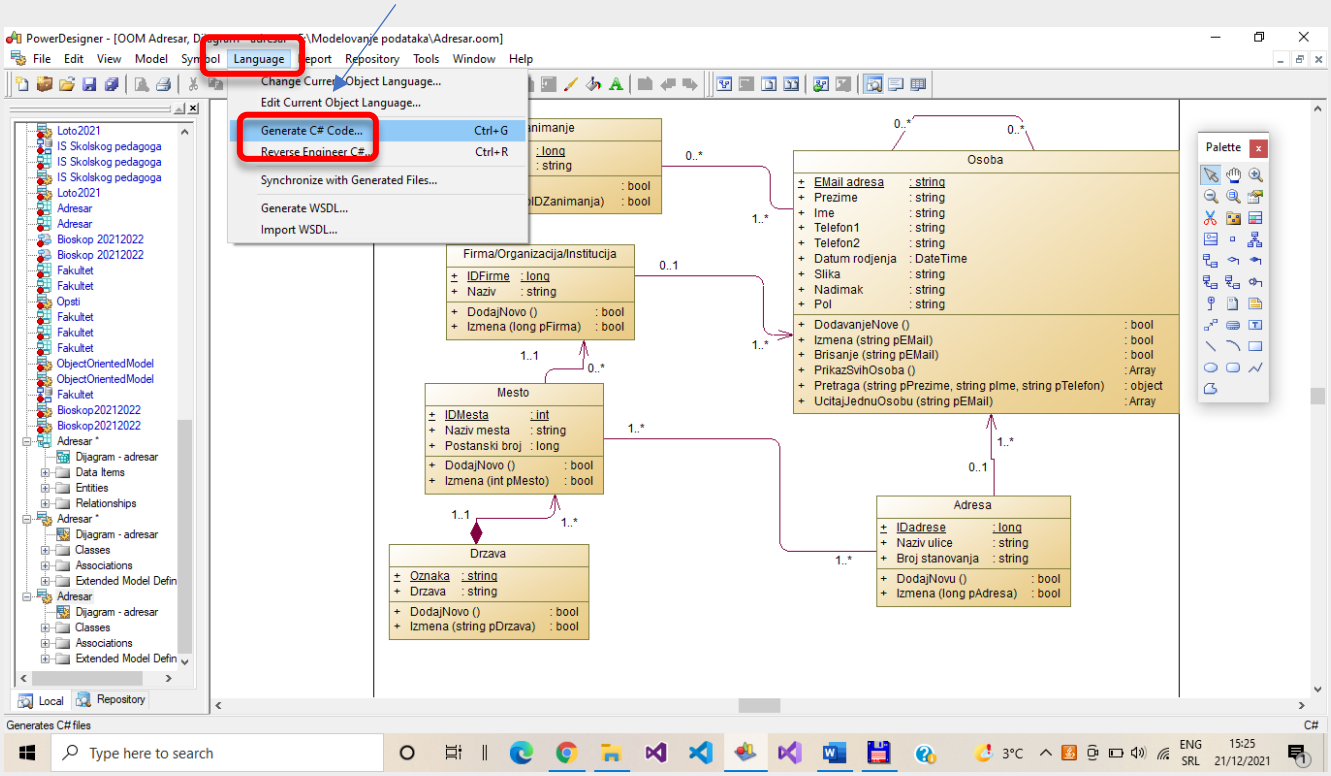


Mogu se odrediti osobine operacije, tj. metode: statik, abstraktna, final.

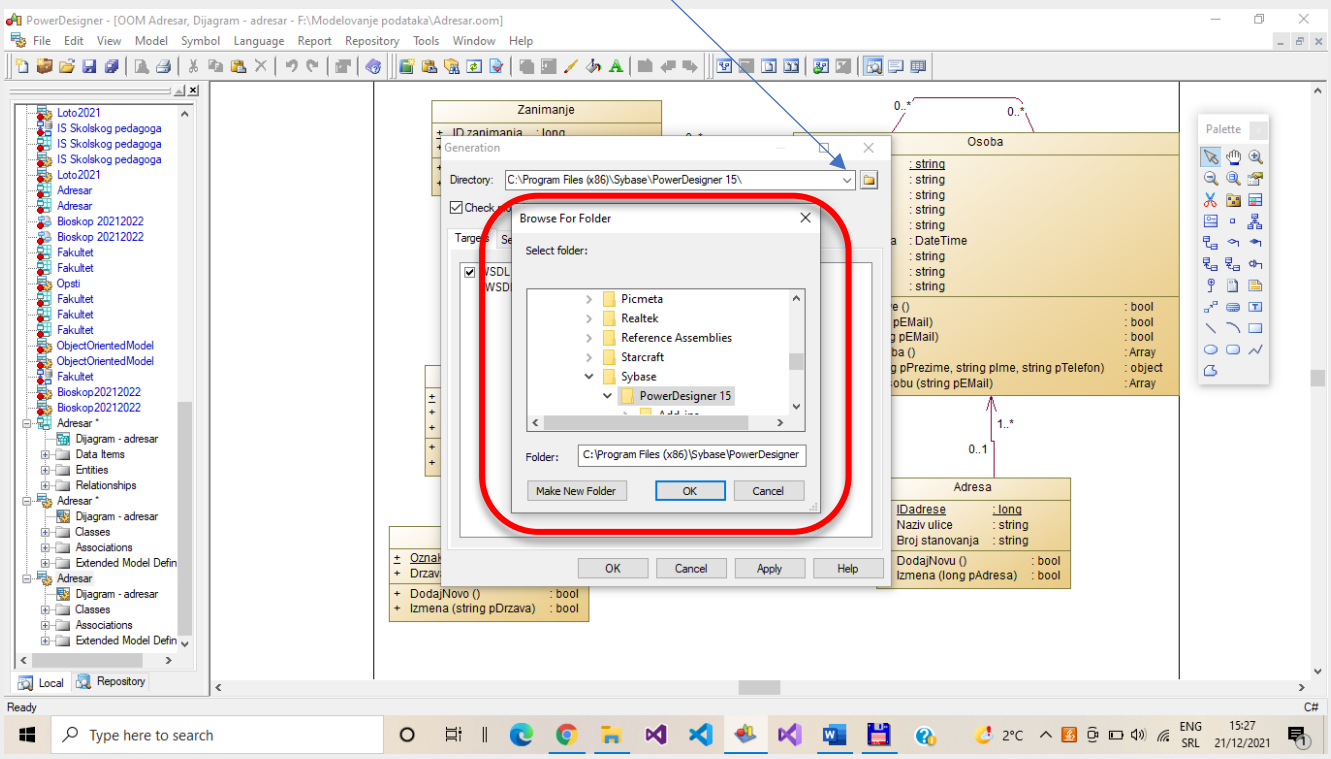
Za određivanje parametara operacije potrebno je aktivirati drugu karticu - “Parameters”. Potrebno je uneti naziv parametra (“Name”), tip podatka (“Data Type”) i vrstu parametra: ulazni – “In”, izlazni “Out” ili ulazno/izlazni “In/out”. Voditi računa da tip podatka parametra mora da odgovara koloni tabele baze podataka ukoliko se radi o parametru koji je povezan sa nekom kolonom iz baze podataka.



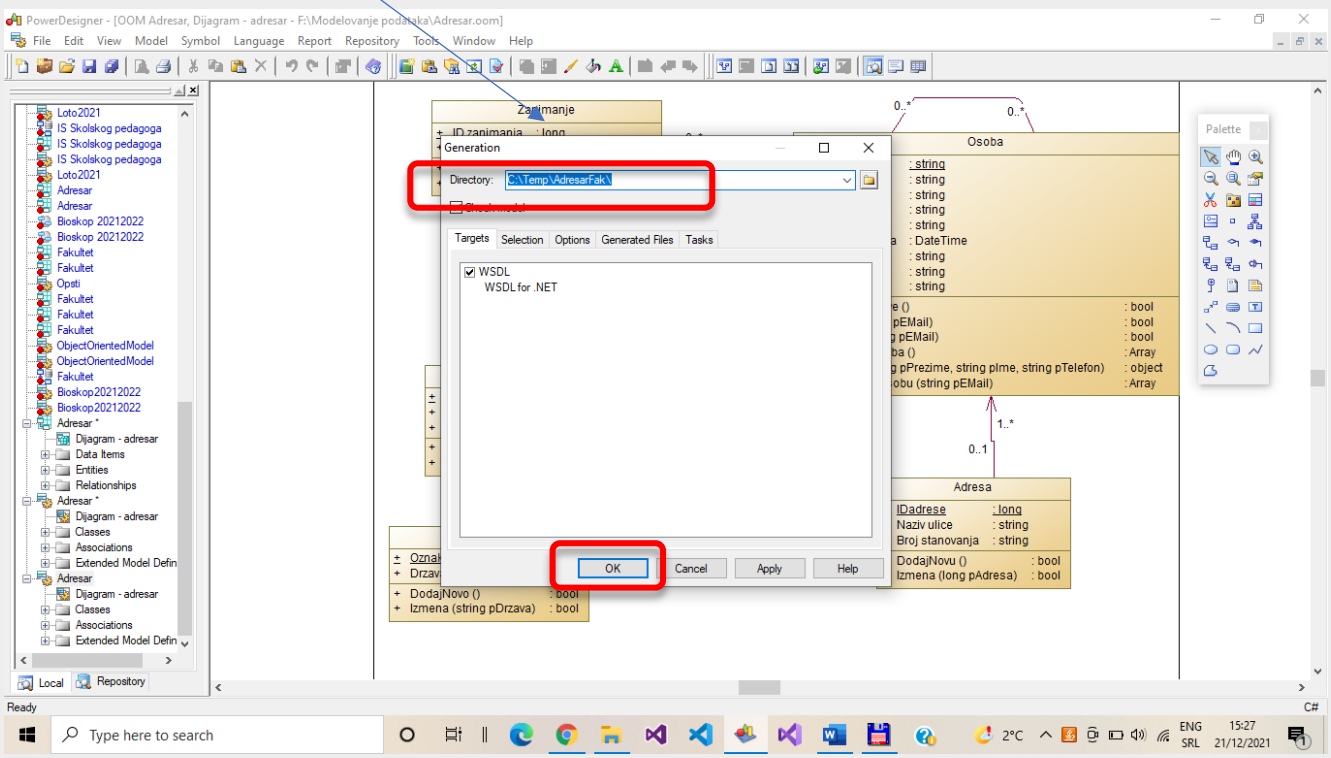
Kada se završi definisanje operacija u svim klasama i njihovih parametara, može se pristupiti implemtaciji objektnog modela u izabranom programskom jeziku. Bira se, sa glavnog menija CASE alata, opcija “Language – Generate C# Code...”.



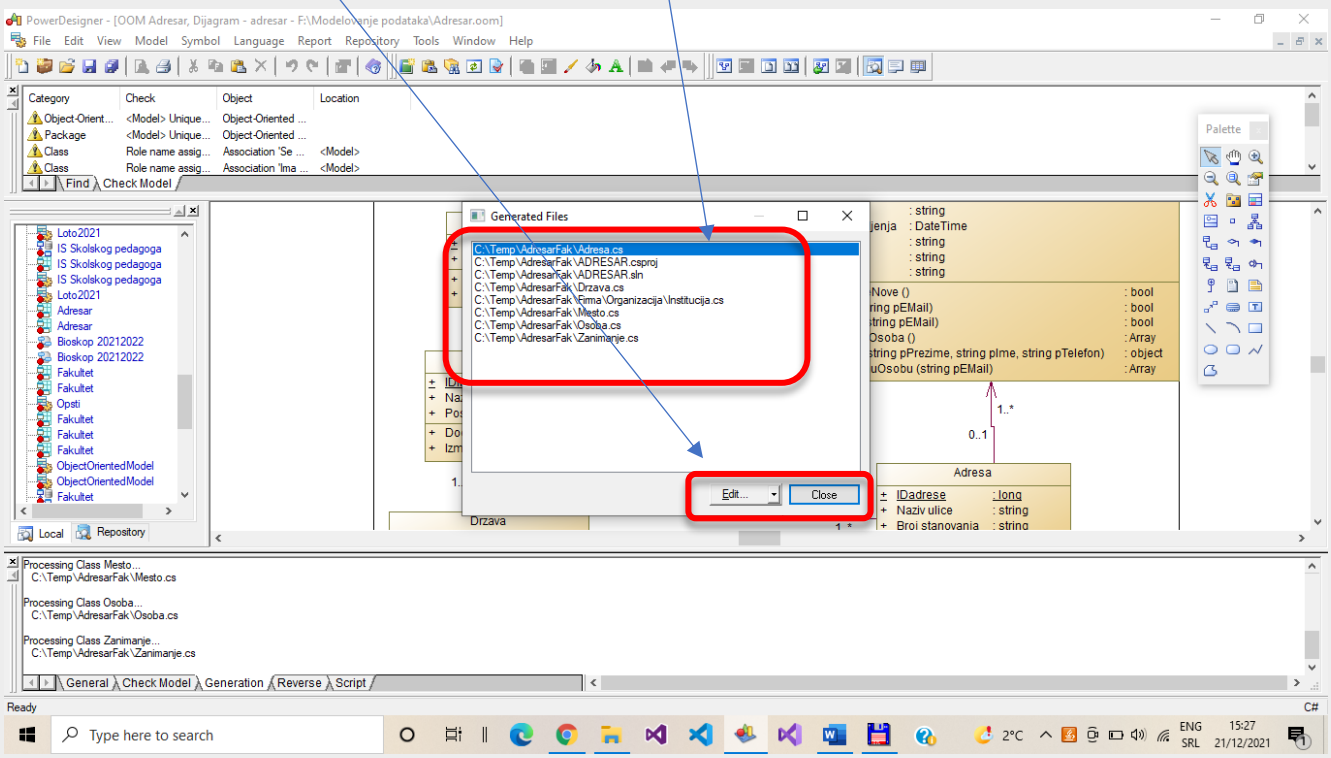
Sledi izbor foldera u koji će CASE alat kreirati projekat tipa “Class Library” za bilioteku klasa koja odgovara apstraktnoj slici iz OOM modela.



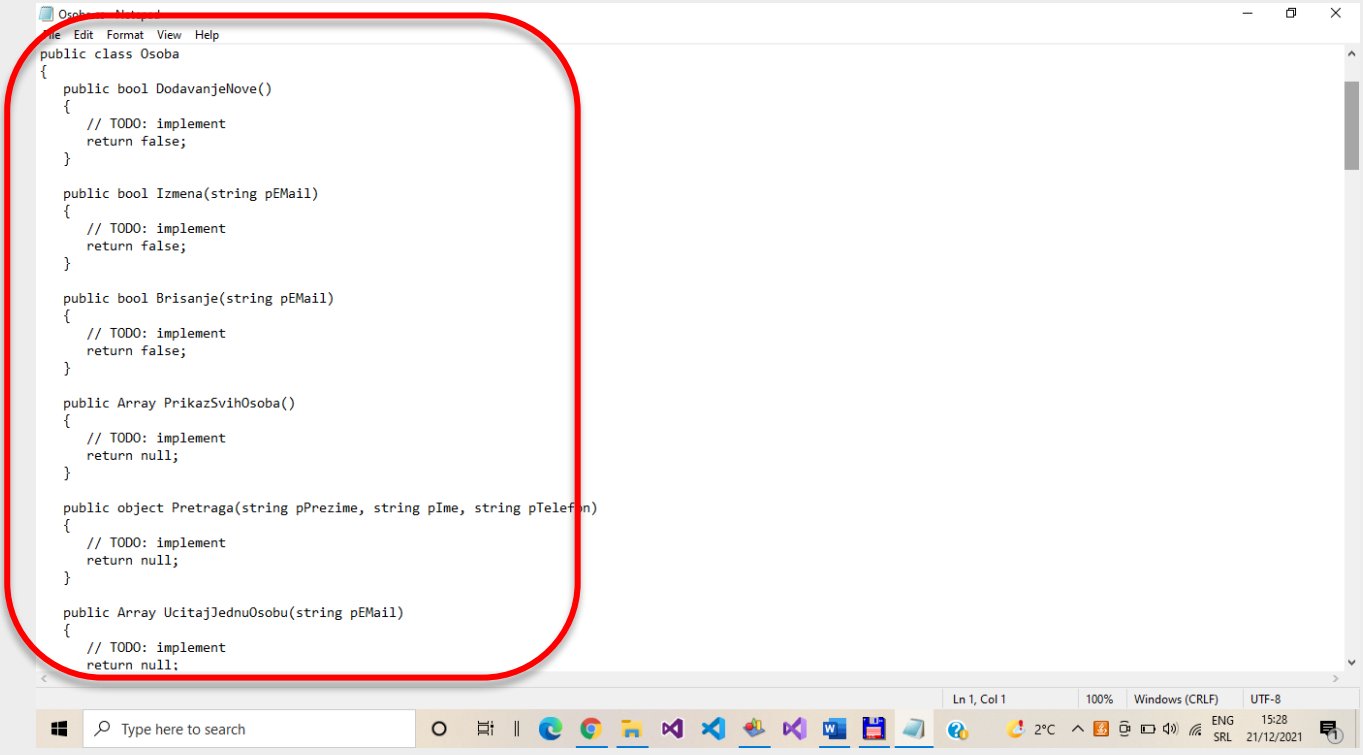
Izabrani folder u primeru je upisan u rubriku “Directory”, u okviru prozora “Generation”. Završetak postupka je preko tastera “Ok”.



Sledi prikaz informacija o generisanim fajlovima/datotekama. Svaku datoteku je moguće otvoriti u nekom tekst procesoru/editoru ili u razvojnom okruženju programskog jezika (u primeru bi to bio MS Visual Studio .NET). Opcija je “Edit”, tj. izmena. Taster “Close” zatvara dijalog bez pregleda klase.



Prikaz klase “Osoba.cs”. Može se primetiti da su deklarisani atributi, definisana zaglavlja metoda, dok je programerima ostalo da napišu telo svake metode.



```
public class Osoba
{
   public bool DodavanjeNove()
   {
      // TODO: implement
      return false;
   }

   public bool Izmena(string pEMail)
   {
      // TODO: implement
      return false;
   }

   public bool Brisanje(string pEMail)
   {
      // TODO: implement
      return false;
   }

   public Array PrikazSvihOsoba()
   {
      // TODO: implement
      return null;
   }

   public object Pretraga(string pPrezime, string pIme, string pTelefon)
   {
      // TODO: implement
      return null;
   }

   public Array UcitajJednuOsobu(string pEMail)
   {
      // TODO: implement
      return null;
```