Réseaux et sécurité

Exercices – 03

Frédéric Loulergue

Université d'Orléans
Laboratoire d'Informatique Fondamentale d'Orléans

Fall 2022

Exercise 1

Give the loop annotations so that the contract can be verified:

```
/*@ requires \valid(a + (0.. size -1));
    assigns \nothing;
    behavior all_negative:
      assumes \forall integer k; 0 <= k < size ==> a[k] <= 0;
      ensures \result == true;
    behavior one_positive:
      assumes \exists integer k; 0 <= k < size && a[k] > 0;
      ensures \result == false;
    disjoint behaviors;
    complete behaviors; */
bool is_negative(int * a, size_t size)
{
  for(size_t i = 0; i < size; i++)
    if (a[i] > 0)
      return false;
  return true;
}
```

Exercise 2

Give the loop annotations so that the contract can be verified:

```
/*@ requires 0 <= size  && \valid(a +  (0.. size −1));
    assigns  \nothing;
    behavior all_negative :
      assumes \forall  integer  k;  0 <= k <  size  ==> a[k]  <= 0;
      ensures \result  == true;
    behavior one_positive:
      assumes \exists integer  k;  0 <= k <  size  && a[k] > 0;
      ensures \result  == false ;
     disjoint  behaviors;
    complete behaviors; */
bool is_negative(int  * a,  int  size)
{
  for(int  i  = size  − 1;  0 <= i ;  i−−)
    if  (a[i]  > 0)
      return  false ;
  return true;
}
```

Exercise 3

Write a correct and complete functional specification that avoids runtime errors and loop annotations to verify it.

```
#include <limits.h>
#include <stddef.h>
#include <stdint.h>

void concat(int * t1, size_t size1, int * t2, size_t size2, int * dst)
{
  for(size_t i = 0; i < size1 + size2; i++)
    if (i < size1)
      dst[i] = t1[i];
    else
      dst[i] = t2[i - size1];
}
```

Exercise 4

Loop Variant

Prove the termination for the functions of previous exercises.