NAMA : Laurensius Nathan Surya Rachmanto

NIM : 225150400111049

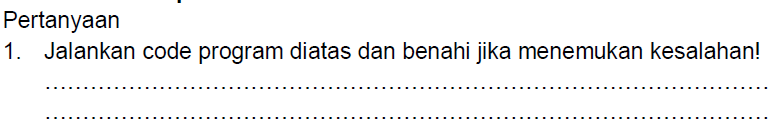
KELAS : SI - B

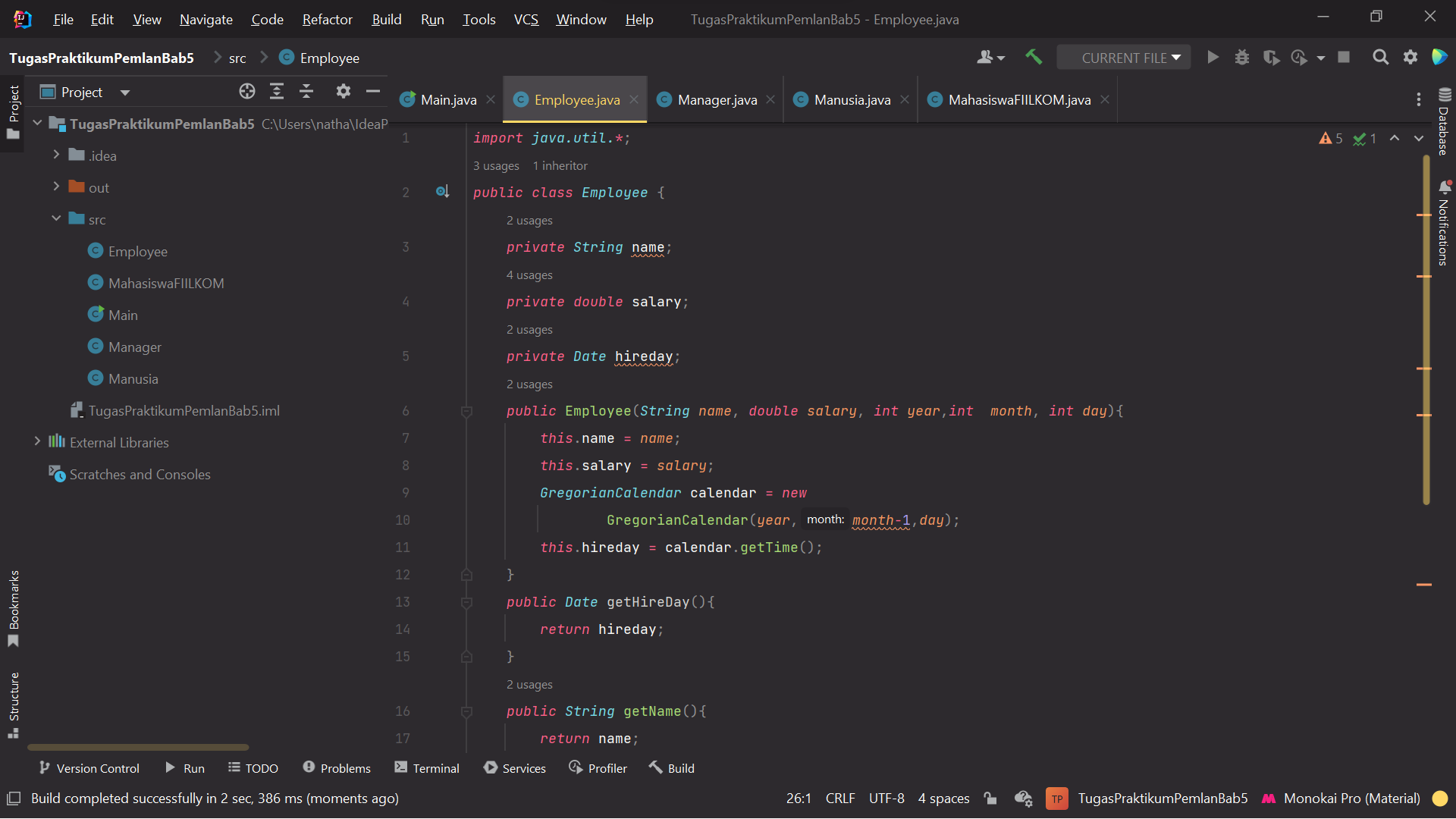
BAB : 5

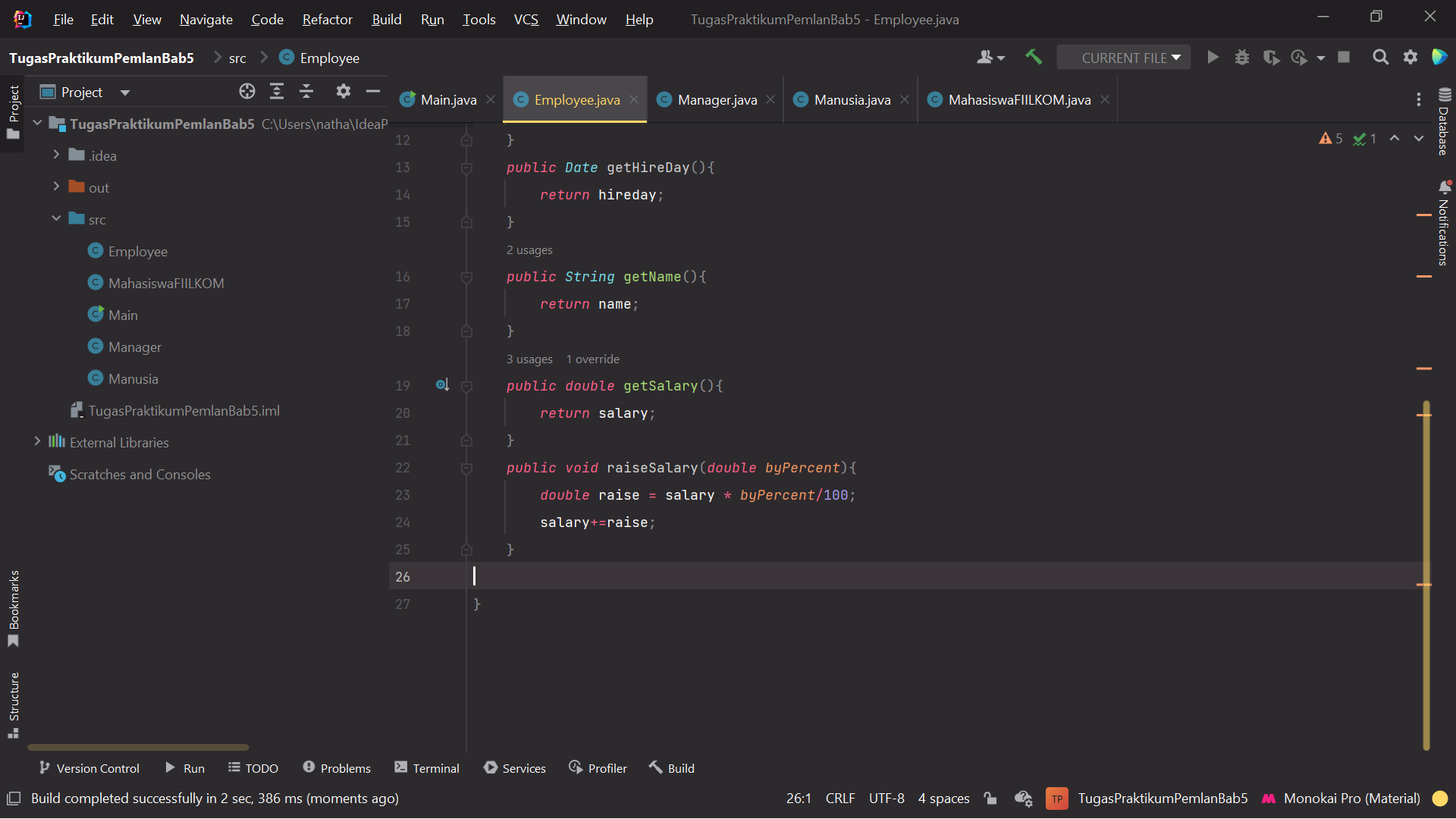
ASISTEN :Fahru Iskandar Setiawan

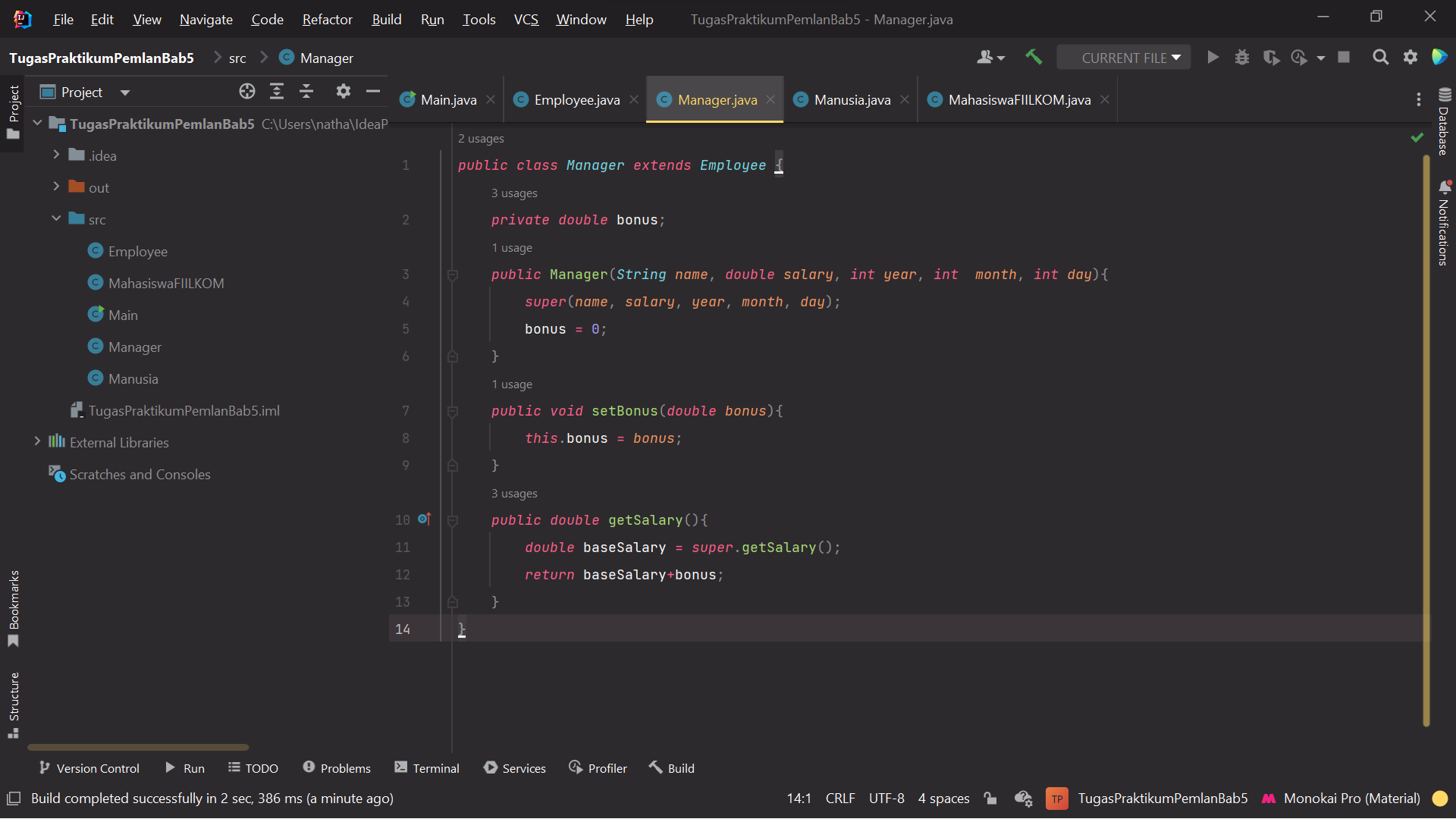


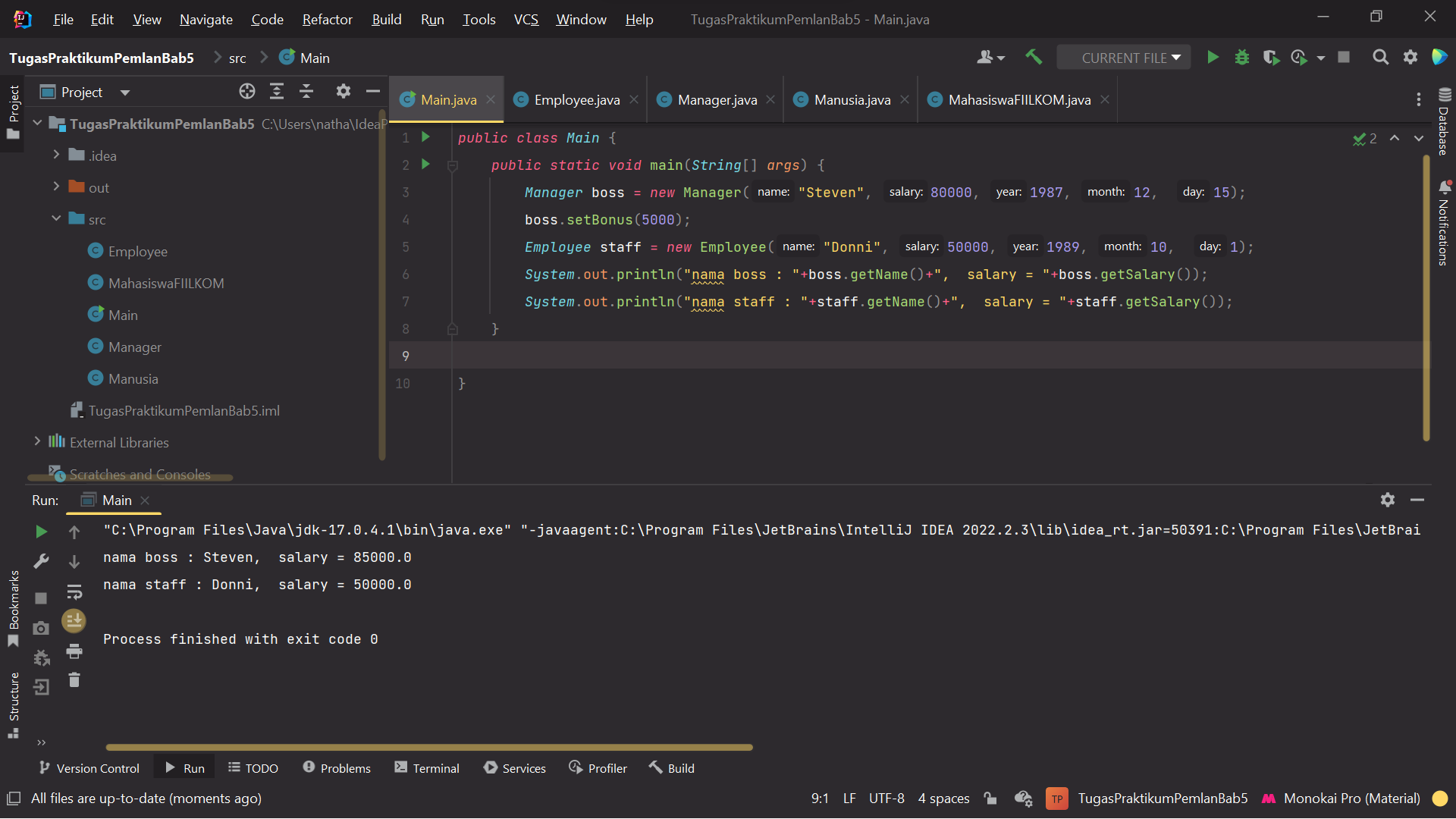
1. **Data dan Analisis hasil percobaan**

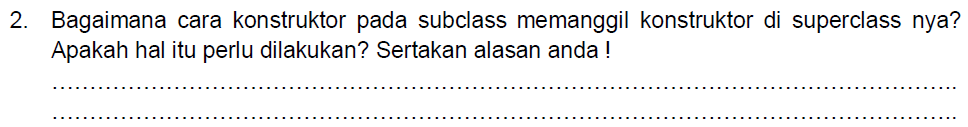


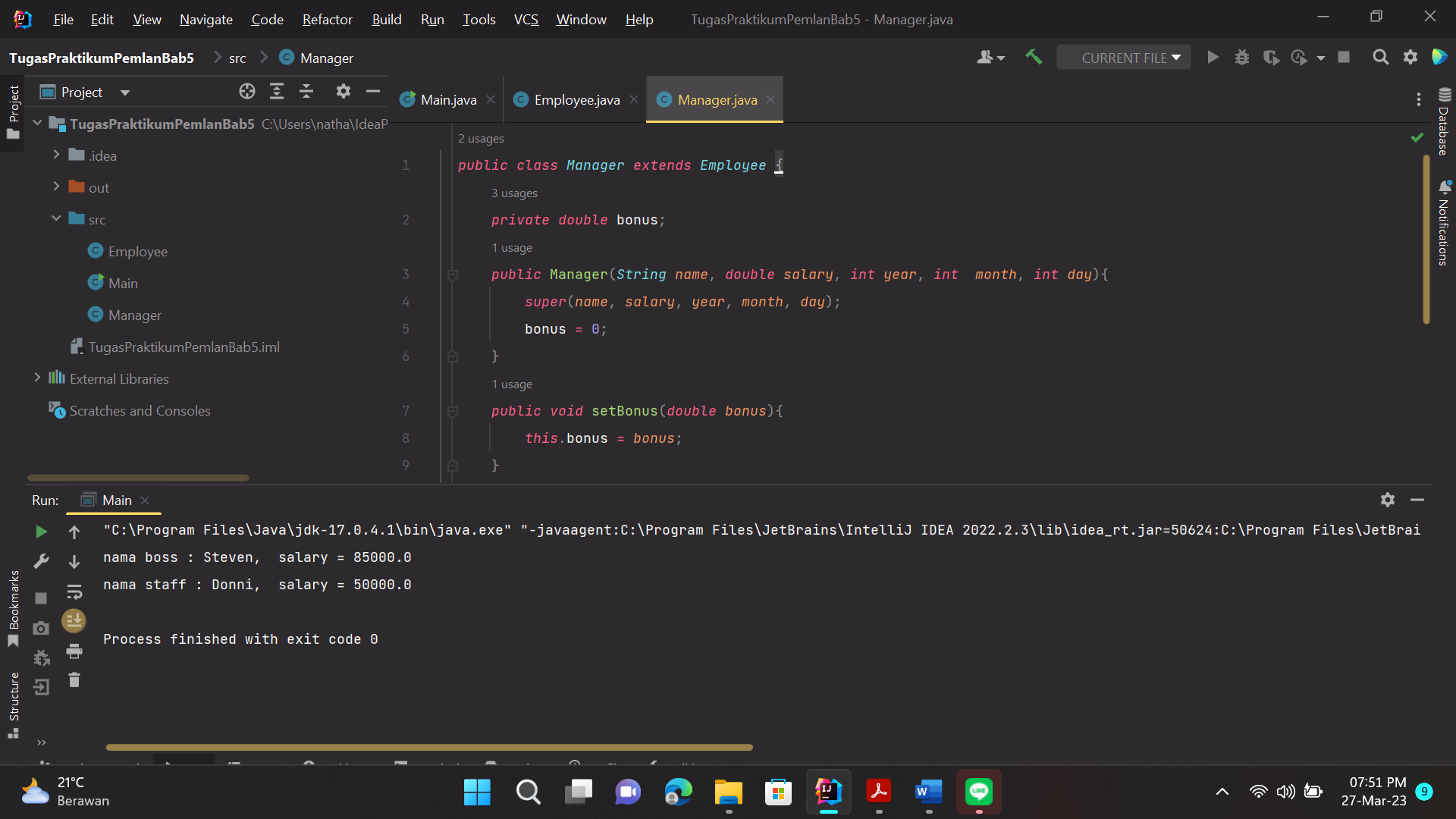




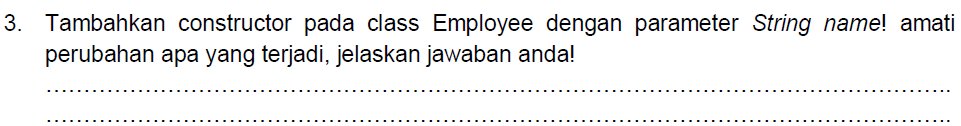


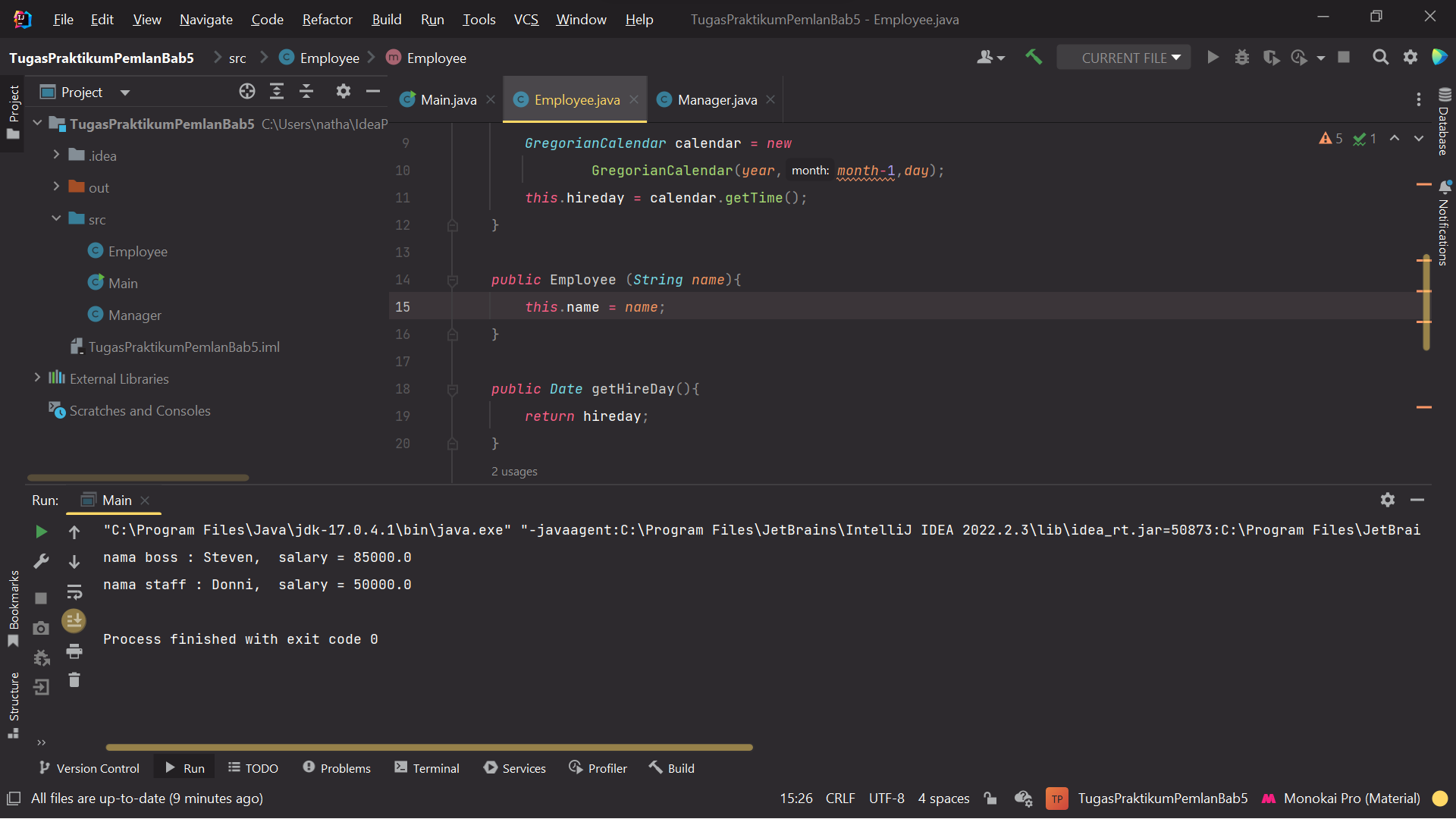




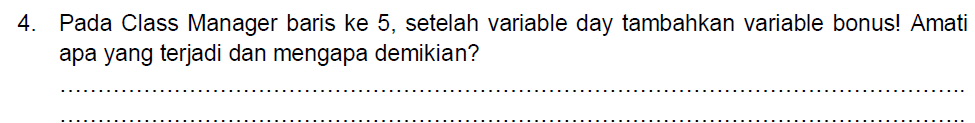


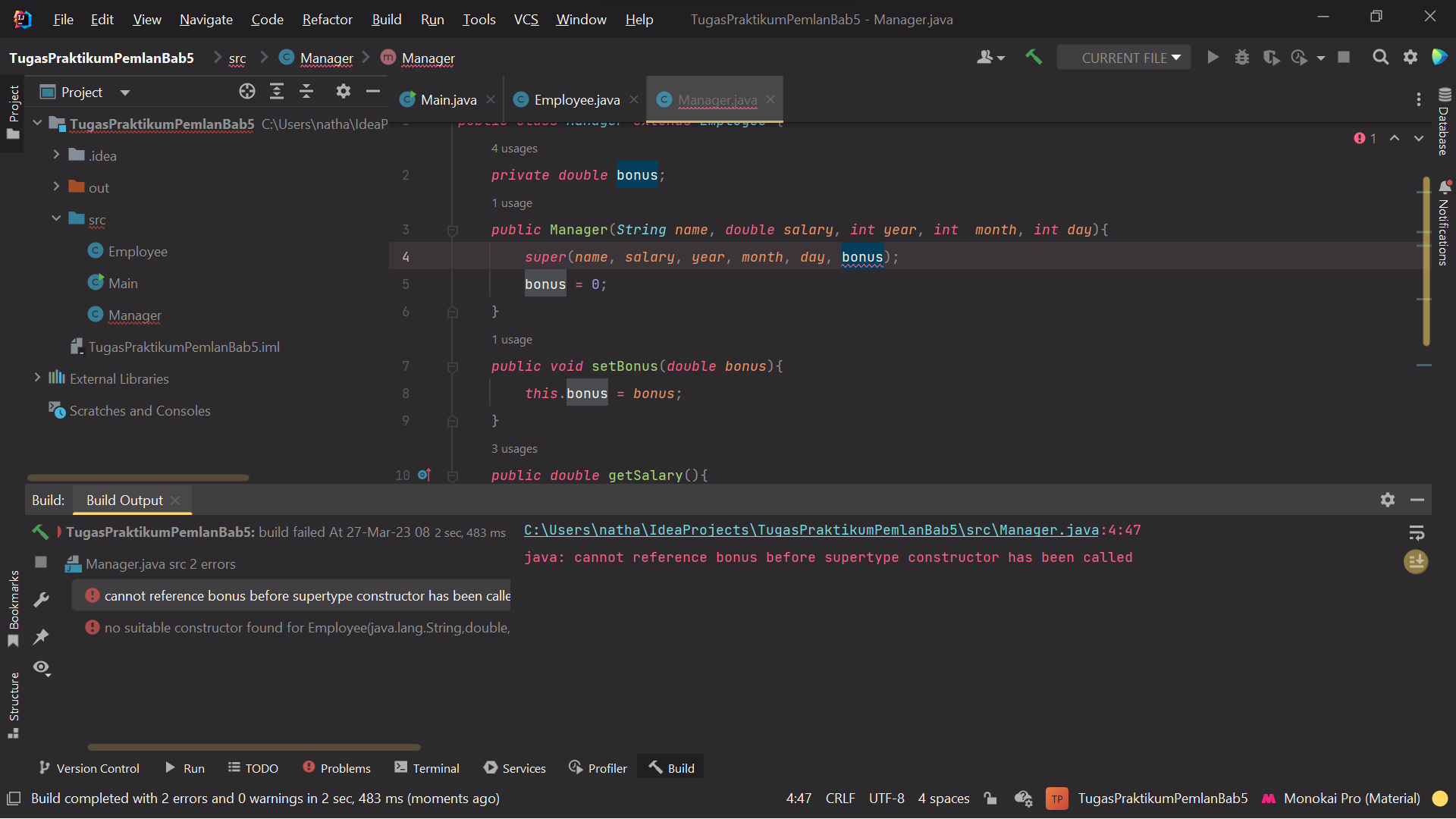
Seperti pada contoh gambar di atas pada line ke 3 sampai 6, cara untuk memanggil superclass dari suatu subclass atau kelas turunan adalah dengan statement “super” seperti pada line 4 pada gambar. Line 4 merujuk pada konstruktor dari kelas employee yang merupakan superclass dari kelas manajer. Menurut saya hal ini diperlukan untuk alasan efektivitas dan efisiensi. Dengan metode pemanggilan konstruktor superclass pada subclass, programmer tidak perlu 2 kali kerja untuk membuat atribut yang sama dari beberapa subclass yang berbeda.



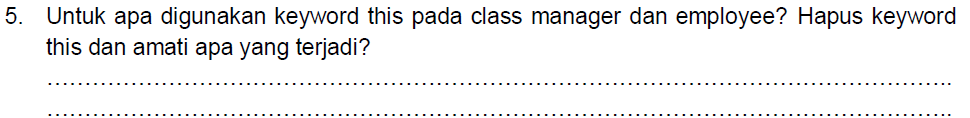


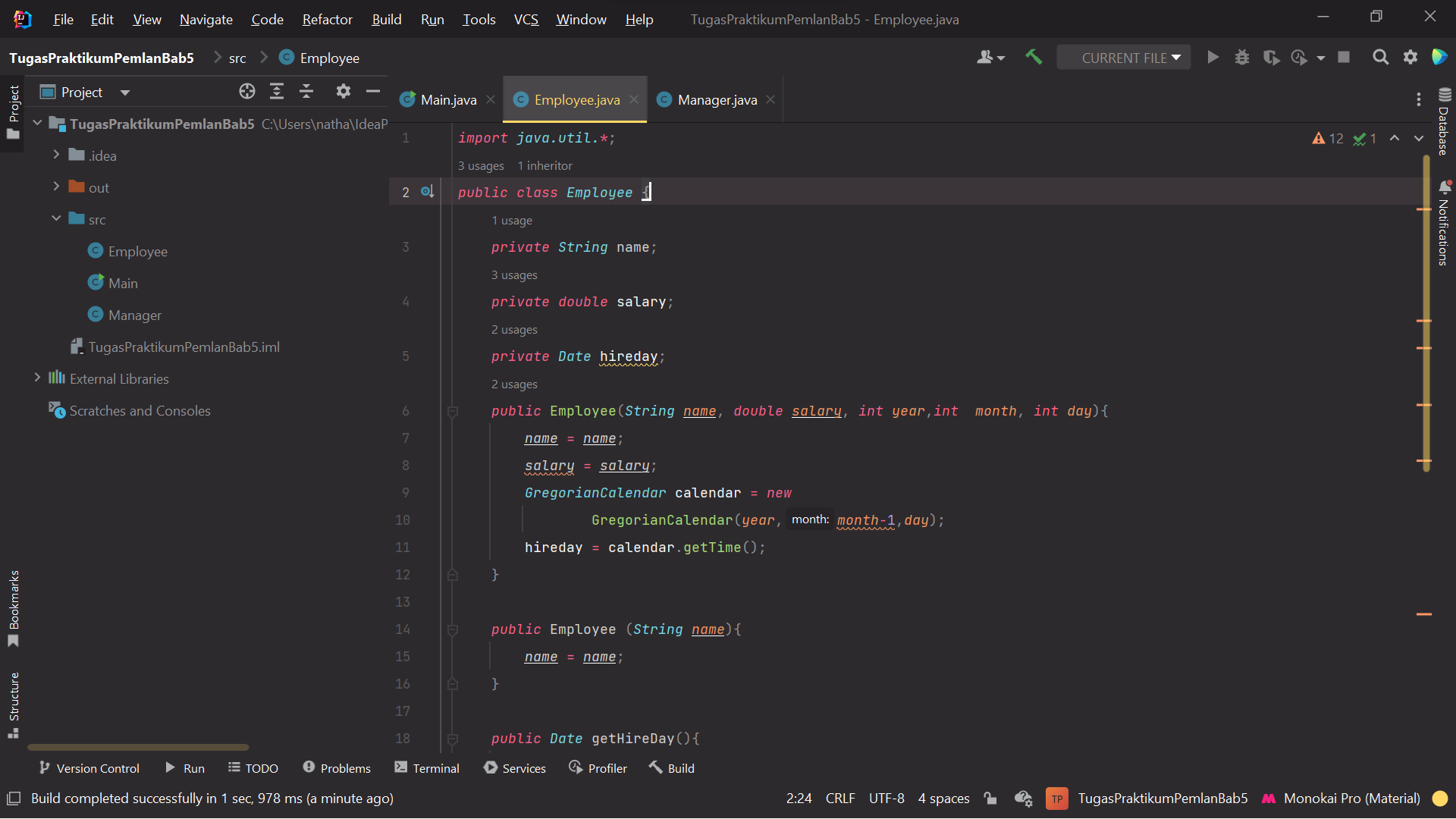
Tidak ada perubahan yang terjadi. Hal ini disebut dengan overloading konstruktor, dimana kostruktor suatu kelas bisa memiliki behaviour berbeda dengan syarat memiliki parameter dan tipe data parameter yang berbeda.

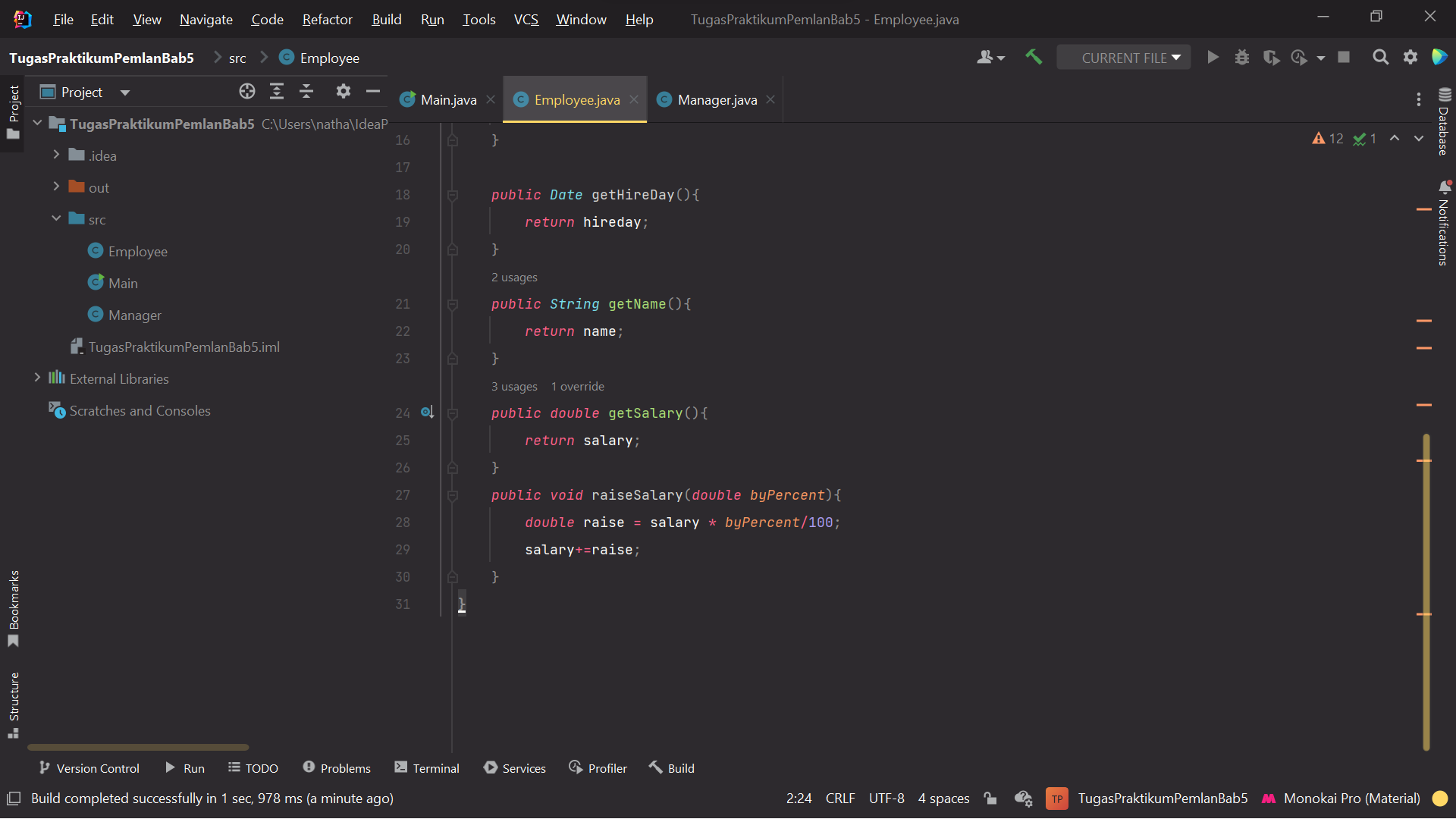


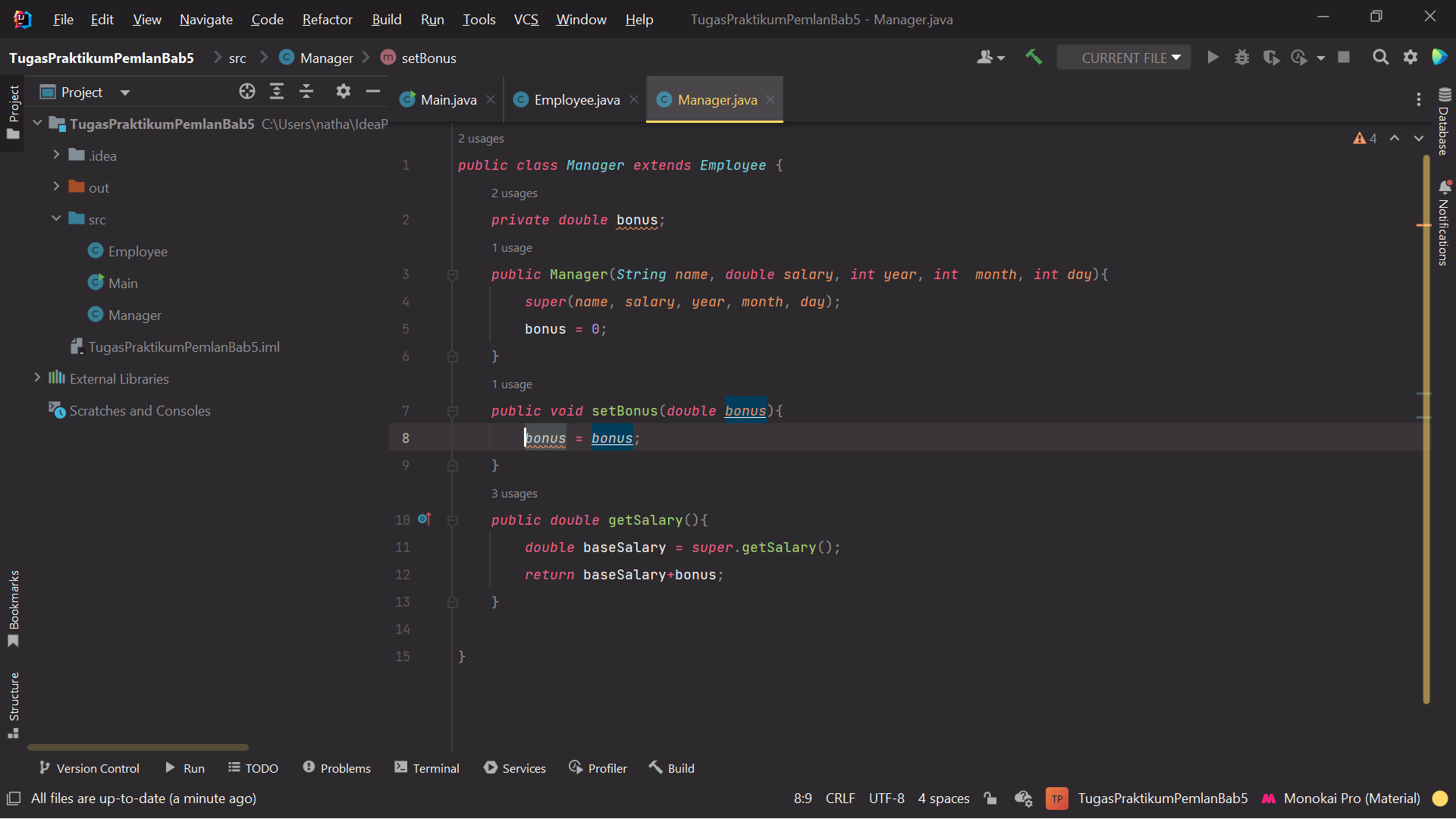


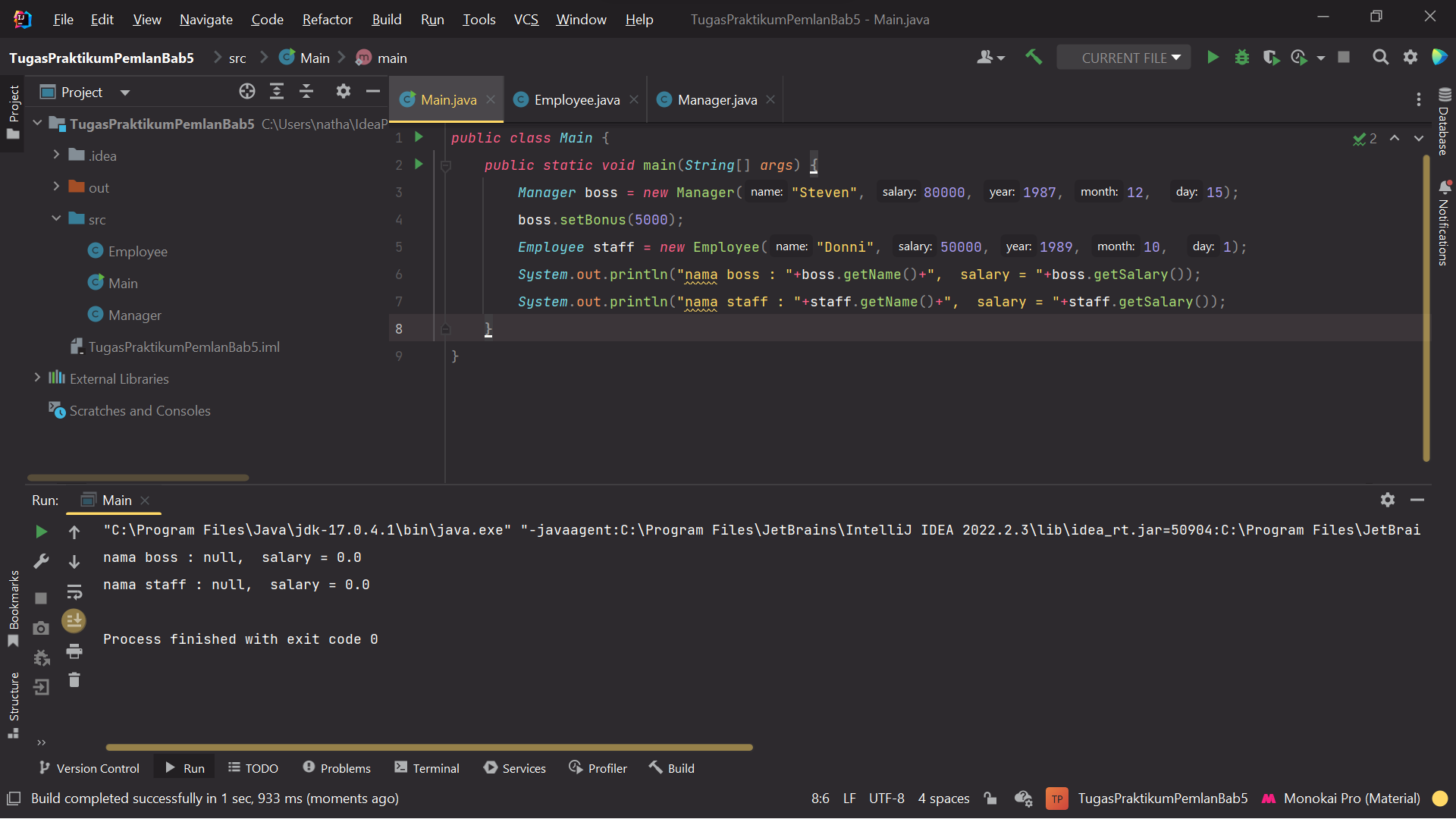
Akan terjadi error. Hal ini dikarenakan konstruktor pada superclass employee hanya diinisiasikan untuk memiliki 5 parameter pada konstruktornya, sehingga apabila kita menaruh 6 parameter akan terjadi error.



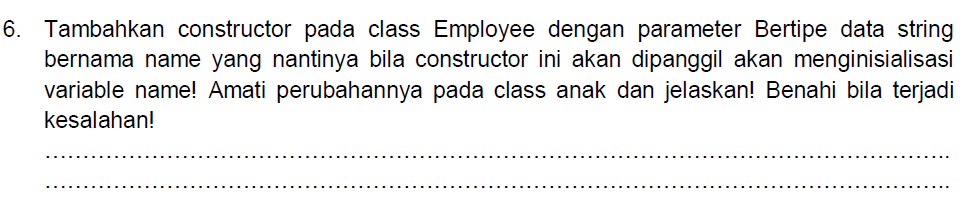


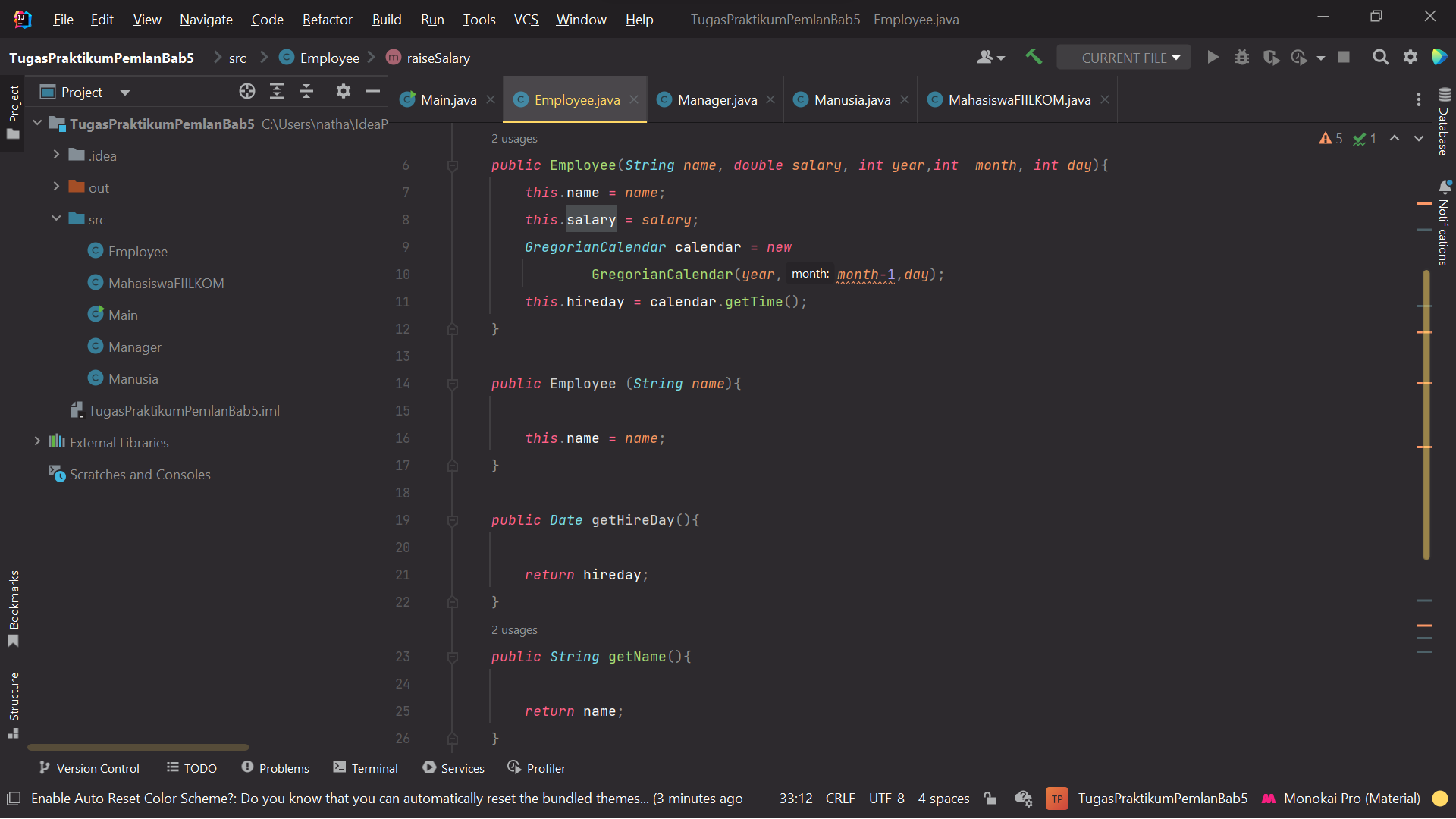


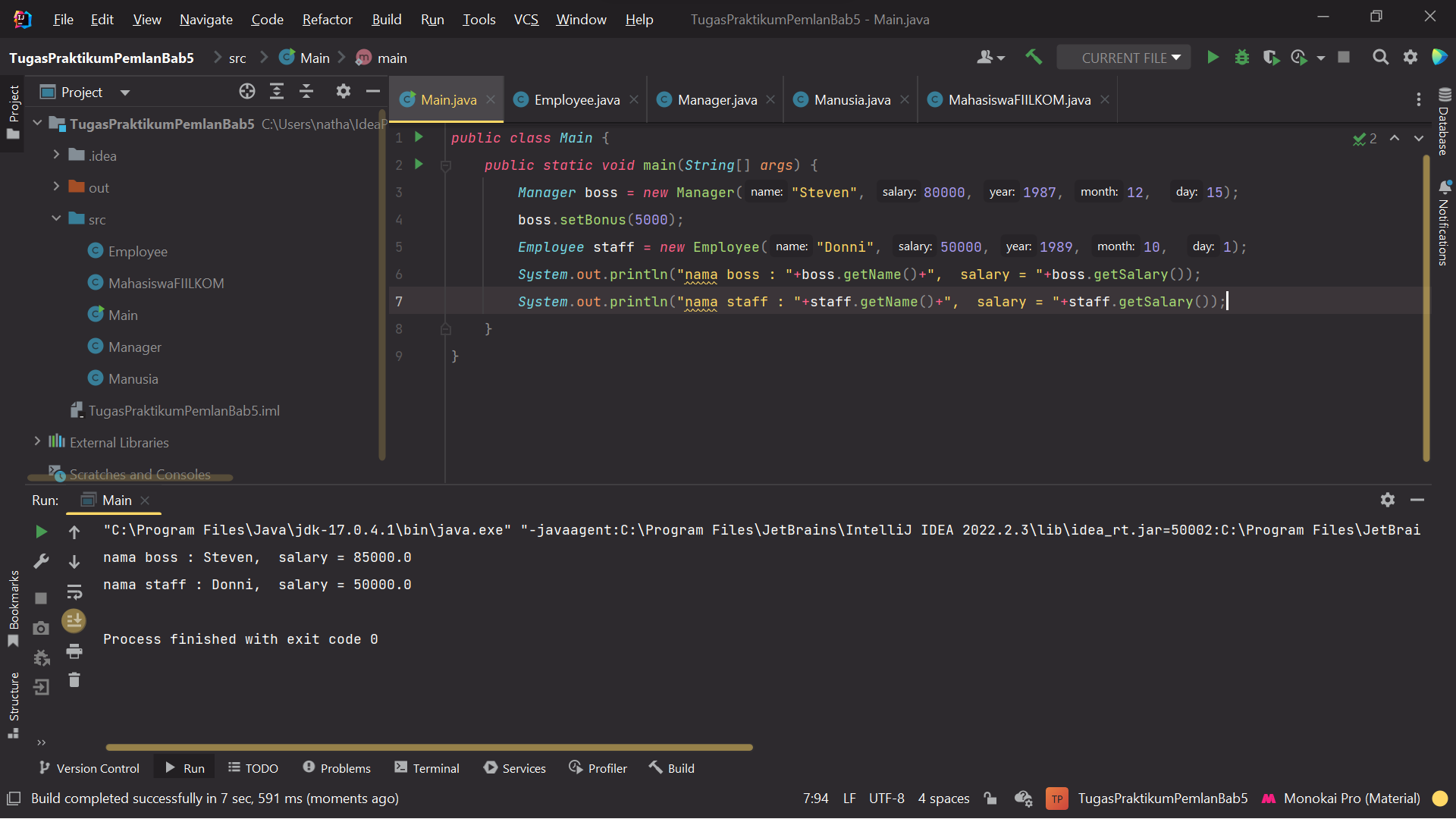


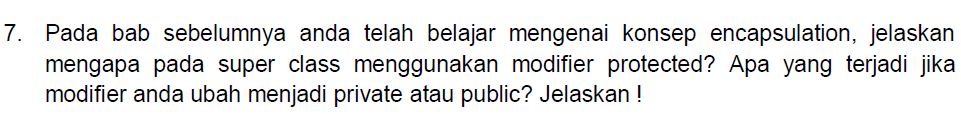


Keyword “this” pada kelas di atas untuk menunjuk pada local variables. Jadi apabila keyword “this” dihilangkan pada setiap konstruktor pada kelas employee dan manager, maka local variable mereka tidak dapat di set value dari parameter yang sudah ditentukan karena tidak ada yang menunjuk pada local variable tersebut.

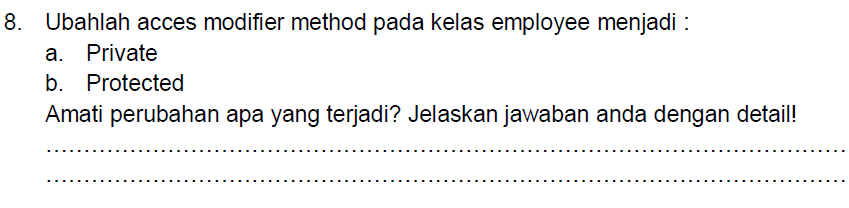




Tidak ada perubahan dikarenakan hal tersebut merupakan overloading constructor. Pada subclass manager pun tidak berpengaruh, karena method tersebut sama sekali tidak dipanggil pada subclass manager maupun kelas main. Apabila ada pemanggilan konstruktor tersebut pun tidak akan terjadi error, hanya saja atribut selain nama jadi belum memiliki nilai.

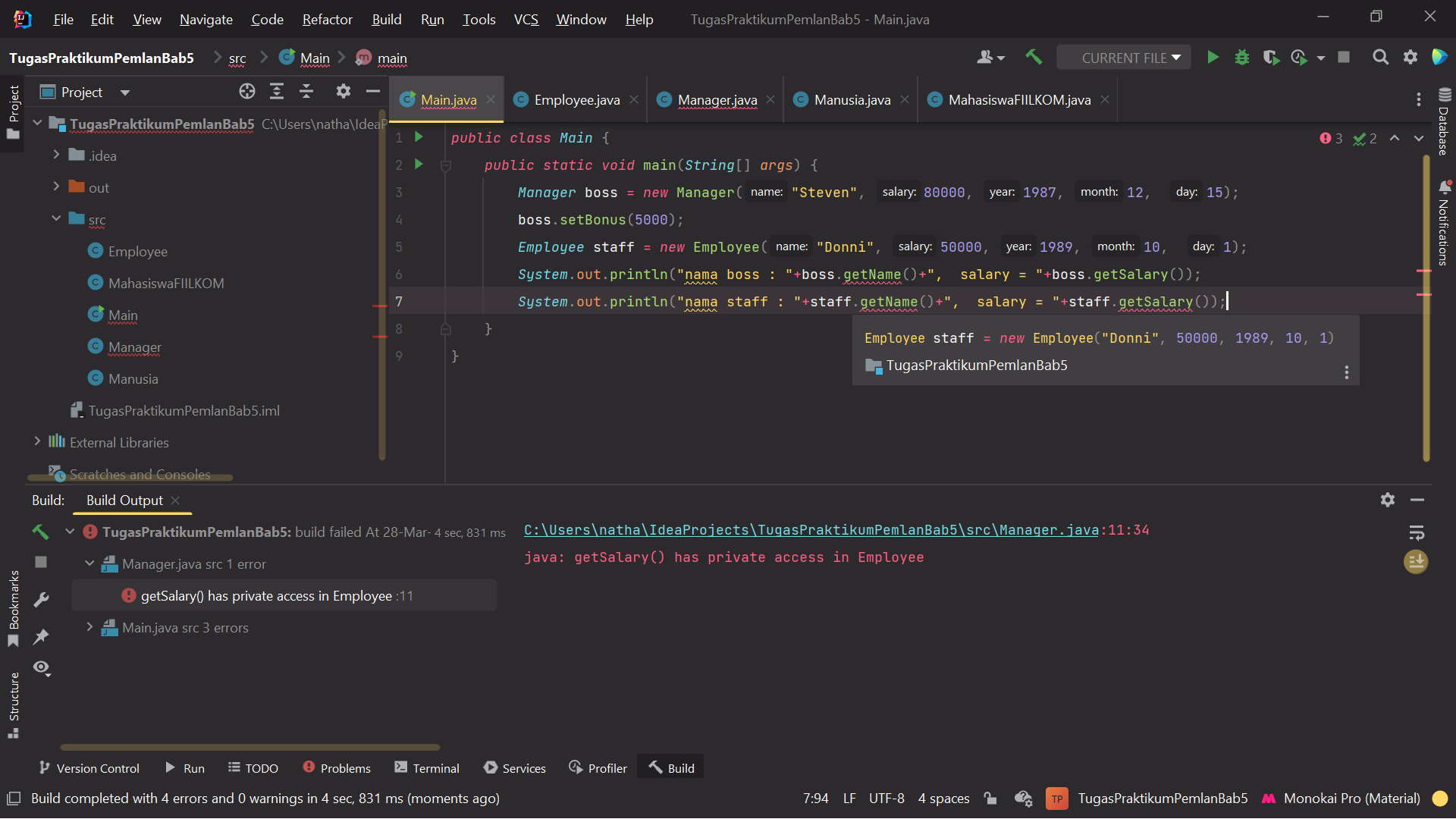


Modifier protected memungkinkan suatu method atau atribut dapat diakses pada suatu kelas dan turunan dari kelas tersebut atau pada kelas dalam package yang sama. Sedangkan private hanya menyediakan hak akses bagi method dan atribut pada kelas yang bersangkutan saja. Yang terakhir public, memungkinkan method dan atribut untuk diakses dari kelas mana saja. Jadi, pada superclass, sebaiknya suatu method atau atribut yang memiliki kesamaan dengan kelas turunannya menggunakan access modifier protected, agar hanya dapat diakses oleh kelas turunannya saja. Dan alasan mengapa access modifier public tidak cocok pada superclass adalah karena apabila atribut pada kelas yang vital diubah nilainya secara sengaja dapat mengakibatkan kesalahan perhitungan pada keseluruhan program. Sedangkan, pada access modifier private menyebabkan sebuah atribut atau method pada suatu kelas tidak dapat diakses oleh kelas turunannya.

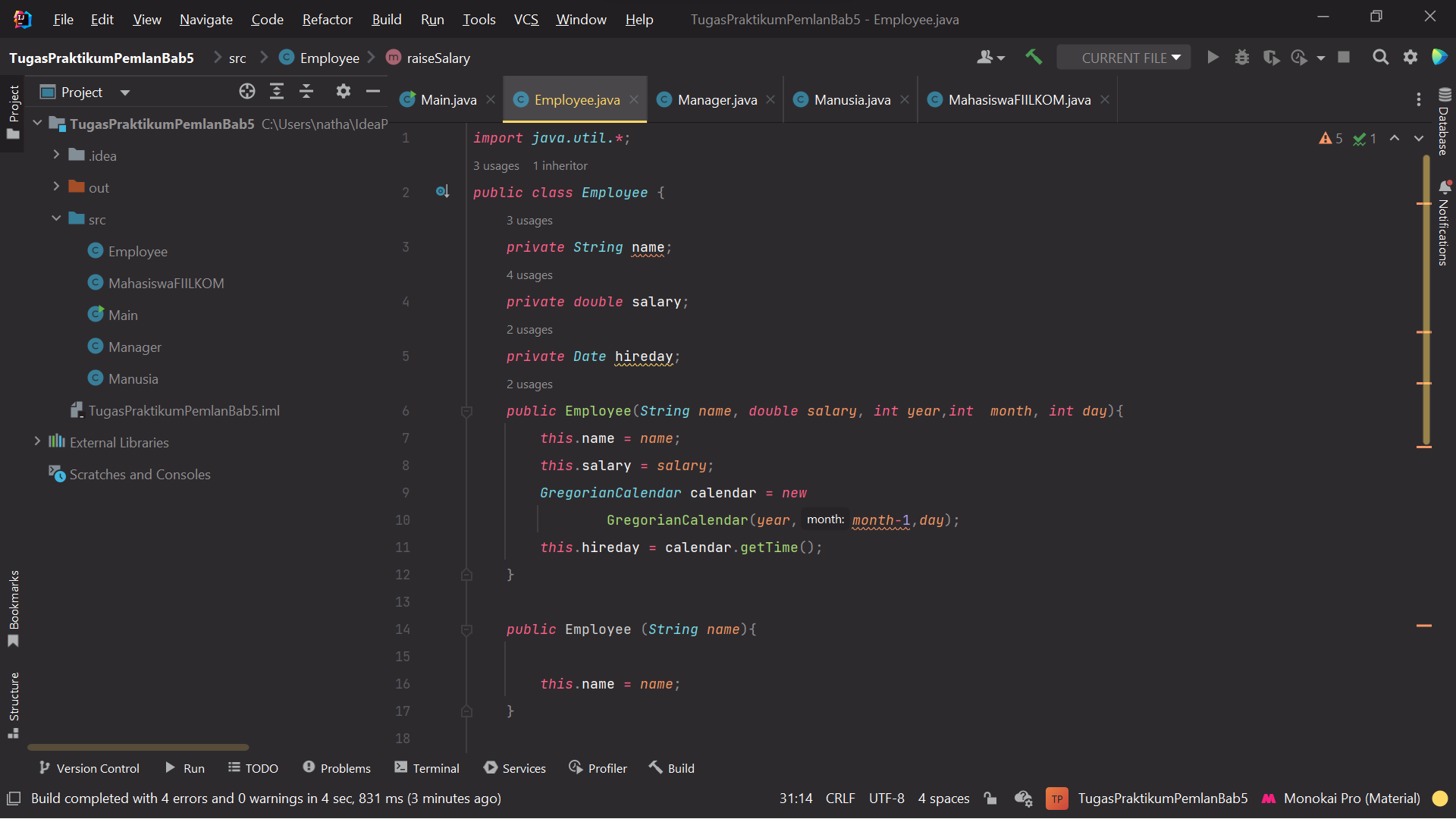


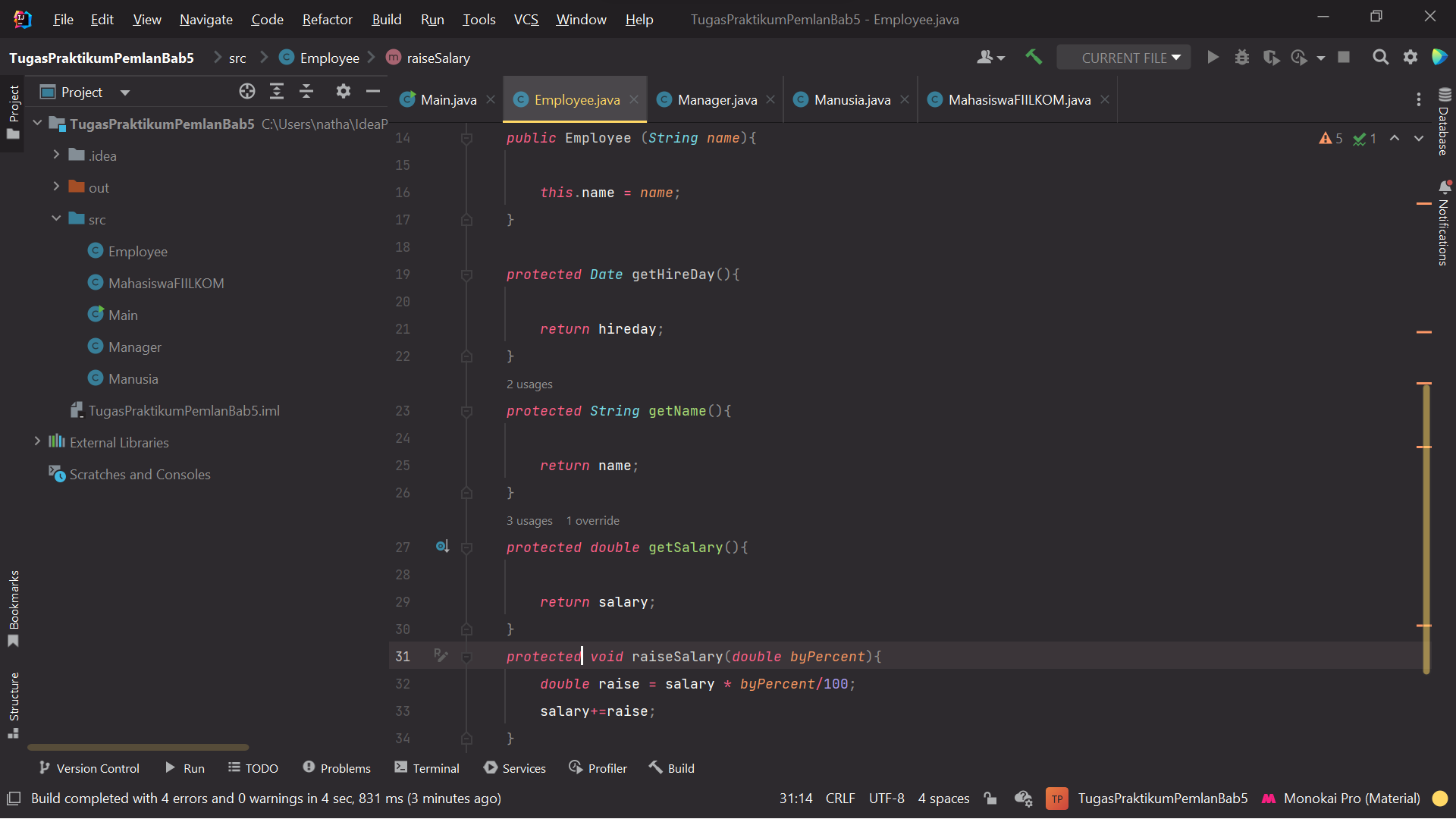


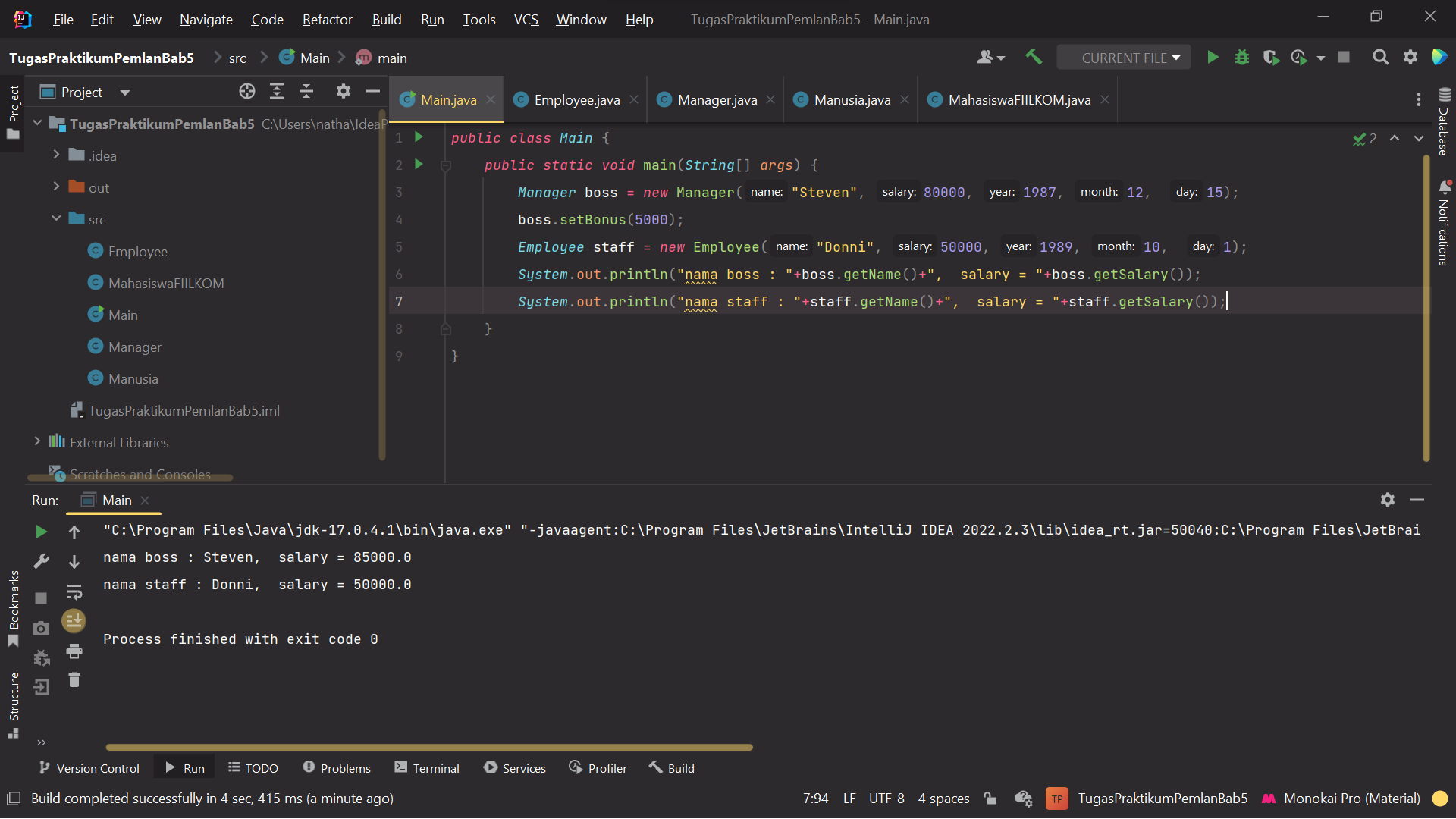




Apabila semua method di kelas Employee diubah menjadi private, maka method yang bersangkutan tidak dapat diakses dari kelas main maupun kelas manager yang merupakan kelas turunan dari kelas employee.







Pada access modifier protected tidak berpengaruh apa – apa, hal ini dikarenakan pada access modifier ini method yang masih berada dalam pakage yang sama tetap bisa diakses dari kelas lainnya. Oleh karena itu, method pada kelas employee dapat diakses pada kelas manager karena merupakan subclassnya dan pada kelas main yang masih berada pada package yang sama.

1. **Tugas Praktikum**
   1. **Source code**

|  |
| --- |
| package soal2;  public class Manusia {  private String nama, nik;  private boolean jenisKelamin, menikah;  public Manusia(String nama, String nik, boolean isLakiLaki, boolean menikah){  this.nama = nama;  this.nik = nik;  this.jenisKelamin=isLakiLaki;  this.menikah=menikah;  }  public void setNama(String nama){  this.nama = nama;  }  public void setNik(String nik){  this.nik = nik;  }  public void setJenisKelamin(boolean isLakiLaki){  this.jenisKelamin = isLakiLaki;  }  public void setMenikah(boolean menikah){  this.menikah = menikah;  }  public String getNama(){  return this.nama;  }  public String getNik(){  return this.nik;  }  public String getJenisKelamin(){  String kelamin;  if(this.jenisKelamin){  kelamin = "L";  }  else {  kelamin = "P";  }  return kelamin;  }  public boolean getMenikah(){  return this.menikah;  }  public double getTunjagan(){  if(menikah){  if (jenisKelamin){  return 25;  }  else{  return 20;  }  }  else {  return 15;  }  }  public double getPendapatan(){  return getTunjagan();  }  public String toString(){  return "Nama: " + this.nama +  "\nNIK: "+ this.nik+  "\nJenis Kelamin: "+getJenisKelamin()+  "\nPendapatan: "+getPendapatan();  }  }  package soal2;  public class MahasiswaFIILKOM extends Manusia{  private String nim;  private double ipk;  public MahasiswaFIILKOM(String nama, String nik, boolean isLakiLaki, boolean menikah, String nim, double ipk){  super(nama,nik,isLakiLaki,menikah);  this.nim = nim;  this.ipk = ipk;  }  public void setNim(String nim){  this.nim = nim;  }  public void setIpk(double ipk){  this.ipk = ipk;  }  public String getNim(){  return this.nim;  }  public double getIpk(){  return this.ipk;  }  public double getBeasiswa(){  if (3.0 <= this.ipk && this.ipk<=3.5){  return 50;  }  else if(3.5<=this.ipk && this.ipk<=4.0){  return 75;  } else {  return 0;  }  }  public String getStatus(){  String angkatan = "20" +this.nim.substring(0,2);  int temp = Integer.parseInt(this.nim.substring(6,7));  String prodi="";  switch (temp){  case 2:  prodi = "Teknik Informatika";  break;  case 3:  prodi = "Teknik Komputer";  break;  case 4:  prodi = "Sistem Informasi";  break;  case 6:  prodi="Pendidikan Teknologi Informasi";  break;  case 7:  prodi = "Teknologi Informasi";  break;  }  return prodi + ", "+angkatan;  }  @Override  public double getPendapatan(){  return super.getPendapatan()+this.getBeasiswa();  }  @Override  public String toString() {  return "Nama: " + getNama() +  "\nNIK: "+ getNik()+  "\nJenis Kelamin: "+super.getJenisKelamin()+  "\nPendapatan: "+this.getPendapatan() +  "\nNIM: "+getNim()  +"\nIPK: "+getIpk()+  "\nStatus: "+getStatus();  }  }  package soal2;  import java.time.LocalDate;  public class Pekerja extends Manusia {  private double gaji;  private LocalDate tahunMasuk;  private int jumlahAnak;  public Pekerja(String nama, String nik, boolean isLakiLaki, boolean menikah,double gaji, LocalDate tahunMasuk, int jumlahAnak) {  super(nama, nik, isLakiLaki, menikah);  this.gaji = gaji;  this.tahunMasuk=tahunMasuk;  this.jumlahAnak=jumlahAnak;  }  public void setGaji(double gaji) {  this.gaji = gaji;  }  public void setTahunMasuk(LocalDate tahunMasuk) {  this.tahunMasuk = tahunMasuk;  }  public void setJumlahAnak(int jumlahAnak) {  this.jumlahAnak = jumlahAnak;  }  public double getGaji() {  return gaji;  }  public int getJumlahAnak() {  return jumlahAnak;  }  public LocalDate getTahunMasuk() {  return tahunMasuk;  }  public double getBonus(){  int lamaBekerja = LocalDate.now().getYear() - tahunMasuk.getYear();  if (lamaBekerja >= 0 && lamaBekerja <= 5) {  return 0.05 \* gaji;  } else if (lamaBekerja > 5 && lamaBekerja <= 10) {  return 0.1 \* gaji;  } else {  return 0.15 \* gaji;  }  }  public double getTunjangan() {  return jumlahAnak \* 20;  }  @Override  public double getPendapatan(){  return super.getPendapatan()+getGaji()+getBonus()+getTunjangan();  }  @Override  public String toString() {  return "Nama: " + getNama() +  "\nNIK: "+ getNik()+  "\nJenis Kelamin: "+super.getJenisKelamin()+  "\nPendapatan: "+this.getPendapatan() +  "\nTahun Masuk: " + getTahunMasuk() +  "\nJumlah Anak: " + getJumlahAnak() +  "\nGaji: " + getGaji();  }  }  package soal2;  import java.time.LocalDate;  public class Manager extends Pekerja {  private String departemen;  public Manager(String nama, String nik, boolean isLakiLaki, boolean menikah, double gaji, LocalDate tahunMasuk, int jumlahAnak, String departemen) {  super(nama, nik, isLakiLaki, menikah, gaji, tahunMasuk, jumlahAnak);  this.departemen = departemen;  }  public void setDepartemen(String departemen) {  this.departemen = departemen;  }  public String getDepartemen() {  return departemen;  }  @Override  public double getTunjangan(){  return super.getTunjangan()+(0.1\*getGaji());  }  @Override  public double getPendapatan() {  return super.getPendapatan()+(0.1\*getGaji());  }  @Override  public String toString() {  return "Nama: " + getNama() +  "\nNIK: "+ getNik()+  "\nJenis Kelamin: "+super.getJenisKelamin()+  "\nPendapatan: "+this.getPendapatan() +  "\nTahun Masuk: " + getTahunMasuk() +  "\nJumlah Anak: " + getJumlahAnak() +  "\nGaji: " + getGaji()+  "\nDepartemen: "+getDepartemen();  }  }  package soal2;  import java.time.LocalDate;  public class MainManusia {  public static void main(String[] args) {  Manusia man1 = new Manusia("Adi","234345",true,true);  System.out.println(man1);  Manusia man2 = new Manusia("Lili","093854038549",false,true);  System.out.println(man2);  Manusia man3 = new Manusia("Joko","9834904",true,false);  System.out.println(man3);  MahasiswaFIILKOM mhs1 = new MahasiswaFIILKOM("nathan","312312312",true,false,"225150400111049",2.5);  System.out.println(mhs1);  MahasiswaFIILKOM mhs2 = new MahasiswaFIILKOM("nathan","312312312",true,false,"225150400111049",3.45);  System.out.println(mhs2);  MahasiswaFIILKOM mhs3 = new MahasiswaFIILKOM("nathan","312312312",true,false,"225150400111049",3.93);  System.out.println(mhs3);  Pekerja pkj1 = new Pekerja("Yusi","304958",true,true,5000, LocalDate.of(2021,2,21),2);  System.out.println(pkj1);  Pekerja pkj2 = new Pekerja("Bayu", "398457",true,false,8000,LocalDate.of(2014,3,17),0);  System.out.println(pkj2);  Pekerja pkj3 = new Pekerja("Lutfi","38495",true,true,8900,LocalDate.of(2003, 1,7),10);  System.out.println(pkj3);  Manager mnj1 = new Manager("Rudi","304958",true,true,7500,LocalDate.of(2008,4,2),1,"HRD");  System.out.println(mnj1);  }  } |

* 1. **Screenshot hasil**

|  |
| --- |
|  |