Math 491, Problem Set #1: Solutions

1. (a) *Write and run a program to compute* $f(n) = \sum_{k=0}^{n}(-1)^k\binom{n}{k}^2$.

   Here's one that'll do the job:

```
f := proc(n)
local k;
RETURN(sum((-1)^k*binomial(n,k)^2,k=0..n));
end;
```

   Notice that the variable `k` is declared to be `local`; that means that if you had a variable named `k` sitting around in your Maple session before you ran the program `f`, the value of the variable won't be affected.

   Running the above program for $n = 1$ through $n = 10$ gives the values 0, -2, 0, 6, 0, -20, 0, 70, 0, -252.

   Note by the way that you can run into problems with the way Maple tries to be "helpful" by using what it knows. If you tell Maple

```
a := sum((-1)^k*binomial(n,k)^2,k=0..n);
```

   it will eagerly simplify this to an expression involving the gamma function. If you then ask it to evaluate this expression for a particular value of n, say by typing $\mathtt{subs(n = 2, a);}$, you'll still get an expression involving the gamma function. At this point you could gruffly tell it

```
simplify(%);
```

   (or

```
simplify(");
```

   with very old versions of Maple), meaning "simplify the expression you just gave me", and it will indeed give you the answer -2. But suppose you now type

```
simplify(subs(n=3,a));
```

Then Maple balks at the fact that the denominator of a (or rather, the analytic expression for `a` that Maple has introduced) blows up, and scrupulously refuses to replace the fraction by 0 even though the numerator is finite and the denominator is infinite. Ditto for `eval(subs(n = 3, a));`.

All computer algebra systems have features like this; when you encounter one, please let me know about it!

(b) *Devise a conjecture about the value of $f(n)$.*

When n is odd, $f(n)$ vanishes, while when n is even (say $n = 2m$), $f(2m) = (-1)^m \binom{2m}{m}$.

(c) *Prove your conjecture using the algebraic interpretation of $\binom{n}{k}$ as the coefficient of x^k in $(1+x)^n$.*

$\binom{n}{k}$ is the coefficient of x^k in $(1+x)^n$, and $(-1)^k\binom{n}{k}$ is the coefficient of x^k in $(1-x)^n$. Hence, as in the example explained in class, $f(n)$ is equal to the coefficient of x^n in $(1+x)^n(1-x)^n$. We can rewrite this polynomial as $(1-x^2)^n$. Note that all the exponents in the expansion of this polynomial are even. Hence, if n is odd, $f(n) = 0$ as claimed. On the other hand, when n is even (say $n = 2m$), we see that the coefficient of x^{2m} in $(1-x^2)^n$ is equal to the coefficient of x^m in $(1-x)^n$, which is equal to $(-1)^m\binom{n}{m} = (-1)^m\binom{2m}{m}$, as claimed.

(d) *Prove your conjecture using the interpretation of $\binom{n}{k}$ as the number of combinations of n things taken k at a time.*

As in the example studied in class, we view this as a problem of signed counting, where we are choosing a committee of size n from a group consisting of n men and n women, and where a committee counts as plus or minus according to whether the number of men is even or odd.

Case I: n is odd. Every plus committee can be matched up with a minus committee, and vice versa, if one simply complements the committee. That is, if you have a committee with an even (resp. odd) number of men, and you create a new committee consisting of all the people who weren't on the first committee, then the new committee will have an odd (resp. even) number of men. Hence the alternating sum is 0.

Case II: n is even, with $n = 2m$. Let's assume that the n men and n women are married in pairs, and that the couples are numbered 1 through n. Call a committee *boring* if it consists entirely of couples, and *interesting* otherwise. I claim that there is a way to pair up each interesting committee with another interesting committee of opposite sign. Namely, find the lowest-numbered person on the committee whose spouse isn't on the committee, and replace that person by his/her mate. You can check (and this fact is important!) that if this operation is performed a second time, you get back the committee you started with. Hence the operation gives rise to a way of pairing up all the possible interesting committees. Since this pairing pairs up plus-committees with minus-committees, all the interesting committees end up cancelling one another, and the signed enumeration of all the committees gives the same answer as the signed enumeration of just the boring committees. But each boring committee, consisting of m men and their wives, has sign $(-1)^m$, and the number of boring committees is $\binom{n}{m}$ (since there are n couples from which one must choose m couples). Hence the alternating sum is $(-1)^m\binom{n}{m}$, as claimed. (Note that this approach applies to Case I as well: When n is odd, there are no boring committees.)

Observe that the manifest algebraic analogy

$$\binom{n}{k} \quad \text{is to} \quad (-1)^k\binom{n}{k} \quad \text{as} \quad \binom{n}{k}^2 \quad \text{is to} \quad (-1)^k\binom{n}{k}^2$$

is reflected at the combinatorial level by the way in which we introduce extra structure (thereby breaking some of the symmetry of the original problem). When analyzing $\sum(-1)^k\binom{n}{k}$, we introduced a *special element* of the set. When analyzing $\sum\binom{n}{k}^2$, we introduced a partition of the set into *two equal subsets*. When analyzing $\sum(-1)^k\binom{n}{k}^2$, we do not superimpose these extra structures in the most obvious way (i.e., by dividing the set of people into men and women and choosing a special person, or perhaps choosing a special man and a special woman), but we do in a sense superimpose the two sorts of structure: we introduce a partition of the set into *two equal subsets*, and for **each** element of one subset we choose a *special element* of the other set, in such a way that the two subsets get paired up. Can anyone come up with a more rigorous way of thinking about this?

2. *Define a sequence of functions $f_0(x), f_1(x), f_2(x), \ldots$ where $f_0(x) = x$, $f_1(x) = x$, and for all $n > 1$, $f_n(x) = ([f_{n-1}(x)]^2 + 1)/f_{n-2}(x)$. Thus, $f_2(x) = x + x^{-1}$, $f_3(x) = x + 3x^{-1} + x^{-3}$, etc.*

   A good program for this in Maple is

```
f := proc(n)
if (n < 2) then x;
else expand(simplify((f(n-1)^2+1)/f(n-2)));
fi; end;
```

   Note that this is an example of *recursive programming*. Also note that the `RETURN` statements have been eliminated. The value that is returned by a Maple program is the last value that was computed inside the program, whether or not it was inside a `RETURN` statement. Note the use of semi-colons; unfortunately, Maple can be fussy about this. Also note that, just as each `proc` must be matched by an `end`, each `if` must be matched by `fi` (mnemonic: "fi" is "if" spelled backwards).

   You may be wondering why both `expand` and `simplify` are present, and in that particular order. If you are new to Maple, try to modify this program in various ways to see whether it still works, and if it does work, what it does. For instance, what happens if you replace "`expand(simplify(...))`" by "`simplify(expand(...))`" or by "`simplify(...)`" or by "`expand(...)`" or by just "`...`"?

   (a) *Formulate a conjecture about the values of $f_n(1)$.*

   Typing `subs(x=1,f(1));`, `subs(x=1,f(2));`, etc., we get the values $1, 1, 2, 5, 13, 34, \ldots$, so we conjecture that the sequence of values of $f_n(1)$ consists of every other term of the Fibonacci sequence.

   (b) *Formulate a conjecture about the values of $f_n(-1)$.*

   Typing `subs(x=-1,f(1));`, etc., we get the values $-1, -1, -2, -5, -13, -34, \ldots$, so we conjecture that the sequence of values of $f_n(1)$

consists of the negative of every other term of the Fibonacci sequence. In fact, we can prove by induction that each polynomial $f_n(x)$ is an odd function of x, so this conjecture is a trivial consequence of the previous conjecture.

(c) *Formulate a conjecture about the values of $f_n(i)$, where $i = \sqrt{-1}$.*

This time we use `subs(x=I,f(1));`, etc. (Did you figure out that Maple has complex numbers, too? If you didn't, it's good to know that Maple has a `help` facility. In this case, the command

```
help(imaginary)
```

(no semicolons needed for `help` commands) would have told you that the square root of minus one is `I`. Another way to ask the question is just to type

```
?imaginary
```

(again, no semicolon needed).

Note that if you had used the `expand(...)` version of the program (with no `simplify(...)`), you'd find that $f_4(I)$ is undefined, because of a vanishing denominator in a sub-expression.

To list the values of $f_n(i)$ as i goes from 0 to 10 (say), here's a handy shortcut:

```
for n from 0 to 10 do subs(x=I,f(n)); od;
```

(`od` pairs with `do` just as `fi` pairs with `if`.) Running this program one gets the sequence of values $i, i, 0, -i, -i, 0, i, i, 0, -i, -i$ displayed vertically down the screen. One may conjecture that this pattern continues, with period 6.

If you prefer horizontal display, use the `seq` operator: the command

```
[seq(subs(x=I,f(n)),n=0..10)];
```

gives the sequence of values in the form of a list. The opening and closing brackets tell Maple that it is building a list; we'll learn much more about working with lists later in the term.

Incidentally, to end a Maple session, all you need to do is type `quit` (no semicolon needed).

I should say here (and should have said sooner) that I am by no means an expert at Maple; if any of the above comments about programming seem confusing or incorrect, please let me know, so that I can correct them!