

Examen de Algorithmique et Struct. de Données

Durée : 0h30

Aucun document autorisé

Mobiles interdits

Nom :

Prénom :

-
- 1. Expliquez ce que calcule la fonction récursive `mystère` suivante ?

```
private String mystère(int n, int cand) {
    if (n==1) return "1";
    if (n%cand == 0)
        return cand + "x" + mystère(n/cand, cand);
    return mystère(n, cand+1);
}

public String mystère(int n) {
    assert n>0;
    return mystère(n, 2);
}
```

et qu'affiche le code suivant ?

```
int n = 24;
System.out.println(n + " = " + mystère(n));
```

```
/*
 * Antécédent : n>0 et cand>=2
 */
String mystère(int n, int cand) {
    assert cand>=2;
    if (n==1) return "1";
    if (n%cand == 0)
        return cand + "x" + mystère(n/cand, cand);
    return mystère(n, cand+1);
}

/**
 * Antécédent : n>0
 * Rôle : renvoie la décomposition de n en facteurs premiers
 */
String facteursPremiersR(int n) {
    assert n>0;
    return mystère(n, 2);
}
```

Cette fonction affiche la décomposition d'un nombre entier ses facteurs premiers. L'exécution du code précédent affiche sur l'entrée standard :

```
24 =2 x 2 x 2 x 3 x 1
```

Notez que l'antécédent de cette fonction est : $n > 0$.

-
- 2. Écrivez une version itérative de la fonction précédente.
-

```
/**
 * Ancédédent : n>0
 * Rôle : renvoie la décomposition de n en facteurs premiers
 */
String facteursPremiers(int n) {
    String res="";
    int nbPremier=2;
    while (n>1) {
        if (n%nbPremier == 0) {
            //nbPremier est un facteur premier
            res+= nbPremier + "x";
            n/=nbPremier;
        }
        else
            nbPremier++;
    }
    //nbPremier == 1;
    return res + "1";
}
```

.....