COMP2022: Formal Languages and Logic
Assignment 1

Due: 23:00pm Sunday 9th September (week 6)

1 Lists [30%]

Exercise 1.1

Let Q, K, and A be arbitrary expressions. Consider the Church encoding for the list $\{Q, K, A\}$.

- Write the expression needed to construct this list using $CONS$
- Rewrite the $CONS$ macros to write the expression using the $PAIR$, $TRUE$ and $FALSE$ macros

Exercise 1.2

Let

$$H \equiv \Big(\lambda fga.(ISNIL\ a)\ NIL\ \ CONS\ \ g\ (HEAD\ a)\ \ f\ g\ (TAIL\ a))\Big)\Big)$$

and

Reduce $\big(F\ (\lambda x.SUCC\ x$

- You may reduce simple expressions like $(SUCC$ $3\}) \twoheadrightarrow_\beta \{2, 3\}$ directly. However, put a comment beside the line to state which macro you're

- You may do more than one β-reduction on the same line, as long as it's obvious what you've done.
- After working through the first recursive call in detail, you may reduce the subsequent recursive calls on one line each.

Exercise 1.3

Describe what the function F from the previous question does *in general* (i.e. where the two expressions applied to it are not necessarily the examples above.)

2 Trees [30%]

In this question you will invent an encoding in lambda calculus to represent a binary tree. You may assume that we already have an encoding of integers which supports basic arithmetic, including negative numbers (i.e. -1); addition $(+\ a\ b)$; and comparisons $(<\ a\ b)$, $(=\ a\ b)$;

Exercise 2.1

For each of the following tree operations, invent an expression which encodes it, briefly describe how it works, and β-reduce an example to show it that works.)

- NIL should represent an empty tree.
- $(MAKETREE\ e\ a\ b)$ should make a tree with e at the root, and with the given subtrees a, b attached as the left and right children. Give a couple of examples.
- $(ROOT\ t)$ should return the element stored at the root of the tree.
- $(LEFT\ t)$ should return the subtree which is the left child of the root.
- $(RIGHT\ t)$ should return the subtree which is the right child of the root.
- $(ISEMPTY\ t)$ should return TRUE if the tree is empty, FALSE otherwise.
- $(ISLEAF\ t)$ should return TRUE if the tree is just a leaf, FALSE otherwise.

You do not have to include error handling (i.e. it doesn't matter what a nonsense expression like $(LEFT\ NIL)$ reduces to).

Exercise 2.2

Using the operations you defin [illegible] lex functions. briefly describe how they [illegible]

- $(SUM\ t)$ should sum all the values stored in the tree.
- $(HEIGHT\ t)$ should return the height of the tree.

- $(ISPROPER\ t)$ should return $TRUE$ if t is a proper tree (every position is either a leaf, or it has 2 children).
- $(MAKEPROPER\ t)$ should return a proper tree, equivalent to the original tree except wherever a position had exactly one child, that child is no longer in the tree (i.e. we return the maximal proper subtree of t.)

3 Sorting [20%]

Exercise 3.1

Suppose we wanted to use our tree as a binary search tree for Church numerals. Write expressions for the following functions. briefly describe how they work, but you do not need to give fully worked examples.

- $(INSERT\ t\ x)$ should return a tree which is the same as t, except with x inserted at the correct leaf position (reminder of the basic algorithm: if x is less than the element at the current position, move left, otherwise move right, until you find an empty place to put the element as a new leaf).
- $(SEARCH\ t\ x)$ should return $TRUE$ if x exists in t, or $FALSE$ otherwise.

(note: you don't have to implement a *balanced* binary search tree. An unbalanced one is fine.)

Exercise 3.2

$(INORDER\ t)$ should return an inorder traversal of the tree as a list).

Exercise 3.3

$(SORT\ a)$ should sort a list using the treesort algorithm (make a binary search tree using the elements of the list, then return the inorder traversal of the tree as a list).

Exercise 3.4

Modify your algorithm and data structure so that it accepts an arbitrary comparison function (instead of assuming we're always using numbers).

4 LISP [20%]

Implement a binary search tree in LISP, and use it to sort some numbers with treesort. You do *not* need to implement all the methods above, and you can (and probably should!) use a simpler encoding.

The following method signatures are required:

```
(defun insert (tree x)
  ;; inserts x to the binary search tree
)
(defun list-to-tree (mylist &optional tree)
  ;; note: the second argument 'tree' will be nil by default
  ;; inserts every element of the list 'mylist' into the tree
)
(defun inorder (tree)
  ;; list giving the inorder traversal of the tree
)
```

You are not required to implement any of the other methods (but some will be *very* useful). You can (and probably should) use a simpler encoding.

For full marks:

- It should work
- The code should be implemented in a functional style. e.g.
  - There should be no global variables
  - The body of the functions should not need 'let' statements, 'for' loops etc.
  - However, partia
- The code should include s

Include your full source code in your report, as well as some examples of r how it works:

e.g.

```
(print
  (insert nill 1)
)
;; this will output your representation of a tree containing just 1

(print
  (insert (insert nill 1) 3)
)
;; this will output your representation of a tree containing just 1 and 3

(print
  (list-to-tree (list 4 6 2 0 2 8 2))
)
;; this will output your representation of a larger tree

(print
  (inorder (list-to-tree (list 4 6 2 0 2 8 2)))
)
;; this should output (0 2 2 2 4 6 8)
```

5 Submission details

Due **23:00pm Sunday 9th September 2018**. I strongly encourage you to submit draft work prior to the deadline. However, TurnItIn only accepts late submissions if a submission has not yet been made, and it's common for students to submit a minute or two late... So I have set the submission deadline on Canvas to **23:59:00** to save myself from a flood of urgent emails at 23:01.

5.1 Late submission

The late submission policy is detailed in the administrivia lecture slides from week 1. I will *not* penalise submissions made in the 59 minute grace period. From 23:59:00 onwards you're very nearly an hour late, so the full 20% penalty is applicable. Please notify me if you intend to make a late submission, or if you believe you will not be able to submit, to make it easier for me to support you.

5.2 Submission format

You must submit a report as a single document (.pdf or .docx) to TurnItIn. The written parts of the report must be *text*, not images of hand-writing. Any diagrams can be images, of course.

Don't forget to include your LISP code and examples.

LaTeX is highly recommended for typesetting your formulas (I'll put a template on Ed in a few days). It's acceptable to use a \ to denote a λ, if your editor doesn't support it. e.g. $TRUE = (\backslash xy.x)$

5.3 A note on Academic Integrity

I would very much prefer that you invented your own encodings based what you've learned in the past 4 weeks. However, if your submission does rely on any examples / work found outside the the course, then:

1. Cite your sources prope
2. Take care to distinguis
3. Explain the cited work in your own words, to demonstrate you fully

Appropriately cited work will be awarded *partial* marks in b on of how much the student has contributed to the answer. Using other's work ons is plagiarism, which can be subject to severe penalties (and it makes me sad every time I have to report a student for it.)