Andrew Giardina 4/2/2021

Language Study

Homework 5

Problem 1

even(0). // Fact: Zero is an Even Number

even(s(s(X))) ← even(X). // Rule: X is Even if X-2 is Even

odd(s(0)). // Fact: One is an Odd Number

odd(s(s(X))) ← odd(X). // Rule: X is Odd if X-2 is Odd

Problem 2

/\*

\* Format: int\_quotient(X,Y,IR) Meaning: X / Y = IR (IR: Integer Result)

\*/

int\_quotient(0,X,0). // Fact: 0 / X = 0

int\_quotient(X,s(0),X). // Fact: X / 1 = X

int\_quotient(X,Y,s(IR)) ← // 1st Base Case

plus(X,0,Y). // Checks X = Y, IR++, end

int\_quotient(X,Y,IR) ← // 2nd Base Case

plus(X,Z,Y), // Z = Y - X, Attempting to prove Y > X (Y != X by 1st Base)

natural\_number(Z). // Confirming Y > X, IR is done, end

int\_quotient(X,Y,IR) ← // Recursive Case

natural\_number(X), // X > 0 (X is still valid, X != 0 by 2nd Base)

plus(Y,Z,X), // Z = X - Y

int\_quotient(Z,Y,s(IR)). // Recurse with new X & IR values

Problem 3

adjacent(X,Y,[X,Y|Zs]). // Base Case: X & Y are adjacent when X is the head, and

// Y follows the head.

adjacent(X,Y,[Z|Zs]) ← // X & Y currently not in list, so denote list with Z as head

adjacent(X,Y,Zs). // Recurse on list by removing Z

last(X,[X]). // Base Case: X is last element when X is ONLY element

last(X,[R|Rs]) ← // X currently not in, denote list with R as head

last(X,Rs). // Recurse on list by removing R

Problem 4

double([],[]). // Base Case: Empty lists satisfy the logic,

// similar to Append's base case

double([X|List],[X,X|ListList]) ← // X must appear next to X in ListList, if this is

// true, remove them

double(List,ListList). // Recurse by removing all 'X's, will continue until

// empty if true