

Homework:

QUESTION 1

What do the following expressions evaluate to?

* Booleans

```
(equal 0 t)
(equal 3 6/2)
(equal #c(3 0) 3)
(and)
(and t)
(and t f)
```

QUESTION 2

What do the following expressions evaluate to?

* Numbers

```
42/6
#c(16/8 0)
(integerp 22/11)
(integerp nil)
(integerp #c(1 0))
(integerp '(1 2))
(rationalp nil)
(rationalp 4/6)
(rationalp 10)
(rationalp #c(1 1))
(rationalp '(1 2))
(natp 22/11)
(denominator 42/6)
(denominator 44/33)
(+)
(+ 3)
(+ 3 4)
(* 3 nil)
(* 3 4)
(/ 3 6)
(/ 3 0)
(nfix 0)
(nfix -1)
(ifix -1)
(ifix nil)
(rfix #c(1 0))
(natp (ifix -5))
(integerp (nfix -5))
(zp 0)
(zp nil)
(zp 33/22)
```

QUESTION 3

What do the following expressions evaluate to?

* Lists

```
(endp 10)
(endp nil)
(endp '(1 2))
(rest (cons 1 (cons 2 nil)))
(first (1 2 3))
(first '(1 2 3))
(first (cons 1 '(2 3)))
(first 2/3)
(rest '(1 2 3))
```

```
(rest nil)
(rest 2/3)
(true-listp (cons 1 '(2)))
(true-listp (cons 1 2))
(true-listp nil)
(true-listp 12)
```

Note: For the remaining questions, you are to define ACL2 functions. Use ACL2s in programming mode and use the design recipe described in class. Remember that this is the design recipe we saw in 211 extended with the following guidelines and checks:

Design guideline 1:

Test your programs with values that are outside of their intended domain.

Design guideline 2:

Atomic data outside the intended domain should be recognized and handled in a non-recursive way with the proper idiom.

Design guideline 3:

When expecting a true list, treat the final cdr as nil.

Design check 1:

After designing a program, perform a totality check.

QUESTION 4

Define an ACL2 function f that takes a natural number and returns a natural number, and that computes the following:

```
f(0) = 0
f(1) = 1
f(n) = 3f(n-1) + 4f(n-2) when n > 1
```

Note: This is sequence number A0155212 from The On-Line Encyclopedia of Integer Sequences. You can verify that $f(n)$ is the number of walks of length n between any two distinct vertices of the complete graph on 5 vertices.

QUESTION 5

Define the ACL2 function `div3`, a function that given a natural number returns `t` if the number is divisible by 3. Use the design recipe we've seen in class! Here's the idea: 0 is divisible by 3, but 1 and 2 are not. Any larger natural number, k , is divisible by 3 iff $k-3$ is divisible by 3.

Use `div3` to define `divi3`, a function that checks if an integer is divisible by 3. (Hint, you can do this without using recursion.)

QUESTION 6

Define the ACL2 function `quotient`, which takes as input a natural number, a , and a positive natural number, b , and return the largest natural number, x , such that x times $b \leq a$, e.g., `(quotient 12 3) = 4` and `(quotient 13 3) = 4`.