Programming Paradigms (COMP3007) Exam Review

IMPORTANT *Structure of exam as per professor:*

- A total of 8 pages with 4 questions including sub questions
    - **Expect the following definitions and comprehension questions**
        - Iterative vs. recursive
        - Comparisons
        - Small code snippets
        - Let, lambda, define, cons
        - Coding in scheme and prolog
            - Lists, objects, streams, data abstraction, sum of squares, complex objects, local state, closure, recursion, iteration
    - **Contour diagrams**
    - In terms of prolog, queries similar to assignment #4 as well as recursion and list problems
        - Possible applications of cut

IMPORTANT Definitions and Comprehension:

Prolog is declarative programming.
Scheme is Functional programming.

Special Forms is an expression that follows special rules.
eg)Define, Let, Lambda, If, Cond, And, Or, Begin,Sequence

Applicative Order: Evaluates expressions before they are bound to a variable. This is more efficient because it can avoid multiple evaluations.

Normal Order: Evaluates expressions only when their value is needed. Unneeded code may never be evaluated.

Procedural abstraction is a way of hiding the implementation of the function using a black box method. You hide the implementation.

Lexical Scoping: variables in the containing scope are visible within the nested scopes. Scoping can be nested to any depth.

Linear Recursive Process: Expansion followed by contraction. There is a chain of deferred operations(*)The interpreter keeps track of operations to perform later. The amount of information grows linearly.

Linear Iterative Process: State can be summarized by a fixed number of state variables together with a rule on how to update them. The end test is optional. The number of steps grows linearly.

Scheme executes an iterative process in constant space which is called tail-recursive.

Abstraction barriers: isolate difference levels of the system.
Referentially Transparent: A language that supports the notion that references can be substituted for their values, without hanging the result of an expression.

Imperative Programming: Programming that makes extensive use of assignment

Binding: is an association of a name with a value.

Bound variable is a variable for which binding exists.
Free variable is used locally but is bound in an enclosing scope.
The duration of time that a variable is bound is called its extent.
A **frame** is a table of bindings.
An **environment** is a sequence of frames. The **root frame** is the primitive environment.

Contour Model a graphical model of runtime environments for block structured programming languages.
Model consists of a set of contours each corresponding to a given environment both lexical and dynamic.

Lexical Scoping: The environments are searched in the order matching the nesting of their definitions.
Dynamic scoping: The environments are searched in the reverse order of invocation.

The **interpreter** which determines the meaning of expressions in a programming language is just another program.
An **evaluator** that is written in the same language that it evaluates is said to be metacircular.
-Show how to define the operational semantics of the language
-Provides an executable specifications for the language
-Demonstrates how bindings and environments work
-Provides a non-trivial test-case for the language developed.
Two main **Functions for the Scheme Metacircular Interpreter**:

1. Eval and then 2. Apply

Prolog has only one data type: Terms: Number, Atom (any symbol that starts with a lowercase, constant value), Variable(any symbol that starts with an uppercase, un specified value), and Compound Term(a tuple of terms tagged with a relation name)

Two **Forms of Clauses**.

1. A **fact** is a statement that is universally true.
2. A **rule** is a statement that is conditionally true.

Cuts: how to stop prolog from continuing to succeed.

1. Faster Execution
2. Fewer backtrack points

Green cut: only prune computational paths that do not lead to new solutions. Cuts that do not affect the programs meaning.

Red cut: Cuts whose presence in a program changes the meaning of the program.

IMPORTANT Studying Components:

1. Lecture material on *COMP3007* website
2. Assignment material
3. Midterm material located in appendix

Lecture Material Notes:

Introduction

Program - a thing that might be edited with a text editor
Process - an abstract computation to which the program gives rise, upon being executed.
Domain - subject matter that the process is about.

Attributes of a good programming language:

1. Readability and clarity
2. Writability
3. Naturalness for application
4. Reliability
5. Ease of program verification
6. Programming environment
7. Portability
8. Cost

Types of languages:

Imperative programming: command driven, sequence of statements (C, Pascal).
Procedural programming: provides modularity, subroutines (C, Pascal).
OO Programming: data encapsulated, complex objects built from simple ones, inheritance, polymorphism (C++, Java).
Declarative programming: focus on statements, does not describe the control flow (SQL, Prolog).
Functional programming: primitive functions to build complex ones, evaluate mathematical functions (Scheme).
Logic programming: facts and logical rules, filters applied to data, queries (Prolog).
Meta programming: genetic programming, compilers.
Parallel programming: division of labour, multiple processes, multiple partial solutions.
Event driven programming: proceed in response to events, handlers registered to events (User interfaces).
Visual programming: programs display visually (XCode).

Programming languages are comprised of:

1. Primitive expressions

2. Means of combination
3. Means of abstraction

Scheme
The part of the program where a binding applies is the **scope** of the binding.
A set a bindings in memory is called the **environment**.

Evaluation rule:
1. Evaluate subexpression

Procedural Abstraction
Building Abstractions with Data:
Streams:
Mutable State, Environments, and Objects
Meta-Circular Interpreter:
Prolog:
Prolog Recursion:
Prolog Cuts:

Assignment material:

Assignment 1
Assignment 2
Assignment 3
Assignment 4

IMPORTANT Direct Questions from Fall 2016 COMP 3007

1. What is the definition of linear recursive programming?
2. What is the definition of linear iterative programming?
3. Implement stream-cons, stream-car, and stream-cdr
4. Implement a Stack using cons pairs. Ensure that the stack has the operations push, pop, peek, and size.
5. Using the following facts:
    a. male(X) % X is male
    b. female(X) % X is female
    c. father(X,Y) %X is the father of Y
    d. mother(X,Y) %X is the mother of Y
    e. married(X,Y) %X is married to Y

Write prolog rules to define the following 11 relationships:

1. parent(X,Y) %X is the parent of Y
2. different(X,Y) %X and Y are different
3. is_mother(X) % X is a mother
4. is_father(X) % X is a father
5. aunt(X,Y) % X is an aunt of Y
6. uncle(X,Y) % X is an uncle of Y
7. sister(X,Y) % X is a sister of Y
8. brother(X,Y) % X is a brother of Y
9. grandfather(X,Y) % X is a grandfather of Y
10. grandmother(X,Y) % X is a grandmother of Y
11. Ancestor(X,Y) % X is ancestor of Y

6. Consider this database of facts, describing what something is made up of.

```
has(bicycle,wheel,2).
has(bicycle,handlebar,1).
has(bicycle,brake,2).
has(wheel,hub,1).
has(bicycle,frame,1).
has(car,steering_wheel,1).
has(car,stereo,1).
```

Write a predicate partof(X,Y) that succeeds if Y is part of X.

```
?- partof(wheel,spoke).
True.
?- partof(bicycle,spoke).
True.
?- partof(car,spoke).
False.
```

partof(X,Y) can also be used to enu

```
partof(X,Y):- has(X,Y,_); has(X,Z,_), partof(Z,Y).

?- partof(bicycle,X).
X = wheel;
X = handlebar;
X = break;
X = frame;
X = hub;

?- partof(X,spoke).
```

X = wheel;
X = bicycle.

7. Missing lines from the meta-circular interpreter and you fill it in.

8. Create infinite streams with force and delay.