Génie Logiciel Avancé

Année 2018-2019

Prof. Burkhart Wolff
wolff@lri.fr

T. Balabonski, D. Gallois-Wong, H. Brabra
blsk@lri.fr, diane.gallois-wong@lri.fr,
hayet.brabra@telecom-sudparis.eu

TD bonus - Révisions test et preuve

Semaine du 4 décembre 2018

Considérons le programme `tri_insertion` suivant, qui trie un tableau `t` passé en argument selon un algorithme de tri par insertion. Le programme utilise une fonction annexe `permutation` dont le code ne nous est pas fourni : cette fonction sera considérée comme une boîte noire, dont le comportement ne pourra être décrit que par une spécification logique.

Tri par insertion

```
void permutation(int t[], int i, int j) {
  // Implémentation masquée
}

void tri_insertion(int t[], int n) {
  int i;
  for(i=1; i<n; i++) {
    int j = i;
    while(j>0 && t[i] < t[j-1]) {
      j--;
    }
    permutation(t,j,i);
  }
}
```

Question 1 : spécification. Écrire des spécifications pour les fonctions `tri_insertion` et `permutation`. La première doit trier le tableau, tandis que la seconde doit permettre à la première de fonctionner.

Question 2 : test en boîte noire. Proposer un jeu de tests pour la fonction `permutation`.

Question 3 : test en boîte blanche. Dessiner le graphe de flot de contrôle de la fonction `tri_insertion`. Dresser la liste des chemins passant trois fois dans la boucle externe. Proposer un jeu de tests pour cette fonction, qui doit couvrir le code et les décisions.

Question 4 : preuve. Donner les invariants des boucles interne et externe de la fonction `tri_insertion`, et prouver sa correction.