TP 1 Programmation – DUT 1 – Premières procédures

P.Courtieu

Septembre 2017

1 Annotation, variants et invariants de boucle

Donnez des triplets de Hoare corrects pour les programmes suivants. On s'autorise à utiliser des constantes (a, b, c,...) pour désigner les valeurs initiales des variables. Lorsqu'un programme contient une ou plusieurs boucles, donnez pour chacune un invariant et un variant suffisant pour prouver le triplet. Il n'est pas demandé d'écrire les déductions (autrement dit : pas de règle d'inférence).

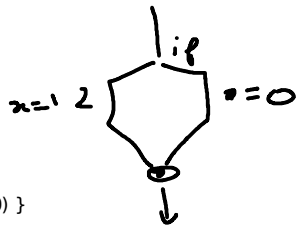
1.
```
/* requires: true                x<3              x=x0                    */
x = x+1;
/* ensures: x=x0+1               x<4              x=x0+1                  */
```

2.
```
/* requires: y = 22              y = 33           y = x0          y=x0    */
x = y;
/* ensures: {x = 22 && y = 22}   {x = 33 && y = 33}  {x = x0 && y = x0 }  {x=y && y=x0}  */
```

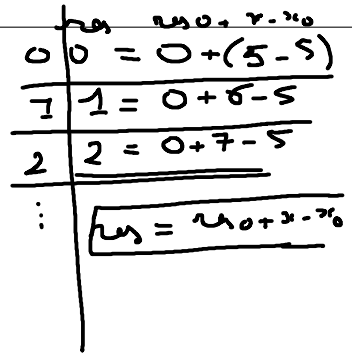
3.
```
/* requires:   { x = 5 && i > 0}                                          */
x = x + i;
x = 3;
/* ensures:    { x = 3 && i > 0 }                                         */
```

4.
```
/* requires:  { x = 0 }   { x = 10 }   {x=20}   { x > 0 && x = x0}        */
x = x + 3;
if (x>12) {
  x = 12;
}                                                  {   ((x0+3>12) -> x = 12)
                                                     && ((x0+3<=12) -> x = x0+3) }
/* ensures:   {x=3}       { x = 12 }   {x=12}                             */
```

5.
```
/* requires:                { x > 0 && x = x0}                            */
x = x + 3;
if (x>12) {
  x = 12;
} else {
  x = 0;
}                          {   ((x0+3>12) -> x = 12)
                             && ((x0+3<=12) -> x = 0) }
/* ensures:                                                               */
```



6.
```
/* requires: {x = 5 && i = 7 && res =0}                                   */
while(x<i){ /*   variant:                                                 */
            /* invariant:  i=7 && x<=i && res = res0 + (x-x0)              */
  res = res + 1;                                  res-res0=x-x0
  x = x+1;
}
/* ensures:  { res = i - 5 && i = 7 && x=7}                               */
```



```
}
/* ensures:                                                    */
```

7.
```
/* requires:                                                   */
while(x<i){
  res = res + x;
  x = x+1;
}
/* ensures:                                                    */
```

8.
```
/* requires:                                                   */
while(x>i){ /*   variant:                                     */
            /* invariant:                                     */
  res = res + i;
  x = x-1;
}
/* ensures:                                                    */
```

9.
```
/* requires:                                                   */
while(true){ /*   variant:                                    */
             /* invariant:                                    */
  res = res + i;
  x = x-1;
}
/* ensures:                                                    */
```

10.
```
/* requires:                                                   */
while(false){ /*   variant:                                   */
              /* invariant:                                   */
  res = res + i;
  x = x-1;
}
/* ensures:                                                    */
```

11.
```
/* requires:                                                   */
i=0;
while(i < 12){ /*   variant:                                  */
               /* invariant:                                  */
  t[i] = t[i]-1;
  i = i+1;
}
/* ensures:                                                    */
```

12.
```
/* requires:                                                   */
i=11;
while(i >= 0){ /*   variant:                                  */
               /* invariant:                                  */
  t[i] = t[i]-1;
  i = i - 1;
}
/* ensures:                                                    */
```

2 Algorithmes sur les tableaux

1. Écrivez et annotez un algorithme de recherche d'une valeur entière dans un tableau. une variable doit contenir le numéro de la case contenant la valeur recherchée si elle existe, et -1 sinon.
2. Même question pour une recherche optimisée dans laquelle on suppose le tableaux trié par ordre croissant (recherche par dichotomie).
3. Même question pour un algorithme de tri par insertion (on cherche la plus petite et on l'échange avec la case 1, puis on recommence à partir de la case 2, etc.)
4. Même question pour la recherche du deuxième plus grand élément dans un tableau.

3 Algorithmes sur les tableaux

L'algorithme de Dijkstra est un algorithme célèbre de recherche de plus court chemin dans un graphe (pensez à la recherche d'itinéraire dans le réseau routier...). Il est bien expliqué ici :
https://haltode.fr/algo/structure/graphe/plus_court_chemin/dijkstra.html.

Annotez le pseudo-code donné dans cette page web.