**COMP 141 - Exam 2**

**Section1**

1. What types would ML infer for the following function: fun double x = 2 \* x;

* The types that will be inferred will be integers. X will be an integer and the answer will be an integer

2. What is static type checking? In what ways is it better than dynamic type checking? Give an example of an error that can be caught through static type checking.

* Statically typed programming languages do type checking (the process of verifying and enforcing the constraints of types) at compile time. Dynamically typed programming languages do type checking at run-time. Advantages of static type checking:
  + More **secure**: more errors found prior to execution
  + More **efficient**: no type-checking at runtime
* Ex. Type mismatch errors. In ML, when composing functions the output of the first function must be the the same type as the input of the second function. Int and int/ real and real

**Section2**

1. Define what a data type is, and specify 4 data types commonly found in imperative languages.

* A data type is a set of values with a set of operations on those values having certain properties.
* Examples are bool, char, float, double

4. Give an example of a language that is:  
 A. Strongly Typed

* This means the language specifies complete type systems that can be statically applied and guarantees all data-corrupting errors detected at translation time. C is an example of a strongly typed language.

B. Weakly Typed

* This means the language has loopholes in the type system that may allow some unsafe programs through. Perl is an example of a weakly typed language

C. Untyped

* This means the language does not have any type declaration. The language Mathematica is untyped.

**Section3**

5. Formulate the following facts and inference rules as a set of Prolog clauses:

Alice is a student.

Bob is a student.

Alice does HW.

Bob does not do HW.

Alice studies hard.

Bob studies hard.

If a student studies hard and does the HW, that student is a good student.

If a student studies hard or does the HW, that student will learn.

student(alice).

student(bob).

doesHW(alice).

not(doesHW(bob)).

studyHard(alice).

studyHard(bob).

goodStudent(X) :-

student(X),

doesHW(X),

studyHard(X).

willLearn(X) :-

student(X),

doesHW(X).

willLearn(X) :-

student(X),

studyHard(X).

7. Define the following terms related to logical programming:

a. Predicate - Function that true/false capabilities

b. Fact - Statement that makes a declaration

c. Rule - Inferred truths that can also be called recursively

**Section4**

3. What is the difference between a statement and an expression?

1. Give an example of an expression

* An expression returns a value and produces no side effect.
* An example is: x + 5

1. Give an example of a statement

* Executed for its side effect and returns no value
* An example is: ;

1. Give an example of something that is both

* C = a + b;

9. Give an example piece of code for a situation where *short circuit evaluation* would be useful. Explain why it would be useful.

* An example of a piece of code where short circuit evaluation would be useful would be an or statement: if(x = 1 || y = 3). This is useful because if x = 1 then there is no need to check if y = 3 since only one of the conditions needs to be true to start the if statement.