3 Unification (14 marks total)

3.1 (5x1 mark)

?- X = 3*5.

outcome: yes bindings: X = 3*5

?- X = 3*5, Y is X.

outcome: yes bindings: `X = 3*5 Y = 15`

?- Y is X, X = 3*5.

outcome: error bindings:

?- X = 5, Y = X.

outcome: yes bindings: X=5 Y=5

?- X = 5, Y == X.

outcome: no bindings:

3.2 (3x2 marks)

?- p(X, X) = p(f(3, Z), f(Y, Y)).

outcome: yes bindings: `X = f(3,3)  Y = 3 Z = 3`

?- p(X, f(b, Y)) = p(g(Y, Z), f(Z, a)).

outcome: yes bindings: `X = g(a,b)   Y = a   Z = b`

?- p(f(X), Y, f(f(Z))) = p(Z, f(Z), f(Y)).

outcome: yes bindings: `X = _1   Y = f(f(_1))   Z = f(_1)`

3.3 Most general unifier (3 marks)

For the following terms **t1** and **t2**, find the most general unifier **w**, if it exists, or otherwise show that the terms cannot be unified. Show all the steps in the unification procedure.

t1 = f(f(X, Y), a) **t2 = f(f(f(Y, 3), a), Y)**

S0 = {f(f(X, Y), a) == f(f(f(Y, 3), a), Y)} w0 = {}
S1 = {f(X, Y) == f(f(Y, 3), a), a == Y } w1 = {}
w2 = {Y/a} (can also choose other substitution here)
S2 = w2(S1) = {f(X,a) == f(f(a,3),a), a==a}
S3 = {X == f(a,3), a==a, a==a}
w3 = {X/f(a,3), Y/a}
S4 = w3(S3) = {f(a,3) == f(a,3), a==a, a==a}

Solved, w = w3.

4 Understanding a Prolog Program and Backtracking (16 marks)

```
vowel(a).
vowel(e).
vowel(i).
consonant(b).
consonant(c).

letters([C]) :- consonant(C).
letters([C,V|R]) :- consonant(C), vowel(V), letters(R).

fiveletters1(W) :- letters(W), length(W,5).

fiveletters2(W) :- W = [C1,V1,C2,V2,C3],
            consonant(C1), vowel(V1), consonant(C2),
            vowel(V2), consonant(C3).
```

4.1 (3x3 marks) Given the Prolog program above, what is the first answer for the following Prolog queries, and what are the results of asking for more answers by pressing semicolon **;**

?- `letters(X).`

first: `X = [b]`

next four: `X = [c] ; X = [b,a,b] ; X = [b,a,c] ; X = [b,a,b,a,b]`

?- `fiveletters1(X).`

first: `X = [b,a,b,a,b]`

next four: `X = [b,a,b,a,c], then endless recursion.`

?- `fiveletters2(X).`

first: `X = [b,a,b,a,b]`

next four: `X = [b,a,b,a,c] ; X = [b,a,b,e,b] ; X = [b,a,b,e,c] ; X = [b,a,b,i,b]`

4.2 (3 marks) Do the two queries **`fiveletters1(X)`** and **`fiveletters2(X)`** compute the same answers with backtracking? If yes, why? If no, why not?

No because `fiveletters1 goes into an endless loop after two answers but fiveletters2 keeps generating all 72 solutions`

4.3 (2x2 marks) One way to find out how many answers in total are computed by **`fiveletters2(X)`** is to issue the query

```
?-  findall(X,fiveletters2(X),L),length(L,N).
```

4.3.1 What is the value of N computed by this query? 72

4.3.2 How many *different* answers are contained in L in the result of this query? 72

5. Questions on Constraint Programming (10 marks total)

Are the following statements true or false? Circle **true** or **false** only if you know the answer. Do *not* make a blind guess if you do not know the answer. Giving *more than one* wrong answer will *reduce* your marks according to the formula: marks = right - (wrong - 1).
For example, 5 right 2 wrong = 5 - 1 = 4 marks.
In the questions, CLP stands for “constraint logic program” or “constraint logic programming”.

true		In principle, constraint logic problems could also be solved by using Lisp.
true		CLP can be much more efficient than Prolog backtracking.
true		Prolog’s backtracking mechanism is useful for a CLP solver.
	false	CLP is a more general problem-solving method than logic programming.
true		A variable domain in CLPFD must be a range of consecutive integers.
	false	A problem with two contradictory constraints can still have a solution.
	false	Both Prolog and CLP can handle constraints involving uninitialized variables.
	false	All variables in a CLP have the same domain.
true		A problem with no constraints on a variable can have a solution.
	false	Arc consistency uses constraints over three variables.