- MOVE ( )
- TURNLEFT ( )
- TURNRIGHT ( )
- INFECT ( )
- SKIP ( )
- HALT ( )
- JUMP ( )
- JUMP_IF_NOT_NEXT_IS_EMPTY ( )
- JUMP_IF_NOT_NEXT_IS_NOT_EMPTY ( )
- JUMP_IF_NOT_NEXT_IS_WALL ( )
- JUMP_IF_NOT_NEXT_IS_NOT_WALL ( )
- JUMP_IF_NOT_NEXT_IS_FRIEND ( )
- JUMP_IF_NOT_NEXT_IS_NOT_FRIEND ( )
- JUMP_IF_NOT_NEXT_IS_ENEMY ( )
- JUMP_IF_NOT_NEXT_IS_NOT_ENEMY ( )
- JUMP_IF_NOT_RANDOM ( )
- JUMP_IF_NOT_TRUE ( )

```
   PROGRAM Example1 IS
    BEGIN
        IF next-is-wall THEN
            turnright
            turnright
            infect
        END IF
    END Example1
```

```
PROGRAM Example2 IS
BEGIN
    IF next-is-wall THEN
        turnright
        turnright
        infect
    ELSE
        infect
        move
    END IF
END Example2
```

```
PROGRAM Example3 IS
BEGIN
    WHILE next-is-not-empty DO
        IF next-is-wall THEN
```

```
            turnright
            turnright
            infect
        ELSE
            infect
            move
        END IF
    END WHILE
END Example3









PROGRAM Example4 IS

    INSTRUCTION TurnBackAndInfect IS
        turnright
        turnright
        IF next-is-enemy THEN
            infect
        END IF
    END TurnBackAndInfect

BEGIN
    WHILE true DO
        TurnBackAndInfect
    END WHILE
END Example4









/**
 * Returns the location of the next primitive instruction to execute in
 * compiled program {@code cp} given what the bug sees {@code wbs} and
 * starting from location {@code pc}.
 *
 * @param cp
 *            the compiled program
```

```
 * @param wbs
 *            the {@code CellState} indicating what the bug sees
 * @param pc
 *            the program counter
 * @return the location of the next primitive instruction to execute
 * @requires <pre>
 * [cp is a valid compiled BL program]  and
 * 0 <= pc < cp.length  and
 * [pc is the location of an instruction byte code in cp, that is, pc
 *  cannot be the location of an address]
 * </pre>
 * @ensures <pre>
 * [return the address of the next primitive instruction that
 *  should be executed in program cp given what the bug sees wbs and
 *  starting execution at address pc in program cp]
 * </pre>
 */
public static int nextPrimitiveInstructionAddress(int[] cp, CellState wbs,
        int pc) {
            // don't know what is CellState
        }
```