Due: Tuesday, 3rd November, 15:00 (AEDST)

Submission is through WebCMS/give and should be a single pdf file, maximum size 2Mb. Prose should be typed, not handwritten. Use of LaTeX is encouraged, but not required.

Discussion of assignment material with others is permitted, but the work submitted *must* be your own in line with the University's plagiarism policy.

Problem 1 (16 marks)
Let S be a set, and let $R_1, R_2 \subseteq S \times S$ be equivalence relations on S.

(a) Show that $R_1 \cap R_2$ is an equivalence relation. (6 marks)

(b) For $x \in S$ let $[x]_i$ denote the equivalence class of x under R_i ($i = 1, 2$), and let $[x]$ denote the equivalence class of x under $R_1 \cap R_2$. With justification, define $[x]$ in terms of $[x]_1$ and $[x]_2$. (4 marks)

(c) Prove or disprove: $R_1 \cup R_2$ is an equivalence relation. (6 marks)

Problem 2 (12 marks)
Let S be a set. Prove, or find a counterexample to disprove the following for all binary relations $R_1, R_2 \subseteq S \times S$:

(a) If R_1 and R_2 are reflexive then $R_1; R_2$ is reflexive (4 marks)

(b) If R_1 and R_2 are symmetric then $R_1; R_2$ is symmetric (4 marks)

(c) If R_1 and R_2 are transitive then $R_1; R_2$ is transitive (4 marks)

Problem 3 (34 marks)
Let $R \subseteq S \times S$ be any binary relation on a set S. Consider the sequence of relations $R^0, R^1, R^2, \ldots$, defined as follows:

$$\begin{aligned} R^0 &:= I = \{(x, x) : x \in S\}, \text{ and} \\ R^{n+1} &:= R^n \cup (R; R^n) \text{ for } n \geq 0 \end{aligned}$$

Suppose there exists $i \in \mathbb{N}$ such that $R^i = R^{i+1}$.

(a) Prove that $R^j = R^i$ for all $j \geq i$. (6 marks)

(b) Prove that $R^j \subseteq R^i$ for all $j \geq 0$. (4 marks)

(c) If $|S| = k$, explain why $R^{k^2} = R^{k^2+1}$. (4 marks)

(d) Let $P(n)$ be the proposition that for all $m \in \mathbb{N}$: $R^n; R^m = R^{n+m}$. Prove that $P(n)$ holds for all $n \in \mathbb{N}$.
Hint: Use results from Assignment 1 (6 marks)

(e) If $|S| = k$, show that R^{k^2} is transitive. (4 marks)

(f) If $|S| = k$, show that $(R \cup R^{\leftarrow})^{k^2}$ is an equivalence relation. (6 marks)

(g)* If $|S| = k$ show that $(R \cup R^{\leftarrow})^{k^2}$ is the least upper bound (with respect to $\subseteq$) of all equivalence relations that contain R. (4 marks)

Problem 4* (6 marks)

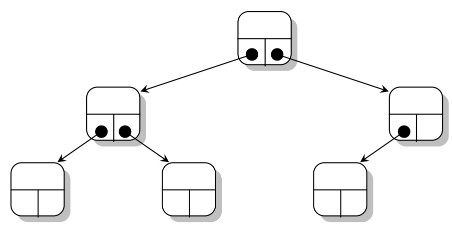
Let $f(n)$ be defined recursively as follows:

$$f(0) = 0 \qquad f(n) = f(\lfloor \frac{n}{3} \rfloor) + 3f(\lfloor \frac{n}{5} \rfloor) + n \text{ for } n \geq 1$$

Show that $f(n) \in O(n)$.

Problem 5 (20 marks)

A *binary tree* is a data structure where each node is linked to at most two successor nodes:



If we include empty binary trees (trees with no nodes) as part of the definition, then we can simplify the description of the data structure. Rather than saying a node has 0, 1, or 2 successor nodes, we can instead say that a node has exactly two *children*, where a child is a binary tree. That is, we can abstractly define the structure of a binary tree as follows:

- (B): An empty tree, τ
- (R): An ordered pair $(T_{\text{left}}, T_{\text{right}})$ where T_{left} and T_{right} are trees.

So, for example, the above tree would be defined as the tree T where:

$$\begin{aligned} &T = (T_1, T_2), \text{ where} \\ &\quad T_1 = (T_3, T_4) \text{ and } T_2 = (T_5, \tau), \text{ where} \\ &\qquad T_3 = T_4 = T_5 = (\tau, \tau) \end{aligned}$$

That is,

$$T = \Big(\big((\tau, \tau), (\tau, \tau) \big), \big((\tau, \tau), \tau \big) \Big)$$

A *leaf* in a binary tree is a node that has no successors (i.e. it is of the form (τ, τ)). A *half-leaf* in a binary tree is a node that has exactly one successor (i.e. it is of the form (T, τ) or (τ, T) where $T \neq \tau$). The example above has 3 leaves (T_3, T_4, and T_5) and 1 half-leaf (T_2). For technical reasons (that will become apparent) we assume that an empty tree has 0 leaves and 1 half-leaves.

(a) Based on the recursive definition above, recursively define a function $\mathsf{count}(\mathsf{T})$ that counts the number of nodes in a binary tree T. (4 marks)

(b) Based on the recursive definition above, recursively define a function $\mathsf{leaves}(\mathsf{T})$ that counts the number of leaves in a binary tree T. (4 marks)

(c) Based on the recursive definition above, recursively define a function $\mathsf{half\text{-}leaves}(\mathsf{T})$ that counts the number of half-leaves in a binary tree T. (4 marks)

(d) If T is a binary tree, let $P(\mathsf{T})$ be the proposition that $\mathsf{count}(\mathsf{T}) = 2 \times \mathsf{leaves}(\mathsf{T}) + \mathsf{half\text{-}leaves}(\mathsf{T}) - 1$. Prove that $P(\mathsf{T})$ holds for all binary trees T. Your proof should be based on your answers given in (b) and (c). (8 marks)

Problem 6 (12 marks)

Consider the following two algorithms that naïvely compute the sum and product of two $n \times n$ matrices.

```
sum(A,B):
  for i ∈ [0,n):
    for j ∈ [0,n):
      C[i,j] = A[i,j] + B[i,j]
    end for
  end for
  return C
```

```
product(A,B):
  for i ∈ [0,n):
    for j ∈ [0,n):
      C[i,j] = add{A[i,k] * B[k,j] : k ∈ [0,n)}
    end for
  end for
  return C
```

Assuming that adding and multiplying matrix elements can be carried out in $O(1)$ time, and add will add the elements of a set S in $O(|S|)$ time:

(a) Give an asymptotic upper bound, in terms of n, for the running time of sum. (3 marks)

(b) Give an asymptotic upper bound, in terms of n, for the running time of product. (3 marks)

When n is even, we can define a recursive procedure for multiplying two $n \times n$ matrices as follows. First, break the matrices into smaller submatrices:

$$A = \begin{pmatrix} S & T \\ U & V \end{pmatrix} \qquad B = \begin{pmatrix} W & X \\ Y & Z \end{pmatrix}$$

where S, T, U, V, W, X, Y, Z are $\frac{n}{2} \times \frac{n}{2}$ matrices. Then it is possible to show:

$$AB = \begin{pmatrix} SW + TY & SX + TZ \\ UW + VY & UX + VZ \end{pmatrix}$$

where $SW + TY$, $SX + TZ$, etc. are sums of products of the smaller matrices. If n is a power of 2, each smaller product (SW, TY, etc) can be computed recursively, until the product of 1×1 matrices needs to be computed – which is nothing more than a simple multiplication, taking $O(1)$ time.

Assume n is a power of 2, and let $T(n)$ be the worst-case running time for computing the product of two $n \times n$ matrices using this method.

(c) With justification, give a recurrence equation for $T(n)$. (4 marks)

(d) Find an asymptotic upper bound for $T(n)$. (2 marks)

Advice on how to do the assignment

Collaboration is encouraged, but all submitted work must be done individually without consulting someone else's solutions in accordance with the University's "Academic Dishonesty and Plagiarism" policies.

- Assignments are to be submitted via WebCMS (or give) as a single pdf file.
- When giving answers to questions, we always would like you to prove/explain/motivate your answers. You are being assessed on your understanding and ability.
- Be careful with giving multiple or alternative answers. If you give multiple answers, then we will give you marks only for your worst answer, as this indicates how well you understood the question.
- Some of the questions are very easy (with the help of external resources). You may make use of external material provided it is properly referenced[1] – however, answers that depend too heavily on external resources may not receive full marks if you have not adequately demonstrated ability/understanding.

[1]Proper referencing means sufficient information for a marker to access the material. Results from the lectures or textbook can be used without proof, but should still be referenced.