Solutions to In-Class Problems Week 7, Wed.

Problem 1.
The Elementary 18.01 Functions (F18's) are the set of functions of one real variable defined recursively as follows:

Base cases:

- The identity function, $\mathrm{id}(x) ::= x$ is an F18,
- any constant function is an F18,
- the sine function is an F18,

Constructor cases:

If f, g are F18's, then so are

1. $f + g$, fg, e^g (the constant e),
2. the inverse function $f^{(-1)}$,
3. the composition $f \circ g$.

(a) Prove that the function $1/x$ is an F18.

Warning: Don't confuse $1/x = x^{-1}$ with the inverse, $\mathrm{id}^{(-1)}$ of the identity function $\mathrm{id}(x)$. The inverse $\mathrm{id}^{(-1)}$ is equal to id.

Solution. $\log x$ is the inverse of e^x so $\log x \in$ F18. Therefore so is $c \cdot \log x$ for any constant c, and hence $e^{c \log x} = x^c \in$ F18. Now let $c = -1$ to get $x^{-1} = 1/x \in$ F18.[1] ■

(b) Prove by Structural Induction on this definition that the Elementary 18.01 Functions are *closed under taking derivatives*. That is, show that if $f(x)$ is an F18, then so is $f' ::= df/dx$. (Just work out 2 or 3 of the most interesting constructor cases; you may skip the less interesting ones.)

Solution. *Proof.* By Structural Induction on def of $f \in$ F18. The induction hypothesis is the above statement to be shown.

[1] There's a little problem here: since $\log x$ is not real-valued for $x \leq 0$, the function $f(x) ::= 1/x$ constructed in this way is only defined for $x > 0$. To get an F18 equal to $1/x$ defined for all $x \neq 0$, use $(x/|x|) \cdot f(|x|)$, where $|x| = \sqrt{x^2}$.

Base Cases: We want to show that the derivatives of all the base case functions are in F18.

This is easy: for example, $d\ \text{id}(x)/dx = 1$ is a constant function, and so is in F18. Similarly, $d\ \sin(x)/dx = \cos(x)$ which is also in F18 since $\cos(x) = \sin(x + \pi/2) \in$ F18 by rules for constant functions, the identity function, sum, and composition with sine.

This proves that the induction hypothesis holds in the Base cases.

Constructor Cases: $(f^{(-1)})$. Assume $f, df/dx \in$ F18 to prove $d\, f^{(-1)}(x)/dx \in$ F18. Letting $y = f(x)$, so $x = f^{(-1)}(y)$, we know from Leibniz's rule in calculus that

$$df^{(-1)}(y)/dy = dx/dy = \frac{1}{dy/dx}. \tag{1}$$

For example,

$$d\sin^{(-1)}(y)/dy = 1/(d\sin(x)/dx) = 1/\cos(x) = 1/\cos(\sin^{(-1)}(y)).$$

Stated as in (1), this rule is easy to remember, but can easily be misleading because of the variable switching between x and y. It's more clearly stated using variable-free notation:

$$(f^{(-1)})' = (1/f') \circ f^{(-1)}. \tag{2}$$

Now, since $f' \in$ F18 (by assumption), so is $1/f'$ (by part (a)) and $f^{(-1)}$ (by constructor rule 2.), and therefore so is their composition (by rule 3). Hence the righthand side of equation (2) defines a function in F18.

Constructor Case: $(f \circ g)$. Assume $f, g, df/dx, dg/dx \in$ F18 to prove $d(f \circ g)(x)/dx \in$ F18.

The Chain Rule states that
$$\frac{d(f(g(x)))}{dx} = \frac{df(g)}{dg} \cdot \frac{dg}{dx}.$$

Stated more clearly in variable-free notation, this is

$$(f \circ g)' = (f' \circ g) \cdot g'.$$

The righthand side of this equation defines a function in F18 by constructor rules 3. and 1.

The other Constructor cases are similar, so we conclude that the induction hypothesis holds in all Constructor cases.

This completes the proof by structural induction that the statement holds for all $f \in$ F18. ■

Problem 2.
Let p be the string [ ]. A string of brackets is said to be *erasable* iff it can be reduced to the empty string by repeatedly erasing occurrences of p. For example, here's how to erase the string [[[ ]] [ ]] [ ]:

$$[[[]][]][] \to [[]] \to [] \to \lambda.$$

On the other hand the string [ ]] [[[[[ ]] is not erasable because when we try to erase, we get stuck:

$$[]][[[[[]] \to][[[[] \to][[[\not\to$$

Let $textErasable$ be the set of erasable strings of brackets. Let $textRecMatch$ be the recursive data type of strings of *matched* brackets given in Definition **??**.

(a) Use structural induction to prove that

$$textRecMatch \subseteq textErasable.$$

Solution. *Proof.* We prove by structural induction on the definition of $textRecMatch$ that the predicate

$$P(x) ::= x \in textErasable$$

is true for all $x \in textRecMatch$.

Base case: ($x = \lambda$) The empty string is erasable by definition of $textErasable$ – it can be reduced to itself by erasing the substring **[** 0 times.

Constructor case: ($x =$ **[** s **]** t for $s, t \in textRecMatch$). By structural induction hypothesis, we may assume that $s, t \in textErasable$. So to erase x, erase s and then erase t to be left with the substring **[ ]**, and one more erasure leads to the empty string.

This completes the proof by structural induction, so we conclude that

$$\forall x.\ x \in textRecMatch \text{ IMPLIES } x \in textErasable$$

which by definition means that $textRecMatch \subseteq textErasable$.

■

(b) Supply the missing parts of the following proof that

$$textErasable \subseteq textRecMatch.$$

Proof. We prove by induction on the length, n, of strings, x, that if $x \in textErasable$, then $x \in textRecMatch$. The induction predicate is

$$P(n) ::= \forall x \in textErasable.\ (|x| \leq n \text{ IMPLIES } x \in textRecMatch)$$

Base case:

What is the base case? Prove that P is true in this case.

Solution. The base case is ($n = 0$). Now $P(0)$ is true because the empty string is the only string of length 0, and it is in $textRecMatch$ by the base case of Definition **??** of $textRecMatch$. ■

Inductive step: To prove $P(n+1)$, suppose $|x| \leq n+1$ and $x \in textErasable$. We need only show that $x \in textRecMatch$. Now if $|x| < n+1$, then the induction hypothesis, $P(n)$, implies that $x \in textRecMatch$, so we only have to deal with x of length exactly $n+1$.

Let's say that a string y is an *erase* of a string z iff y is the result of erasing a single occurrence of p in z.

Since $x \in textErasable$ and has positive length, there must be an erase, $y \in textErasable$, of x. So $|y| = n - 1$, and since $y \in textErasable$, we may assume by induction hypothesis that $y \in textRecMatch$.

Now we argue by cases:

Case (y is the empty string).

Prove that $x \in textRecMatch$ in this case.

Solution. In this case $x = p \in textRecMatch$. ■

Case ($y = [\ s\ ]\ t$ for some strings $s, t \in textRecMatch$.) Now we argue by subcases.

- **Subcase** (x is of the form $[\ s'\ ]\ t$ where s is an erase of s').
  Since $s \in textRecMatch$, it is erasable by part (b), which implies that $s' \in textErasable$. But $|s'| < |x|$, so by induction hypothesis, we may assume that $s' \in textRecMatch$. This shows that x is the result of the constructor step of $textRecMatch$, and therefore $x \in textRecMatch$.
- **Subcase** (x is of the form $[\ s\ ]\ t'$ where t is an erase of t').
  Prove that $x \in textRecMatch$ in this subcase.

  Solution. The proof is essentially identical to the previous case, with t, t' in place of s, s': Now t is erasable by part (b), so $t' \in textErasable$. But $|t'| < |x|$, so by induction hypothesis, we may assume that $t' \in textRecMatch$. This proves that x is the result of the constructor step of $textRecMatch$ and therefore $x \in textRecMatch$.

  ■

- **Subcase**($x = p[\ s\ ]\ t$).
  Prove that $x \in textRecMatch$ in this subcase.

  Solution. Let $t' ::= [\ s\ ]\ t$ and s' be the empty string. Then $x = [\ s'\ ]\ t'$. But we know $s', t' \in textRecMatch$, which implies that $x \in textRecMatch$ because it is the result the $textRecMatch$ constructor step applied to s', t'. ■

The proofs of the remaining subcases are just like this last one. **List these remaining subcases.**

Solution.

- **case** ($x = [\ ps\ ]\ t$),
- **case** ($x = [\ sp\ ]\ t$),
- **case** ($x = [\ s\ ]\ pt$),
- **case** ($x = [\ s\ ]\ tp$).

■

This completes the proof by induction on n, so we conclude that $P(n)$ holds for all $n \in \mathbb{N}$. Therefore $x \in textRecMatch$ for every string $x \in textErasable$. That is,

$$textErasable \subseteq textRecMatch \text{ and hence } textErasable = textRecMatch.$$

■

Problem 3.
Here is a simple recursive definition of the set, E, of even integers:

Definition. Base case: $0 \in E$.

Constructor cases: If $n \in E$, then so are $n + 2$ and $-n$.

Provide similar simple recursive definitions of the following sets:

(a) The set $S ::= \left\{2^k 3^m 5^n \mid k, m, n \in \mathbb{N}\right\}$.

Solution. We can define the set S recursively as follows:

- $1 \in S$
- If $n \in S$, then $2n$, $3n$, and $5n$ are in S.

■

(b) The set $T ::= \left\{2^k 3^{2k+m} 5^{m+n} \mid k, m, n \in \mathbb{N}\right\}$.

Solution. We can define the set T recursively as follows:

- $1 \in T$
- If $n \in S$, then $18n$, $15n$, and $5n$ are in T.

■

(c) The set $L ::= \left\{(a, b) \in \mathbb{Z}^2 \mid 3 \mid (a - b)\right\}$.

Solution. We can define a set $L' = L$ recursively as follows:

- $(0,0), (1,1), (2,2) \in L'$
- If $(a, b) \in L'$, then $(a+3, b)$, $(a-3, b)$, $(a, b+3)$, and $(a, b-3)$ are in L'.

Lots of other definitions are also possible. ■

Let L' be the set defined by the recursive definition you gave for L in the previous part. Now if you did it right, then $L' = L$, but maybe you made a mistake. So let's check that you got the definition right.

(d) Prove by structural induction on your definition of L' that

$$L' \subseteq L.$$

Solution. For the L' defined above, a straightforward structural induction shows that if $(c, d) \in L'$, then $(c, d) \in L$. Namely, each of the base cases in the definition of L' are in L since $3 \mid 0$. For the constructor cases, we may assume $(a, b) \in L$, that is $3 \mid (a - b)$, and must prove that $(a \pm 3, b) \in L$ and $(a, b \pm 3) \in L$. In the first the case, we must show that $3 \mid ((a \pm 3) - b)$. But this follows immediately because $((a \pm 3) - b) = (a - b) \pm 3$ and 3 divides both $(a - b)$ and 3. The other constructor case $(a, b \pm 3)$ follows in exactly the same way. So we conclude by structural induction on the definition of L' that $L' \subseteq L$. ■

(e) Confirm that you got the definition right by proving that

$$L \subseteq L'.$$

Solution. Conversely, we must show that $L \subseteq L'$. So suppose $(c, d) \in L$, that is, $3 \mid (c - d)$. This means that $c = r + 3k$ and $d = r + 3j$ for some $r \in \{0, 1, 2\}$ and $j, k \in \mathbb{Z}$. Then starting from base case $(r, r) \in L'$, we can apply the $(a \pm 3, b)$ constructor rule $|k|$ times to conclude that $(c, r) \in L'$, and then apply the $(a, b \pm 3)$ rule $|j|$ times to conclude that $(c, d) \in L'$. This implies that $L \subseteq L'$, which completes the proof that $L = L'$. ■

(f) See if you can give an *unambiguous* recursive definition of L.

Solution. This is tricky. Here is an attempt:

base cases: $(0, 0), (1, 1), (2, 2), (-1, -1), (-2, -2), (-3, -3)(1, -2), (2, -1), (-1, 2), (-2, 1) \in L$

Now the idea is to constrain the constructors so the two coordinates have absolute values that increase differing by at most 1, then one coordinate only can continue to grow in absolute value. Let

$$\text{Sg}(x) ::= \begin{cases} 1 \text{ if } x \geq 0, \\ -1 \text{ if } x < 0. \end{cases}$$

constructors: if $(a, b) \in L'$, then

- if $||a| - |b|| \leq 1$, then $(a + 3\text{Sg}(a), b + 3\text{Sg}(b)), (a + 3\text{Sg}(a), b), (a, b + 3\text{Sg}(b)) \in L'$,
- if $|a| > |b| + 1$, then $(a + 3\text{Sg}(a), b) \in L'$,
- if $|b| > |a| + 1$, then $(a, b + 3\text{Sg}(b)) \in L'$.

■