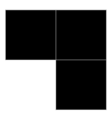
General Instructions

- This is just a summary of the full completion and submission requirements from the course Syllabus: `https://q.utoronto.ca/courses/314442#problem-sets`.
- **Worth**: 2%. **Please read the completion and submission requirements carefully**.
- **Due**: By 21:00 on Tuesday 19 September 2023, on MarkUs. Submissions will be accepted up to one week late with a penalty of roughly −10% for each day late—see the Syllabus for full details.
- Problem sets are to be submitted individually. You are free to discuss the problems and their solutions with others, but **you must write and submit your own individual answers**. Please read the Academic Integrity section on the Syllabus for full details of exactly what is allowed and what is not.
- Submissions may be typeset or handwritten *legibly*, but must be made in *one document per question*, *in an accepted format*, *using the filenames specified on MarkUs*—**documents that cannot be displayed *directly on MarkUs* will not be marked**, even if MarkUs allows you to submit them. We recommend (*but do NOT require*) the use of LaTeX to produce high-quality documents.
- Remember that your submission must meet specific conditions to receive credit. **Please read the Syllabus** for full details of how to benefit the most from this problem set, and how to submit your work.

Problems

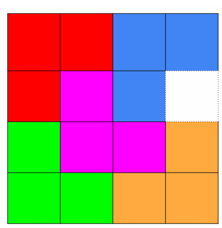
Review

"Review" problems can be solved using only knowledge from prerequisite courses. This does not necessarily mean that they are easy, but they require no new knowledge.

1. A triomino is an L-shaped tile: 

Consider a $2^n \times 2^n$ grid with one square removed. We want to tile this grid using triominoes. By "tile the grid using triominoes", we mean place triominoes on the grid such that each cell in the grid (except for the one that is removed) should be covered by exactly one triomino. Here's an example tiling of a $2^2 \times 2^2$ grid with one cell removed (the removed cell is white and the triominoes are distinguished by their color).

$G_n = \{2^1 \times 2^1$ (-1), $2^2 \times 2^2$ (-1), ..., $2^n \times 2^n$ (-1)$\}$



For each $n \in \mathbb{N}$, let G_n be the set of all $2^n \times 2^n$ grids with one cell removed, and let T_n be the set of all tilings of elements of G_n using triominoes.

16 ×16 96 16 256

For each $n \in \mathbb{N}$, let $P(n)$ be: $\forall g \in G_n, \exists t \in T_n$, t tiles g.

$\forall g \in G_1, \exists t_1 \in T_1 . t_1$ tiles g
$\Downarrow$
$\forall g \in G_2, \exists t_2 \in T_2,$ t_2 tiles g.

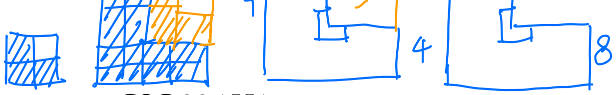
(a) Write out $P(1) \Rightarrow P(2)$ according to the definition of P (but do not expand G and T).

(b) Write out the structure for a direct proof of $P(1) \Rightarrow P(2)$.

① Assume P(1) holds.
② Prove P(2). Since in every 2x2 grid with 1 removed it can be done which.

32×32

1×3+1 4 (1+1)×3+1 8 8 (16+5)×3+1 = 64



1, 1^2 4, 2^2 16, 4^2 64, 8^2 $(64+21)\times 3+1 = 85\times 3+1 = 255+1 = \boxed{256}$ $\boxed{256}$

there will be a 2×2 left in the middle so put another. $2^3\times 2^2 - 1$ can be tiled.

(c) Explain how the diagram above illustrates an instance of that direct proof.

(d) Prove, by Simple Induction, that for each natural n you can tile any $2^n \times 2^n$ grid with one cell missing using only triominoes, i.e., $\forall n \in \mathbb{N}, P(n)$.

Make sure the variable introduced for the Inductive Step has an explicit domain and, if applicable, explicit further restrictions. Explicitly indicate whenever the IH is used. The IH is universally quantified, so for each use of it explicitly indicate which instance of the quantification you are using.

we can always make a square with its corner empty so every time next we'll make the empty part faced inside to construct a small square and put an extra trio inside the middle.

Practice

"Practice" problems are straightforward applications of the course material. Little creativity or insight should be required to solve them.

2. Assume for this question that x represents a non-zero real number. $x \in \mathbb{R} / \{0\}$

For each $n \in \mathbb{N}$, let $c_n = x^n + 1/x^n$. $c_n = x^n + \frac{1}{x^n}$

(a) Implement a Python function with parameters x and n that (ignoring floating-point issues) returns c_n. Clearly state the pre-conditions on x and n in a header comment. $x \in \mathbb{R}/\{0\}$; $n \in \mathbb{N}$.

The implementation must be recursive. Do not use any loops, write any helper functions, nor call any exponentiation functions. Before each (although there might only be one) recursive call: write comments that clearly indicate which values of x and n it handles, why its arguments thus satisfy the pre-conditions, and in what way the arguments are for a "smaller" problem.

HINT: examine $(x+1/x)(x^n+1/x^n)$. $(x+\frac{1}{x})\cdot(x^n+\frac{1}{x^n}) = x^{n+1} + \frac{1}{x^{n-1}} + x^{n-1} + \frac{1}{x^{n+1}} = (x^{n+1}+\frac{1}{x^{n+1}}) + (x^{n-1}+\frac{1}{x^{n-1}})$ $\Rightarrow c_1 \cdot c_n = c^{n+1} + c^{n-1} \Rightarrow c^{n+1} = c_1\cdot c_n - c^{n-1}$

NOTE: remember that you are only submitting a PDF document, not source code! Your answer will be graded mostly by reading your *comments*, to confirm that you are implementing the correct algorithm. (However, if you put together a working program, you will be able to test your answer—and your understanding—even if this is not required for full marks.) $n-1 \geq 0$ $\Rightarrow c_n = c_1 \cdot c_{n-1} - c_{n-2}$

```
def q-2(n, x) -> float
    if type(n) != int or x == 0:
        return 'x'
    elif n == 1:
        c1 = x + 1/x
        return c1
    elif n == 2:
        c2 = x^2 + 1/x^2
        return c2
    else:
        c1 = x + 1/x
        cn-1 = q-2(n-1, x)
        cn-2 = q-2(n-2, x)
        cn = c1 cn-1 - cn-2
        return cn
```

(b) State a recurrence for the sequence c. You may choose any indexing, but must make sure that any variables used for indexing have explicit domains and, if applicable, explicit further restrictions. $n = 3$. $x = 2$.

(c) Prove, by Complete Induction, that if $x + 1/x$ is an integer then so is $x^n + 1/x^n$ for each natural n.

Make sure the variable introduced for the Inductive Step has an explicit domain and, if applicable, explicit further restrictions. Explicitly indicate whenever the IH is used, which instance of its restricted quantification is being used, and why that instance is within the restricted quantification.

3. Consider the sequence of real numbers $a_1, a_2, a_3, \ldots$ defined by:

$$a_n = \begin{cases} 1 & \text{if } n = 1, \\ \left(a_{\lfloor\sqrt{n}\rfloor}\right)^2 + 2 \cdot a_{\lfloor\sqrt{n}\rfloor} & \text{if } n \in \mathbb{N} \text{ with } n \geq 2. \end{cases}$$

$n \in \mathbb{N}$.

(a) Implement a Python function that takes a positive natural number n and returns a_n.

The implementation must be recursive, without any loops nor helper functions, and must document the pre-conditions and recursive call(s) as described in Question 2a (except it does not have an x parameter).

(See the note for Question 2a.)

$a_{n+1} = (a_{\lfloor \sqrt{n+1} \rfloor})^2 + 2a_{\lfloor \sqrt{n+1} \rfloor}$.

$n=2$ | $n=3$ → rtn. 3.

(b) Implement a Python function that takes a positive natural number n and raises an exception if n is 1, otherwise it returns a_n.

The implementation must be recursive, without any loops nor helper functions (in particular it must not call your function from part (a)).

(See the note for Question 2a.)

(c) Determine the smallest natural n_0 so that a_n is a multiple of 3 for each natural $n \geq n_0$. Prove that claim about n_0 using Complete Induction.

Make sure the variable introduced for the Inductive Step has an explicit domain and, if applicable, explicit further restrictions. Explicitly indicate whenever the IH is used, which instance of its restricted quantification is being used, and why that instance is within the restricted quantification.

Extra

"Extra" problems go beyond the learning objectives of the course, making connections to topics we will not cover, or giving you a glimpse of how the material can be taken further. **You do NOT have to attempt "extra" problems for full credit.** They are provided for your interest only.

4. A Rubik's cube is a fun puzzle/toy. If you don't have a Rubik's cube, check out this simulator: `https://ruwix.com/online-puzzle-simulators/`.

Assume the blue face of the cube is always facing you. A *state* of the cube describes the current position of all the stickers. For example, the solved state of the cube is the one where every face has just one color (namely, the one matching the center square of the face).

Let *Rubiks* denote the set of possible Rubik's Cube states, i.e., the set of states that can be obtained by starting at the solved configuration and applying some sequence of turns.

The cube has 6 faces. Let f_i be the operation of turning the ith face of the cube a quarter turn clockwise. Let $Turns = \{f_1, \ldots, f_6\}$ be the set of turns one can perform on the Rubik's cube.

Start with a solved Rubik's cube and consider any fixed sequence of turns. Now repeat that same sequence of turns many times. **Eventually, somewhat magically, the Rubik's cube will return to the solved configuration!** In this problem, we will prove this fact.

(a) A function $f : A \to B$ is called *injective* if distinct inputs get mapped to distinct outputs (`https://en.wikipedia.org/wiki/Injective_function`). Show that for all $f \in Turns$, f is an injective function from *Rubiks* to *Rubiks*.

HINT: You might find useful the following characterization of injective functions: `https://en.wikipedia.org/wiki/Injective_function#Injections_can_be_undone`.

(b) Suppose f, g are functions from $A \to A$, then their composition $g \circ f : A \to A$ is the function that first applies f and then applies g. Prove by induction that for all natural numbers $n \geq 1$, if $g_1, g_2, \ldots, g_n$ are each injective, then the composition $g_n \circ g_{n-1} \circ \cdots \circ g_1$ is injective. Conclude that an if g is an arbitrary (finite) sequence of turns, then g is a injective function from *Rubiks* → *Rubiks*.

(c) Give an upper bound on $|Rubiks|$, i.e., find some K such that $|Rubiks| \leq K$.

(d) Let g be an arbitrary sequence of turns. Denote

$$g^m = g \underbrace{\circ \cdots \circ}_{m \text{ times}} g$$

to be the function that applies g m times. Also, let $s \in \textit{Rubiks}$ be the solved state of the cube. Show that for some $m \in \mathbb{N}$ with $m \geq 1$, $g^m(s) = s$. That is, after m applications of the sequence of moves defined by g, we return to the solved state!

HINT: Consider the sequence $s_0, s_1, s_2, \ldots$, where $s_0 = s$, and $s_i = g^i(s)$. Also, the Pigeonhole Principle (`https://en.wikipedia.org/wiki/Pigeonhole_principle`) might be useful.

Q2(c). If $x+\frac{1}{x}$ is an int, then $x^n+\frac{1}{x^n}$ is an int for each $n\in\mathbb{N}$.

Given statement TP.: $\forall n\in\mathbb{N}.\ P(n)$, which $P(n)$: $x^n+\frac{1}{x^n}$ is an int where $x\in\mathbb{R}/\{0\}$

Let $x\in\mathbb{R}/\{0\}$. Let $n\in\mathbb{N}$. Assume $x+\frac{1}{x}$ is an int.

Proof: we prove this by complete induction on n.

B.C.. Let $n=1$.

By assumption, $P(1)=x+\frac{1}{x}$ is an int.

$P(1)$ is T.

I.S. Let $n>1$

I.H. Assume $\forall k,\ 1\le k<n.\ P(k)$.

WTS: $P(n)$.

From I.H. $P(1)$: $(x+\frac{1}{x})$ is an int.

- - - - - : $P(n-1)$: $(x^{n-1}+\frac{1}{x^{n-1}})$ is an int.

. . . . : $P(n-2)$. $(x^{n-2}+\frac{1}{x^{n-2}})$ is an int.

Since $(x+\frac{1}{x})(x^{n-1}+\frac{1}{x^{n-1}})-(x^{n-2}+\frac{1}{x^{n-2}})$ is an int.

$$(x+\frac{1}{x})(x^{n-1}+\frac{1}{x^{n-1}})-(x^{n-2}+\frac{1}{x^{n-2}})$$
$$=x^n+\frac{1}{x^{n-2}}+x^{n-2}+\frac{1}{x^n}-x^{n-2}-\frac{1}{x^{n-2}}$$
$$=x^n+\frac{1}{x^n}$$ is an int.

$\therefore P(n)$. T.

Q3c: Determine n_0 (a_{n_0} is a multiple of 3); $\forall n \geq n_0$. a_{n_0} is a multiple of 3.

When $n_0 = 2$. n_0 is the smallest natural no. so that a_n is a multiple of 3 for $\forall n \geq n_0$.

$n_0 \neq 1$. as when $n_0 = 1$, $a_n = 1$. which is not a multiple of 3.

Given statement T.P.: $\forall n \in \mathbb{N}. n \geq n_0. P(n)$. which $P(n)$: a_n is a multiple of 3.

Let $n \in \mathbb{N}$.

Proof: We ...... c.i. on n.

B.C. Let $2 \leq n < 4$

$$P(2): a_2 = (a_{\lfloor\sqrt{2}\rfloor})^2 + 2 \cdot a_{\lfloor\sqrt{2}\rfloor}$$
$$= a_1^2 + 2a_1 = 1^2 + 2 = 3. \text{ is a multiple of 3.}$$

$$P(3): a_3 = (a_{\lfloor\sqrt{3}\rfloor})^2 + 2 \cdot a_{\lfloor\sqrt{3}\rfloor}$$
$$= a_1^2 + 2a_1 = 1^2 + 2 = 3$$

B.C is T.

I.S. Let $n \geq 4$.

I.H. Assume $\forall k. 2 \leq k < n, P(k)$.

Since $n \geq 4$, $\lfloor\sqrt{n}\rfloor < n$.

Since $\lfloor\sqrt{n}\rfloor < n$ and $4 \leq n$, $2 \leq \lfloor\sqrt{n}\rfloor$ as 2 is the smallest value of $\lfloor\sqrt{n}\rfloor$.

which $2 \leq \lfloor\sqrt{n}\rfloor < n$.

Since $\lfloor\sqrt{n}\rfloor$ is an integer geq 2, from I.H. we can always find $k' = \lfloor\sqrt{n}\rfloor$ which $P(k')$ is true

which $a_{k'} = 3p$, $p \in \mathbb{N}$.

Thus. $a_n = (a_{\lfloor\sqrt{n}\rfloor})^2 + 2a_{\lfloor\sqrt{n}\rfloor}$

$$= (a_{k'})^2 + 2a_{k'}$$
$$= (3p)^2 + 2 \cdot 3p$$
$$= 9p^2 + 6p$$
$$= 3(3p^2 + 2p).$$

Let $q = 3p^2 + 2p$. Since $p \in \mathbb{N}$, $q \in \mathbb{N}$.

Therefore $a_n = 3q$. $q \in \mathbb{N}$. a_n is a ---- of 3.

$\therefore P(n)$ is T.