

Forum:[Hack Forums](#)

Thread: [n0sya's Math Problem of the Week: Week 1 \(Subsets\)](#)

the wolf of blockchain - November 6, 2016, 5:54 pm

Hello fellow members,

I've decided to post weekly math challenges just for the hell of it. Each problem will be worth around one point, and whoever accumulates the most points gains bragging rights around these parts. Anyways, make sure to post your answer (along with how you got it - no cheating!) in spoiler tags. Your work must be your own, but there are no limits to what resources you can use.

That being said, have fun!

Problem 1 (1 point):

Spoiler ([Click to View](#))

How many subsets of $\{0,1,2,3,\dots,31\}$ do not have a pair of elements which differ by 2?

A correct solution will be posted on November 13th.

aimware - November 6, 2016, 6:27 pm

Spoiler ([Click to View](#)) 2^{31} ? No idea how to get it, just fairly certain that's the answer.

the wolf of blockchain - November 6, 2016, 6:33 pm

(Nov 6, 2016 07:27 PM)aimware Wrote: [\[CITE\]](#) Spoiler ([Click to View](#)) 2^{31} ? No idea how to get it, just fairly certain that's the answer.

Good attempt! That would be the amount of subsets for a set with 31 elements, but that's not what the question asks for ?

Adlil - November 6, 2016, 9:41 pm

(Nov 6, 2016 06:54 PM)n0sya Wrote: [\[CITE\]](#) Hello fellow members,

I've decided to post weekly math challenges just for the hell of it. Each problem will be worth around one point, and whoever accumulates the most points gains bragging rights around these parts. Anyways, make sure to post your answer (along with how you got it - no cheating!) in spoiler tags. Your work must be your own, but there are no limits to what resources you can use.

That being said, have fun!

Problem 1 (1 point):

Spoiler ([Click to View](#))

How many subsets of $\{0,1,2,3,\dots,31\}$ do not have a pair of elements which differ by 2?

A correct solution will be posted on November 13th.

That's pretty interesting, I'll take a crack. I'm going to type up my entire thought process.

Spoiler ([Click to View](#))

Let $S=\{0,1,2,\dots,31\}$

We want to find $|A|=\{S' \text{ in } P(S) \mid \text{For all } x \text{ in } S', \text{ there does not exist } y \text{ in } S' \text{ such that } |x-y|=2\}$

Now with a definition that clean it'd be easy to run a program to find it, even with the size of this

powerset. In fact this would be a prime candidate for quick recursive algorithm as any subset of a proposed set must meet the criteria. But I'd rather try to be mathematically clever...

Now I'd like to try out a few smaller sets and see if there's a clear relationship between the number of elements in the set and it's size.

If S is empty we have 1 element

If S is {0} we have 2 elements({{0},empty})

If S is {0,1} we have 4 elements

If S is {0,1,2} we have 6 elements(new one is {1,2} and {2}... that's not a large increase

If S is {0,1,2,3} we have 9 elements {0,3}, {2,3}, and {3} as new elements

If S is {0,1,2,3,4} we have {0,1,4},{0,3,4},{0,4},{1,4},{3,4},{4} as new elements for a total of 15.

So what can we see from this? Well, say set S_n has n elements. I see that for all $n \geq 3$, $|S_n| = |S_{(n-1)}| + |S_{(n-2)}|$, that's right, something similar to the Fibonacci sequence. With that prediction I think $|S_{\{0,1,2,3,4,5\}}| = 24$ ie that adding 5 to the set would give 9 new elements. Well let's see, we would add on {0,1,4,5},{0,1,5},{0,4,5},{1,2,5},{1,4,5},{2,5},{4,5},{5}... darn that didn't work. But at this point I actually doubt my counting, because that's saying the size would be 23, which is a prime number, very, very out of place here... oh and what do you know, I missed {0,5} AND {1,5} whoops.. but my prediction was still off then, as there are 25 elements.

BUT I see another interesting pattern here, when n is even, $|S_n|$ is a square. Now that's something good. But what's the growth rate here, we have 1,4,9,25... or for sqrts, 1,2,3,5. Kinda weirdly reminiscent of the fibonacci sequence again(0,1,1,2,3,5). I'd be more confident if I could calculate how many subsets there were for 7 elements but well... my prediction is 64 and that's too many ahaha...

Here's a thought, when you add the n th element, it adds the number of subsets that already exist that don't contain an element 2 less than it. So when we add 6 we look at every set we already have that doesn't contain 4. Well we had 10 sets with 4 so we'd add $25-10=15$ sets. So the total is 40 sets. Now when we add 7 we add every set that doesn't have 5 in it, which is 10(weird to note kinda) + how many more we added when we included 6. So we know there less than 70 sets... let's see what sets would include 5 when we added in 6... {6,5,2,1},{6,5,2},{6,5,1,0},{6,5,1},{6,5,0},{6,5}... so the total size for a set with $n=7$ is 64! That follows my prediction.

So we have a fibonacci sequence for the square root of the size of our set every-other number. Off by a couple spots too... So I'm seeing 1,2,3,5,8 corresponding to 0-8 elements(even positions). Well we have 32 total elements, so we're looking for the 16th spot in this sequence. That's not too bad to just add up...

1,2,3,5,8,13,21,34,55,89,144,233,377,610,987,1597... So I'd suppose the answer is 1597^2 , and this is the only part I'm using a calculator since I don't have paper atm and have just been typing this. Anyways my answer is:

***** 2550409 *****

A later recount of this revealed I miscounted a bit in my earlier statement.

I wrote out the fibonacci sequence, with the idea that 1 correlated to 0 elements, 2 to 2 elements, 3 to 4 elements and so on. Following that, 1597 went with 30 elements, but in this case there are 32, so I need to reproach my solution to the next position in the Fibonacci sequence squared. So

$(987+1597)^2 = 6677056$

***** 6677056 *****

This brought a bit of nostalgia, really, in hindsight there's better ways to do this but I tackled this problem like I would've back when I did MathCounts in middle school.

the wolf of blockchain - November 6, 2016, 9:55 pm

(Nov 6, 2016 10:41 PM)Adlil Wrote: [\[CITE\]](#) That's pretty interesting, I'll take a crack. I'm going to type up my entire thought process.

-snip-

That was a very interesting way to tackle this mathematically! I'm impressed. To keep the spirit of the game, I won't say if it's right or wrong, but checking with an algorithm never hurts ?

Phizo - November 7, 2016, 11:20 am

Can you clarify on your definition of a pair? It is ambiguous. I have already developed a solution, it is just a matter of fine-tuning with respect to your definition.

At the moment, I assume you define a pair as any set of the form $\{a, b\}$ such that $a = b-2$ and $b = a+2$. You have not defined any particular relation, so I can only assume the pairs are unordered at this point. $\{0, 2\}$ is clearly a pair, but how about a superset $S = \{0, 1, 2\}$? At this point, it is ambiguous, because no relation is defined and set equality allows the set to be rearranged: $S = \{0, 2, 1\} = \{1, 2, 0\}$.

Thanks. :)

the wolf of blockchain - November 7, 2016, 11:52 am

(Nov 7, 2016 12:20 PM)Phizo Wrote: [\[CITE\]](#) Can you clarify on your definition of a pair?

To clarify, the question asks how many subsets exists such that no two elements in them differ by two. The set $\{1,2,5\}$ would pass, but the set $\{1,2,3,5\}$ would not. I hope this clears it up!

WOLGA - November 7, 2016, 10:26 pm

hello, the answer is [\(Click to View\)](#)
6677056

-wolga

imajerk - November 8, 2016, 12:42 am

That is actually not an easy question... the way I would solve it is to set up a recurrence relation. Let $a(n)$ denote the # of such subsets from the set $\{0,1,2,...,n\}$. The number that we're interested is denoted by $a(31)$, I imagine. So the way I see how to solve this is to divide the problem into two cases. When set contains the #31 and when it does not. When it does not we need to count the number of subsets of $\{0,1,2,...,30\}$ which is $a(30)$. And when it does contain 31 we need to count the subsets of the form $\{31,...\}$ which is related to $a(29)$, but not quite, we would need to fix a small counting issue, and so the recursion is roughly; $a(n)$ is a sum of $a_{(n-1)}$ and $a_{(n-2)}$.

Does that make any sense to anyone else?

Adlil - November 8, 2016, 6:42 am

(Nov 8, 2016 01:42 AM)imajerk Wrote: [\[CITE\]](#) That is actually not an easy question... the way I would solve it is to set up a recurrence relation. Let $a(n)$ denote the # of such subsets from the set $\{0,1,2,...,n\}$. The number that we're interested is denoted by $a(31)$, I imagine. So the way I see how to solve this is to divide the problem into two cases. When set contains the #31 and when it does not. When it does not we need to count the number of subsets of $\{0,1,2,...,30\}$ which is $a(30)$. And when it does contain 31 we need to count the subsets of the form $\{31,...\}$ which is related to $a(29)$, but not quite, we would need to fix a small counting issue, and so the recursion is roughly; $a(n)$ is a sum of $a_{(n-1)}$ and $a_{(n-2)}$.

Does that make any sense to anyone else?

Spoiler [\(Click to View\)](#)

I followed what you said, but the sum of $a_{(n-1)}$ and $a_{(n-2)}$ won't quite do it, because $a(31) = a(30) + \text{all subsets of } a(30) \text{ that does not contain the element 29}$. $a(29)$ won't do it because it has many elements that actually contain 29.

MasterCurser - November 8, 2016, 8:06 am

I'm fairly certain that this is from abstract algebra, and I really don't want to travel back to those theorems. Good luck to everyone trying it out.

Phizo - November 8, 2016, 1:27 pm

2 ez tbh imo etc et al. :)

I decided to solve it mostly using first-order predicate calculus, computationally in Prolog, which I did follow up with some handwork.

Here is an illustration of the program that computes all solutions in $P([0, 5])$, where P is a function which maps a given set into a powerset (the set of all subsets):

```
PHP Code
<?php
| ?- [n0sya].
compiling /private/tmp/n0sya.pl for byte code...
/private/tmp/n0sya.pl compiled, 21 lines read - 3838 bytes written, 6 ms
```

```
(1 ms) yes
| ?- make_list(0, 5, L), maplist(go, L).
```

```
Reduced powerset: [[0],[]]
Number of subsets: 2
```

```
Reduced powerset: [[0,1],[1],[],[0]]
Number of subsets: 4
```

```
Reduced powerset: [[1,2],[2],[],[1],[0],[0,1]]
Number of subsets: 6
```

```
Reduced powerset: [[2,3],[3],[],[2],[1],[1,2],[0,3],[0],[0,1]]
Number of subsets: 9
```

```
Reduced powerset: [[3,4],[4],[],[3],[2],[2,3],[1,4],[1],[1,2],[0,3,4],[0,4],[0],[0,3],[0,1,4],[0,1]]
Number of subsets: 15
```

```
Reduced powerset: [[4,5],[5],[],[4],[3],[3,4],[2,5],[2],[2,3],[1,4,5],[1,5],[1],[1,4],[1,2,5],[1,2],[0,4,5],[0,5],[0],[0,4],[0,3],[0,3,4],[0,1,4,5],[0,1,5],[0,1],[0,1,4]]
Number of subsets: 25
```

It can solve for: $[0, 10]$ in ~7ms.

$[0, 15]$ in ~179ms

$[0, 20]$ in ~6 seconds

$[0, 21]$ in ~12 seconds

$[0, 22]$ in ~24 seconds

$[0, 24]$ in ~1.9 minutes

[0, 25] in ~3.47 minutes

[0, 26] in ~6.8 minutes

As you may notice, an increment in set cardinality causes an increase in computational time by a factor of ~2.09 on average, which can be represented functionally by $t(f(n+k)) = t(f(n)) \cdot 2.09^k$, where t is a function that measures computational time and $f = \text{go}$ (my top-level Prolog predicate).

That is, the running time is $O(2^n)$ (exponential with a linear exponent), which is fairly inefficient. Note that Prolog (and logic programming in general) is not concerned with runtime efficiency: the declarative paradigm promotes theorem-proving, and consequently program correctness, to programmers and proof procedures, which are often computationally expensive.

As it is such a computationally complex problem, I cannot practically compute the solution for the set [0, 31], as it will take $t(f(31)) \sim t(f(26)) \cdot 2.09^5 = \sim 4.5$ hours.

So, I will compute the rest of the solution by hand. :)

Spoiler ([Click to View](#))

Allow P to represent the powerset and $f = \text{go}$ (my top-level Prolog predicate). I analyse f with the intent of finding a pattern such that I can conceptually derive solutions for any x in $f(x)$.

I worked backwards in order to determine a pattern and define the necessary base conditions. I layed out $f(x) = y$ such that $0 \leq x \leq 11$.

Quite clearly, f is akin to the Fibonacci function, but with a separate rule for odd numbers in its domain.

n is even $\Rightarrow f(n) = f(n-1) + f(n-2)$

n is odd $\Rightarrow f(n) = f(n-1) + f(n-3) + f(n-4)$

Code $f(0) = 2 = f(-1) + f(-1)$

$f(1) = 4 = f(0) + f(-1) + f(-1)$

$f(2) = 6 = f(1) + f(0)$

$f(3) = 9 = f(2) + f(0) + f(-1)$

$f(4) = 15 = f(3) + f(2)$

$f(5) = 25 = f(4) + f(2) + f(1)$

$f(6) = 40 = f(5) + f(4)$

$f(7) = 64 = f(6) + f(4) + f(3)$

$f(8) = 104 = f(7) + f(6)$

$f(9) = 169 = f(8) + f(6) + f(5)$

$f(10) = 273 = f(9) + f(8)$

$f(11) = 441 = f(10) + f(8) + f(7)$

These are the base cases I subsequently defined:

Code $P(\{\}) = \{\{\}\}$

$f(-1) = 1 = |P(\{\})|$

The time has come. What is $f(31)$? A Husaid would never FUCK know a thing of this. :/ jaz0r ?

Code $f(31) = f(30) + f(28) + f(27)$

$= [f(29) + f(28)] + [f(27) + f(26)] + [f(26) + f(24) + f(23)]$

$= [f(29) + f(28)] + [f(27) + 602070] + [602070 + 229970 + 142129]$

$= [f(29) + f(28)] + [974169 + 602070] + 974169$

$= [f(29) + 1576239] + 1576239 + 974169$

$= [[f(28) + f(26) + f(25)] + 1576239] + 1576239 + 974169$

$$= [[1576239 + 602070 + 372100] + 1576239] + 1576239 + 974169$$

$$= 6677056$$

Hence, $f(31) = 6677056$. That'll be \$10. Cheers and thanks.

the wolf of blockchain - November 8, 2016, 2:25 pm

(Nov 8, 2016 02:27 PM)Phizo Wrote: [\[CITE\]](#) 2 ez tbh imo etc et al. :\

-big snip-

Very well done! You've laid out an intricate processes to get your solution (even picking up what Adlil left off!), and for some reason it reminds me of a Strong Proof back in discrete. As always, I won't say if it's right or wrong, but I'm going to challenge myself to form a solution with pure mathematics.

Phizo - November 8, 2016, 7:35 pm

(Nov 8, 2016 03:25 PM)n0sya Wrote: [\[CITE\]](#) Very well done! You've laid out an intricate processes to get your solution (even picking up what Adlil left off!), and for some reason it reminds me of a Strong Proof back in discrete. As always, I won't say if it's right or wrong, but I'm going to challenge myself to form a solution with pure mathematics.

You can easily form a purely mathematical solution by hypothesising f 's rules (as I did) and then proving them using mathematical induction, which is what I planned to do at the end, but decided I couldn't be bothered after everything I had written. I would argue that my solution is purely mathematical despite that, though. I defined my solutions in terms of branches of mathematical logic: first-order logic and set theory, respectively. The first-order logic is just abstracted by a notational system for describing computation (i.e. a programming language), and the set theory (mostly functions/relations) assumes definitions described by the logic, as redefinition is simply a redundancy.

Thanks for the compliments. Please keep the challenges coming. :)

For anyone who is interested, I came up with a simple solution to dramatically speed up the problem solver I developed. I incorporated an optimisation technique called memoisation. Technically, it is no longer pure logic programming by incorporating this. Without this optimisation: for every call to $f(n)$, $f(x)$ is also called for all x in $[0, (n-1)!]$ (excluding the base case $f(-1)$). This is expensive when you consider, if you want $f(31)$, you call $f(26)$ 5 times, which takes ~34 minutes. So, imagine with all the rest of the calls to f (hence why it takes ~4.5 hours). With the optimisation: for every call to $f(n)$, $f(x)$ is also called for all x in $[0, n-1]$. This basically allows each solved subproblem to be stored for reuse, so that it is not recomputed ever again. Essentially, the factorial function f is now linear.

I verified that the computed $f(31)$ matches my answer.

Solving for $f(80)$ instantly:

PHP Code

```
<?php
```

```
| ?- [n0ysa2].
```

```
compiling /private/tmp/n0ysa2.pl for byte code...
```

```
/private/tmp/n0ysa2.pl compiled, 42 lines read - 7219 bytes written, 7 ms
```

```
(2 ms) yes
```

```
| ?- go(80).
```

```
Number of subsets: 116139356908771352
```

```
yes
```

These are the code changes (impure code is often fairly gross):

