Réseaux et sécurité

Exercices – 02

Frédéric Loulergue

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

Fall 2022

Exercise 1

Contract

Give a correct and complete functional specification using only ==> as a logical connective.

```
int max(int a, int b)
{
  if (a < b)
    return b;
  else
    return a;
}
```

Exercise 2

Contract

Write a correct and complete functional specification that avoids runtime errors.

```
void swap(int * a, int * b)
{
  int tmp = *a;
  *a = *b;
  *b = tmp;
}
```

Exercise 3

Contract

Write a correct and complete functional specification that avoids runtime errors.

```
void increment(int * value, int step)
{
  *value += step;
}
```

Exercise 4

Contract

Write a correct and complete functional specification that avoids runtime errors.

```
#include <stddef.h>

int is_null(void * p)
{
  return p == NULL;
}
```

Exercise 5

\old

Assuming a variable x has type **int** *.
What is the difference – in a postcondition – between:

- \old(*a) and
- *\old(a)?

Exercise 6

Contract

Write a correct and complete functional specification that avoids runtime errors.

```
int is_positive(int * a, int size)
{
  /*@ loop invariant 0 <= i <= size;
    @ loop invariant \forall integer k; 0 <= k < i ==> 0 <= a[k];
    @ loop assigns i;
    @ loop variant size - i; */
  for(int i = 0; i < size; i++)
    if (a[i] < 0)
      return 0;
  return 1;
}
```

Exercise 7

Contract

Write a correct and complete functional specification that avoids runtime errors.

```c
void is_positive(int * a, int size, int * r)
{
  int i;
  *r = 1;
  /*@ loop invariant 0 <= i <= size;
    @ loop invariant *r == 1 <==> \forall integer k; 0 <= k <i ==> 0 <= a[k];
    @ loop assigns i, *r;
    @ loop variant size - i; */
  for(i = 0; i < size; i++)
    if (a[i] < 0)
      *r = 0;
}
```

Exercise 8

Contract

Write a correct and complete functional specification that avoids runtime errors without using \forall .

```
int is_positive(int * a, int size)
{
  /*@ loop invariant 0 <= i <= size;
    @ loop invariant \forall integer k; 0 <= k < i ==> 0 <= a[k];
    @ loop assigns i;
    @ loop variant size - i; */
  for(int i = 0; i < size; i++)
    if (a[i] < 0)
      return 0;
  return 1;
}
```

Exercise 9

Contract

Write a correct and complete functional specification that avoids runtime errors, using **behaviors**.

```
int is_positive(int * a, int size)
{
  /*@ loop invariant 0 <= i <= size;
    @ loop invariant \forall integer k; 0 <= k < i ==> 0 <= a[k];
    @ loop assigns i;
    @ loop variant size - i; */
  for(int i = 0; i < size; i++)
    if (a[i] < 0)
      return 0;
  return 1;
}
```