Laboratoire 05E – Les interfaces

CONTEXTE : La compagnie d’assurances *Les Assurances Sélectives* tente sa chance sur le marché des assurances automobiles. Pour ce faire, elle emploie une grille tarifaire simplifiée afin d’attirer des clients rapidement. Un futur assuré peut savoir s’il est éligible à cette assurance en ne fournissant qu’un minimum de détails sur lui et son véhicule ( auto ou motoneige ).

FOURNIS : Les classes Vehicule, Proprietaire, l’interface Assurable

**À FAIRE : Une classe** Automobile **représentant une automobile destinée à être assurée; elle doit donc mettre en œuvre l’interface Assurable.**

\*\*\* Une automobile n’est PAS assurable si :

* La personne a moins de 21 ans
* L’automobile est une taxi ( sa plaque débute par T, TR ou TS // *je sais que c’est répétitif, données de la SAAQ* )

\*\*\* Le coût de l’assurance-automobile est :

* La valeur de la constante POURCENT\_TAXE multipliée par la valeur de la voiture
* Si la ville habitée par le propriétaire est Montréal, on doit ajouter 6% du montant calculé à la première étape.

**À FAIRE : Une classe** Motoneige **représentant une motoneige destinée à être assurée; elle doit aussi mettre en œuvre l’interface Assurable.**

\*\*\* Une motoneige n’est pas assurable si :

* Si sa cylindrée est plus de 600 cc

\*\*\* Le coût de l’assurance pour la motoneige est :

* Calculé selon la formule suivante :

500 – ( age \*25 ) où l’âge est représenté par le nombre d’années entre l’année du modèle et l’année présente.

**À FAIRE : Tester avec le GUI** Test **si votre modèle fonctionne**

SOLUTIONS

Classe Propriétaire

|  |
| --- |
| package projetinterface; import java.time.\*; public class Proprietaire {   private String villeRes;  private LocalDate dateNaissance;   public Proprietaire(String villeRes, LocalDate dateNaissance)  {  this.villeRes = villeRes;  this.dateNaissance = dateNaissance;  }   public LocalDate getDate ()  {  return dateNaissance;  }  public String getVille()  {  return villeRes;  } } |

Classe Vehicule

|  |
| --- |
| package projetinterface;  public class Vehicule {  private double valeur;  private int anneeModele;  private Proprietaire pr;  private String numeroPlaque;   public Vehicule ( double valeur, int anneeModele, Proprietaire pr, String numeroPlaque )  {  this.valeur = valeur;  this.anneeModele = anneeModele;  this.pr = pr;  this.numeroPlaque = numeroPlaque;  }   public double getValeur()  {  return valeur;  }   public int getAnneeModele()  {  return anneeModele;  }   public Proprietaire getPr ()  {  return pr;  }   public String getNumeroPlaque()  {  return numeroPlaque;  } } |

Classe Automobile

|  |
| --- |
| package projetinterface; import java.time.\*; import java.util.\*; public class Automobile extends Vehicule implements Assurable {  public Automobile ( double valeur, int anneeModele, Proprietaire pr, String numeroPlaque )  {  super ( valeur, anneeModele, pr, numeroPlaque);  }    @Override  public boolean estAssurable ()  {  *// personne plus que 21 ans* LocalDate present = LocalDate.*now*();  Period intervalle = Period.*between*(getPr().getDate(), present);  int age = intervalle.getYears();  if ( age < 21 )  return false;  else if (getNumeroPlaque().charAt(0) =='T') *// n'assure pas les voitures taxis ( plaques commençant par T, TR, TS --> on peut réduire à T)* return false;  else  return true;  }   public double coutAssurance()  {  if ( estAssurable())  {  double cout = *POURCENT\_TAXE* \* getValeur() ;   if ( getPr().getVille().equals ( "Montréal"))  cout += cout \* 0.06;  return cout;  }  else  return 0;  } } |

Classe Motoneige

|  |
| --- |
| package projetinterface; import java.time.LocalDate; import java.time.Period; import java.util.\*; public class Motoneige extends Vehicule implements Assurable  {  private int cylindree ; *// 499, 599* public Motoneige ( double valeur, int anneeModele, Proprietaire pr, String numeroPlaque, int cylindree )  {  super ( valeur, anneeModele, pr, numeroPlaque);  this.cylindree = cylindree;  }   public int getCylindree ()  {  return cylindree;  }    public boolean estAssurable()  {  *// pas les cylindrees de plus de 600 cc* if ( cylindree > 600 )  return false;  else  return true;  }   public double coutAssurance()  {  if ( estAssurable() )  {  LocalDate present = LocalDate.*now*();  Period age = Period.*between*(getPr().getDate(), present);  GregorianCalendar now = new GregorianCalendar ();  return 500 - ( age.getYears() - getAnneeModele())\*25;  }  else  return -1;   }   public void setCylindree(int cylindree) {  this.cylindree = cylindree;  } } |

Classe Assurable

|  |
| --- |
| package projetinterface;  public interface Assurable {  public boolean estAssurable();  public double coutAssurance();  public static final double *POURCENT\_TAXE* = 0.025; } |