Université Paris 7 - Denis Diderot
L1 Sciences

IF2 : Structures de données et Objets JAVA
Année 2009-2010, 2ème semestre

TD n°3

Exercice 1 Fonction factorielle.

Une représentation itérative de la fonction factorielle peut être définie de la manière suivante :

```
public static int factorielle(int n){

    int i, fact;

    i=0;
    fact=1;

    while(i<n){

         i=i+1;
         fact=fact*i;

    }
    return fact;
}
```

1. Donnez une précondition *Pre* et une postcondition *Post* du programme.
2. Trouvez un invariant **I** pour la boucle.
3. Comment vérifier que **I** est un invariant de la boucle ? (Faire le lien avec *Pre* et *Post*).
4. Le programme termine t-il ? Justifiez votre réponse en donnant une fonction $f(\mathbf{X})$ qui décroît vers une valeur constante (zéro par exemple) où $\mathbf{X}$ représente l'ensemble des variables mises en jeu et modifiées dans le corps de la boucle.
5. Proposez une forme de représenter la fonction factorielle récursive.

Exercice 2 Division entière.

Rappelons tout d'abord, la définition de la division entière de deux entiers naturels `a` et `b` :

$$\forall a \geq 0 \ , \ b > 0 \ , \ a = quotient \cdot b + reste \ , \ 0 \leq reste < b$$

Pour calculer la division entière, nous allons procéder par soustractions successives de la valeur `b` de `a` jusqu'à ce que le résultat de la soustraction soit inférieur à `b`.

1. Ecrivez la méthode `divisionEntier()` basée sur cet algorithme (elle est construite autour d'une boucle `while`).

2. Exhibez un invariant pour la boucle `while` permettant de justifier la correction du programme.

3. Démontrez la finitude de la boucle.

Exercice 3 Algorithme d'Euclide.

Le but de l'exercice est d'écrire un programme qui calcule le pgcd (plus grand commun diviseur) de deux nombres entiers positifs (non tous nuls), à l'aide de l'algorithme d'Euclide. Celui-ci est basé sur la propriété suivante : si a et $b \neq 0$ sont deux entiers positifs, et si l'on note r_1 le reste de la division euclidienne de a par b, alors $pgcd(a, b) = pgcd(b, r_1)$. L'idée de l'algorithme est de recommencer la même opération sur le couple $(b, r_1$

```
            b=b-1;
        }
        return prod;

}
```

1. Donnez une précondition et une postcondition de la boucle externe.
2. Exhibez les deux invariants pour les boucles `while` permettant de justifier la correction du programme.
3. Démontrer la finitude des deux boucles.