Exercice 1 :

package Exercice1;  
  
public class EntierNaturel {  
 private int val;  
  
 public int getVal() {  
 return val;  
 }  
  
 public void setVal(int val) {  
 this.val = val;  
 }  
  
 public EntierNaturel(int val) throws NombreNegatifException {  
 if (val < 0) {  
 throw new NombreNegatifException("La valeur ne peut pas être négative", val);  
 }  
 this.val = val;  
 }  
 public void decrementer() throws NombreNegatifException {  
 if (val == 0) {  
 throw new NombreNegatifException("La valeur est déjà nulle, impossible de décrémenter", val);  
 }  
 val--;  
 }  
}

package Exercice1;  
  
public class NombreNegatifException extends Exception {  
 private int valeurErronee;  
  
 public NombreNegatifException(String message, int valeurErronee) {  
 super(message);  
 this.valeurErronee = valeurErronee;  
 }  
 public int getValeurErronee() {  
 return valeurErronee;  
 }  
}

package Exercice1;  
  
public class Main {  
 public static void main(String[] args) {  
 try {  
 EntierNaturel entier = new EntierNaturel(-5);  
 } catch (NombreNegatifException e) {  
 System.*out*.println("Exception: " + e.getMessage());  
 System.*out*.println("Valeur erronée: " + e.getValeurErronee());  
 }  
  
 try {  
 EntierNaturel entier = new EntierNaturel(10);  
 System.*out*.println("Valeur de l'entier: " + entier.getVal());  
  
 entier.setVal(-3);  
 } catch (NombreNegatifException e) {  
 System.*out*.println("Exception: " + e.getMessage());  
 System.*out*.println("Valeur erronée: " + e.getValeurErronee());  
 }  
  
 try {  
 EntierNaturel entier = new EntierNaturel(1);  
 System.*out*.println("Valeur de l'entier: " + entier.getVal());  
  
 entier.decrementer();  
 entier.decrementer();  
 System.*out*.println("Valeur après décrémentation: " + entier.getVal());  
  
 entier.decrementer(); // Génère une exception  
 } catch (NombreNegatifException e) {  
 System.*out*.println("Exception: " + e.getMessage());  
 System.*out*.println("Valeur erronée: " + e.getValeurErronee());  
 }  
 }  
}

Exception: La valeur ne peut pas être négative

Valeur erronée: -5

Valeur de l'entier: 10

Valeur de l'entier: 1

Exception: La valeur est déjà nulle, impossible de décrémenter

Valeur erronée: 0

Process finished with exit code 0

Exercice 2

package Exercice2;  
  
public class CompteBancaire {  
 private String numeroCompte;  
 private double solde;  
 private String nomTitulaire;  
  
 public CompteBancaire(String numeroCompte, double solde, String nomTitulaire) {  
 this.numeroCompte = numeroCompte;  
 this.solde = solde;  
 this.nomTitulaire = nomTitulaire;  
 }  
  
 public String getNumeroCompte() {  
 return numeroCompte;  
 }  
  
 public void setNumeroCompte(String numeroCompte) {  
 this.numeroCompte = numeroCompte;  
 }  
  
 public double getSolde() {  
 return solde;  
 }  
  
 public void setSolde(double solde) {  
 this.solde = solde;  
 }  
  
 public String getNomTitulaire() {  
 return nomTitulaire;  
 }  
  
 public void setNomTitulaire(String nomTitulaire) {  
 this.nomTitulaire = nomTitulaire;  
 }  
  
 public void deposerArgent(double montant) {  
 solde += montant;  
 }  
  
 public void retirerArgent(double montant) throws FondsInsuffisantsException {  
 if (montant > solde) {  
 throw new FondsInsuffisantsException("Fonds insuffisants pour effectuer le retrait.");  
 }  
 solde -= montant;  
 }  
  
 public void afficherSolde() {  
 System.*out*.println("Solde du compte " + numeroCompte + ": " + solde);  
 }  
  
 public void transfererArgent(CompteBancaire compteDestinataire, double montant)  
 throws FondsInsuffisantsException, CompteInexistantException {  
 if (montant > solde) {  
 throw new FondsInsuffisantsException("Fonds insuffisants pour effectuer le transfert.");  
 }  
 if (compteDestinataire == null) {  
 throw new CompteInexistantException("Le compte destinataire n'existe pas.");  
 }  
 retirerArgent(montant);  
 compteDestinataire.deposerArgent(montant);  
 }  
}

package Exercice2;  
  
public class CompteCourant extends CompteBancaire {  
 private double decouvertAutorise;  
  
 public CompteCourant(String numeroCompte, double solde, String nomTitulaire, double decouvertAutorise) {  
 super(numeroCompte, solde, nomTitulaire);  
 this.decouvertAutorise = decouvertAutorise;  
 }  
  
 @Override  
 public void retirerArgent(double montant) throws FondsInsuffisantsException {  
 if (montant > getSolde() + decouvertAutorise) {  
 throw new FondsInsuffisantsException("Fonds insuffisants pour effectuer le retrait.");  
 }  
 super.retirerArgent(montant);  
 }  
}

package Exercice2;  
  
public class CompteEpargne extends CompteBancaire {  
 private double tauxInteret;  
  
 public CompteEpargne(String numeroCompte, double solde, String nomTitulaire, double tauxInteret) {  
 super(numeroCompte, solde, nomTitulaire);  
 this.tauxInteret = tauxInteret;  
 }  
  
 public void calculerInterets() {  
 double interets = getSolde() \* tauxInteret;  
 deposerArgent(interets);  
 System.*out*.println("Intérêts ajoutés au compte " + getNumeroCompte() + ": " + interets);  
 }  
}

package Exercice2;  
  
public class CompteInexistantException extends Exception {  
 public CompteInexistantException(String message) {  
 super(message);  
 }  
}

package Exercice2;  
  
public class FondsInsuffisantsException extends Exception {  
 public FondsInsuffisantsException(String message) {  
 super(message);  
 }  
}

package Exercice2;  
  
import java.util.ArrayList;  
  
public class Main {  
 public static void main(String[] args) {  
 ArrayList<CompteBancaire> listeComptes = new ArrayList<>();  
  
  
 listeComptes.add(new CompteCourant("123456", 100, "Client 1", 400));  
 listeComptes.add(new CompteEpargne("789012", 16000, "Client 1", 0.05));  
 listeComptes.add(new CompteCourant("8569455", 502000, "Client 2", 500));  
 listeComptes.add(new CompteEpargne("736992", 256000, "Client 2", 0.05));  
  
 listeComptes.remove(2);  
  
  
 try {  
 CompteBancaire compte1 = listeComptes.get(0);  
 CompteBancaire compte2 = listeComptes.get(1);  
  
 compte1.afficherSolde();  
 compte2.afficherSolde();  
  
 compte1.retirerArgent(200);  
 } catch (FondsInsuffisantsException e) {  
 System.*out*.println("Erreur: " + e.getMessage());  
 }  
  
 try {  
 CompteBancaire compte2 = listeComptes.get(1);  
 CompteBancaire compte3 = null;   
  
 compte2.transfererArgent(compte3, 300);  
 } catch (FondsInsuffisantsException | CompteInexistantException e) {  
 System.*out*.println("Erreur: " + e.getMessage());  
 }  
 }  
}

Solde du compte 123456: 100.0

Solde du compte 789012: 16000.0

Erreur: Fonds insuffisants pour effectuer le retrait.

Erreur: Le compte destinataire n'existe pas.

Process finished with exit code 0