Practica 2

Autor: Carlos Giudice

Ejercicio 1

DUDA: como pido que A esté inicializado? Está bien pedir que sea distinto de null?

precondición: $A \neq Null \land B \neq Null \land B.length \geq A.length$

Ejercicio 2

```
void test2() {
    int rv = foo(0, 1);
    assertEquals(0, rv);
}
```

Figure 1: Pseudocode

Ejercicio 3

```
void test3() {
    int rv = foo(5, 3);
    assertEquals(3, rv);
}
```

Figure 2: Pseudocode

Ejercicio 4

Si reemplazamos cualquiera de las desigualdades del primer if por el signo “>” (lo cual llamamos relational operator replacement) y escribimos un test que checkea la correcta detección de un triángulo, veremos que el mutante devuelve el número 4. Esto es una respuesta incorrecta.

Ejercicio 5

root	N0.left	N0.right	N1.left	N1.right	N2.left	N2.right
N0	NULL	N1	N2	NULL	NULL	NULL
N0	N1	NULL	NULL	NULL	NULL	NULL
N0	N1	NULL	NULL	N2	NULL	NULL

Ejercicio 6

```
bool rep_ok(DLLlist input){

    // empty list
    if(input.size == 0 &&
       input.first == NULL &&
       input.last == NULL) {
        return true;
    }

    if(input.first.previous != NULL) return false; // first node should not have a previous node
    if(input.last.next != NULL) return false; // last node should not have a next node

    Node curr = input.first;
    Node prev = NULL;
    for(int i = 0; i < input.size; i++){
        if(curr == NULL) return false; // ensure that there are at least as many nodes as the size field says

        if(prev != NULL) {
            if(prev != curr.previous) return false; // is pointer to previous node broken?
        }

        prev = curr
        curr = curr.next
    }

    if(curr != input.last) return false; // mismatch between last nodes

    return true;
}
```

Figure 3: Pseudocode

Ejercicio 7

1. bt = BinaryTree(NULL);
2. bt = BinaryTree(NULL);
   bt.removeRoot();
3. n1 = Node(NULL, NULL);
   n2 = Node(n1, n1);
   bt = BinaryTree(NULL);

- Las secuencias dos y tres seran descartadas por arrojar errores en su ejecución.
- La secuencia uno puede o no ser útil para evidenciar algún error en el código a testear.
  - `En caso afirmativo será enviada al usuario como información para arreglar el código.`
  - `Si no rompe el código, será enviada al conjunto de componentes donde será un posible elemento a usar a`

Ejercicio 8

```
// loop unroll
// NOTA: en vez de usar break mando el resto del codigo al else del if
int test_me(int x) {
    int [] A = {5, 7, 9};
    int i = 0;
    if(i < 3){
        if(A[i] == x){}
        else {
            i++;
            if(i < 3){
                if(A[i] == x){}
                else {
                    i++;
                    if(i < 3){
                        if(A[i] == x){}
                        else{
                            i++;
                        }
                    }
                }
            }
        }
    }
    return i;
}
```

Figure 4: Pseudocode

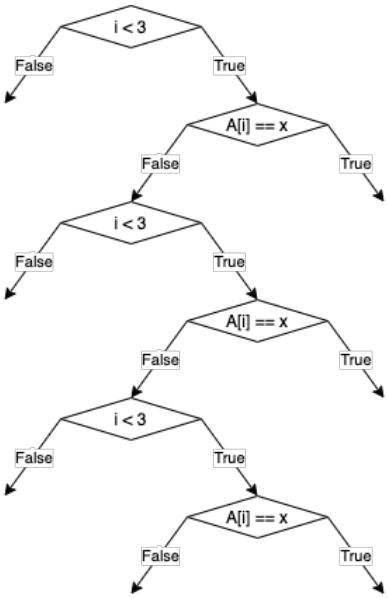

Figure 5: Computation tree

x	cond de ruta enviada
1	$5 \neq x \wedge 7 \neq x \wedge 9 = x$
9	$5 \neq x \wedge 7 = x$
7	$5 \neq x$

Ejercicio 9

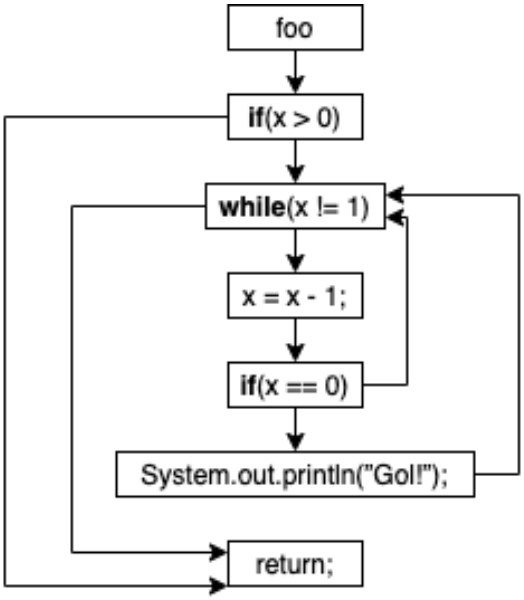

Figure 6: Control flow graph

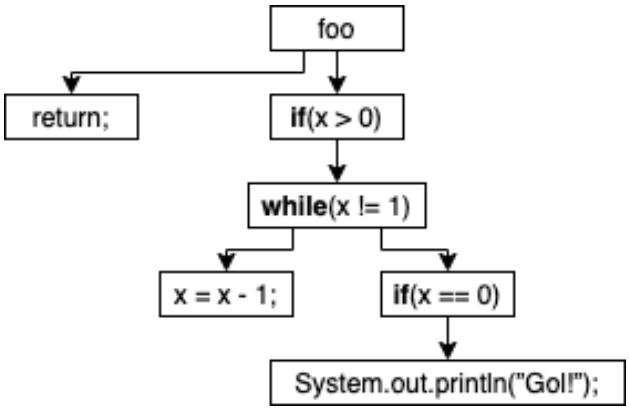

Figure 7: Control dependency graph

- Asumiendo un valor de K=5, la distancia del branch L1 para el input x = -15 es:
  - $L1 - true = a > b \ ? \ 0 : (b - a) + K = -15 > 0 \ ? \ 0 : (0 - (-15)) + 5 = (0 - (-15)) + 5 = 20$
  - $L1 - false = a \leq b \ ? \ 0 : (a - b) = -15 \leq 0 \ ? \ 0 : -15 - 0 = 0$
- Asumiendo un valor de K=5, la distancia del branch L4 para el input x = -10 es:
  - $L4-true = abs(-10 - 0) = 10
  - $L4-false = a != b \ ? \ 0 : K = -10 != 0 \ ? \ 0 : 5 = 0