Rapport du tp1

Exercice 1 :

Pour le 1er exercice j'ai créé trois Classes Java :

EntNet : avec un constructeur qui lance une exception de type ErrConst si la valeur en paramètre n'est pas positif.

```java
package Ex1;
import java.util.Scanner;

public class EntNet {
    private int n;
    public EntNet(int n) throws ErrConst {
        if (n<0) throw new ErrConst();
        this.n = n;
    }
    public int getN() {
        return this.n;
    }

}
```

ErrConst : une classe qui étend la classe Exception, son but c'est que après chaque appel, elle lance un message d'erreur, et interrompe l'exécution.

```java
package Ex1;

public class ErrConst extends Exception{
    public ErrConst(){
        System.out.println("la valeur est négative");
        System.exit(-1);
    };
}
```

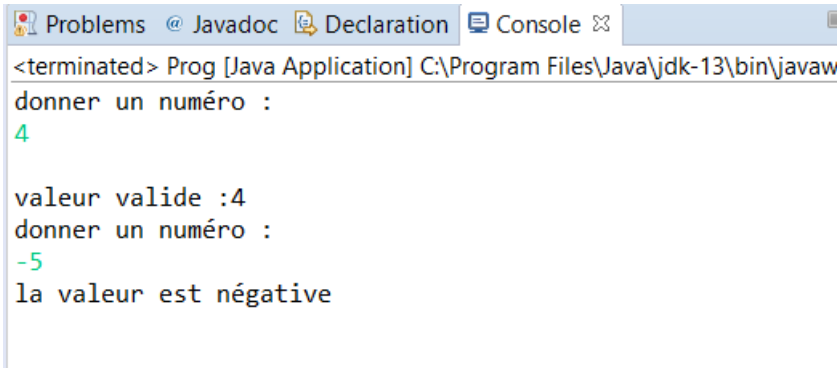
Prog : c'est la class du teste, dont on demande au utilisateur de saisir des valeurs, pour tester si ces valeurs convient(par le biais de l'exception ErrConst).

```java
package Ex1;

import java.util.Scanner;

public class Prog {
    public static void main(String[] args) throws ErrConst {
        Scanner sc = new Scanner(System.in);
        for(int i=1; i<3;i++) {
            System.out.println("donner un numéro :");
            int number = sc.nextInt();
            EntNet e1 = new EntNet(number);
            System.out.println("valeur valide :"+e1.getN());
        }
    }
}
```

Exécution :



Exercice 2 :

Dans le 2ème exercice j'ai utilisé ces techniques :

- Le gestionnaire des exceptions.
- La transmission de quelques informations au gestionnaire.

Class EntNet

```
package Ex2;
public class EntNet {
    private int n;
    public EntNet(int n) throws ErrConst{
        if (n<0) throw new ErrConst(n);
        this.n = n;
    }
    public static int Somme(int a, int b) throws ErrNat{
        //System.out.println(Integer.MAX_VALUE);
        int som = a+b;
        if (som > Integer.MAX_VALUE || som< 0) throw new ErrSom("somme non representable",a,b);
        return som;
    }
    public static int Difference(int a, int b) throws ErrNat{
        int diff = a-b;
        if (diff > Integer.MAX_VALUE || diff< 0) throw new ErrDiff("différence non representable", a,b);
            return diff;
    }
    public static int Produit(int a, int b) throws ErrNat{
        int p = a*b;
        if (p > Integer.MAX_VALUE || p< 0) throw new ErrProd("produit non représentable",a,b);
        //if (p > Integer.MAX_VALUE ) throw new ErrDepasse("limite dépassée",a,b);
        return p;
    }
    public int getN() {
        return this.n;
    }
}
```

Class ErrConst

```java
package Ex2;

public class ErrConst extends Exception{
    int n;
    public ErrConst(int n) {
        this.n = n;
    };
}
```

Class ErrNat

```java
package Ex2;

public class ErrNat  extends Exception {
    int a;
    int b;
    String message;
    public ErrNat( String message, int a, int b) {
        this.message = (message);
        this.a = a;
        this.b = b;
    }
}
```

Class ErrSom

```java
package Ex2;

public class ErrSom extends ErrNat {
    public ErrSom(String message,int a, int b) {
        super(message,a,b);
    }
}
```

Class ErrDiff

```java
package Ex2;

public class ErrDiff extends ErrNat {
    public ErrDiff(String message,int a, int b) {
        super(message,a,b);
    }
}
```

Class ErrProd

```java
package Ex2;

public class ErrProd extends ErrNat {
    public ErrProd(String message,int a, int b) {
        super(message,a,b);
    }
}
```

Class Programme

Le 1er TRY explicite la nature de l'exception en affichant les informations disponibles :

```java
package Ex2;
import java.util.Scanner;

public class Programme {
    public static void main(String[] args){
        Scanner sc = new Scanner(System.in);

        System.out.println("Donner une valeur");
        int valeur = sc.nextInt();
        try{
            EntNet e = new EntNet(valeur);
            System.out.println("Donner deux valeur pour effectuer différentes opérations :");
            System.out.println("Valeur 1:");
            int a = sc.nextInt();
            System.out.println("Valeur 2:");
            int b = sc.nextInt();
            System.out.println("Somme: "+e.Somme(a,b));
            System.out.println("différence: "+e.Difference(a,b));
            System.out.println("Produit: "+e.Produit(a,b));
        }
        catch(ErrConst ec) {
            System.out.println("erreur déclenché par la valeur("+ec.n+"): valeur négative !!\"");
        }catch(ErrNat en) {
            //e.printStackTrace();
            System.out.println("erreur déclenché par les valeurs("+en.a+" et "+en.b+"):"+en.message);
        }
```

Le 2ème TRY :

```java
        try{
            EntNet e = new EntNet(valeur);
            System.out.println("\n \n Donner deux valeur pour effectuer différentes opérations :");
            System.out.println("Valeur 1:");
            int a = sc.nextInt();
            System.out.println("Valeur 2:");
            int b = sc.nextInt();
            System.out.println("Somme: "+e.Somme(a,b));
            System.out.println("différence: "+e.Difference(a,b));
            System.out.println("Produit: "+e.Produit(a,b));
        }
        catch(ErrConst ec) {
            ec.printStackTrace();
            System.out.println(ec);
        }catch(ErrNat en) {
            //e.printStackTrace();
            System.out.println(en);
        }
    }
}
```

Exécution :

```
<terminated> Programme [Java Application] C:\Program Files\Java\jdk-13\bin\javaw.exe (9 fév.
Donner une valeur
9

Donner deux valeur pour effectuer différentes opérations :
Valeur 1:
12

Valeur 2:
-15

erreur déclenché par les valeurs(12 et -15):somme non representable


 Donner deux valeur pour effectuer différentes opérations :
Valeur 1:
12

Valeur 2:
-15
Ex2.ErrSom
```