```
-----------------------------------------------------------------------------------
Logic in Computer Science, November 9th, 2021. Time: 1h30min. No books or lecture
notes allowed.
-----------------------------------------------------------------------------------
-Insert your answers on the dotted lines ... below, and only there.
-Do NOT modify the problems or the @nota lines.
-When finished, upload this file with the same name: exam.txt
-Use the text symbols:     &     v    -       ->          |=        A       E
   for                    AND   OR   NOT   IMPLIES   "SATISFIES"  FORALL  EXISTS
etc.,  like in:
   I  |=  p & (q v -r)     (the interpretation I satisfies the formula p & (q v -
r)  ).
   You can write subindices using "_". For example write x_i to denote x-sub-i.
-----------------------------------------------------------------------------------


Problem 1. (3 points).


1a)
Let F and G be propositional formulas such that F is satisfiable and F -> G is
also satisfiable.
Is it true that G is satisfiable? Prove it using only the definitions of
propositional logic.

...

No. This is false.
Counter example: let F be p and let G be p & -p.
Then F is satisfiable with the model I such that I(p)=1.
And F-> G is also satisfiable, with the model I such that I(p)=0.
But p & -p is unsatisfiable.


1b)
Let F and G be propositional formulas such that F is a tautology.
Is it true that  F & G  is logically equivalent to G?
Prove it using only the definitions of propositional logic.

...

Yes. This is true.
F & G   is logically equivalent to  G                      iff   by def of
logical equivalence
F & G   has the same models as  G                          iff   by def of model
forall I,   I |= F & G   iff  I |= G                       iff   by def of |=
forall I,   eval_I( F & G ) =  eval_I( G )                 iff   by def eval AND
forall I,   min( eval_I( F ), eval_I(G )  ) =  eval_I( G )  iff   since F is a
tautology, we have eval_I(F)=1 for all I (see *)
forall I,   min( 1, eval_I( G )  ) =  eval_I( G )          iff   since eval
always returns either 0 or 1
forall I,           eval_I( G )    =  eval_I( G )          iff   true

*:
F is a tautology                iff  by def of tautology
forall I  we have I |= F        iff  by def of |=
forall I  we have eval_I(F)=1




-----------------------------------------------------------------------------------

Problem 2. (3 points).
```

```
A 1in3-constraint is an expression of the form  1in3( lit1, lit2, lit3 ) where
lit1,lit2 and lit3 are literals.
An interpretation I satisfies  1in3( lit1, lit2, lit3 )  if it satisfies EXACTLY
ONE of lit1, lit2 and lit3.
The 1in3-SAT problem is the problem of deciding the satisfiability of a
conjunction (AND) of 1in3-constraints.
For example,
1in3(x,y,z)  &  1in3(-x,-y, z) &  1in3(-x,y,-z)   is satisfiable (if I(x)=1,
I(y)=0, I(z)=0 then I is a model)
but
1in3(x,y,z)  &  1in3(-x,-y,-z)   is unsatisfiable.


2a) Is 1in3-SAT in NP? Explain in a few words why.

...
Yes it is in NP, because we can check any solution (a model) in linear time.


2b) Let C be a normal 3-SAT clause  l1 v l2 v l3, where l1,l2,l3 are literals over
variables x,y,z.
    Let F be:   1in3(-l1,a,b)  &  1in3(l2,b,c)  &  1in3(-l3,c,d)   (here a,b,c,d
are variables).
    Check for each one of the 7 possible models I of C that then F has a model I'
such
    that I' "extends" I, that is I(x)=I'(x), I(y)=I'(y), I(z)=I'(z).

    Similarly, check that for the (unique) I that is NOT a model of C, there is no
model I' of F extending I
    (and therefore every model I' of F extends a model I of C).

...

The seven models I of C are as follows. In each case we show how to extend it to
an I' that is a model of F:
l1 l2 l3     abcd
 0  0  1     0010
 0  1  0     0000
 0  1  1     0001
 1  0  0     0100
 1  0  1     0101
 1  1  0     1000
 1  1  1     1001

If I is not a model of C, then we have I(l1)=I(l2)=I(l3)=0 and by the
first and last constraints of F, abcd must all be false in I',
contradicting the second constraint, so there is no model I' of F.



2c) Is 1in3-SAT NP-complete? Explain very briefly why. Hint: use 2a) and 2b).

Yes, because by 2a it is in NP, and it is also NP-hard because by 2b) we can
express any 3-SAT
problem S as a 1in3-SAT problem F.

A few more details: for each clause l1 v l2 v l3 in S, we introduce new variables
a,b,c,d
and add 1in3(-l1,a,b) & 1in3(b,l2,c) & 1in3(c,d,-l3) to F.
Then by 2b) S is satisfiable iff F is satisfiable:
by 2b) any model I of S can be extended to a model I' of F, and for any model I'
of F we
can find a model of S by just "forgetting" about the new variables.
```

```
------------------------------------------------------------------------------------------

Problem 3. (3 points).
For each one of the following problems, show that it is polynomial by expressing
it as (or
reducing it to) a polynomial version of SAT.  Be very brief: just give the needed
SAT variables and
clauses and say which polynomial SAT problem it is.  If there is no such
reduction, just write: "Not possible".

3a) 2-coloring: given an undirected graph G and 2 colors,
can we assign a color to each node of G such that adyacent nodes get different
colors?

...
Variables: x_ic meaning "node i of G has color c".
Clauses: for each node i of G,    one clause:   x_i1 v  x_i2
         for each edge i-j of G, two clauses:  -x_i1 v -x_j1  and     -x_i2 v -x_j2
This is 2-SAT which is polynomial.

Another solution: Variables: x_i meaning "node i of G has color 1" (if it is
false, it has the other color).
                  Clauses:   for each edge i-j of G, two clauses:  x_i v x_j  and
-x_i v -x_j.

3b) 3-coloring.

...
Not possible.



3c) Amazon. Assume
    M is a list of Amazon products we MUST buy.
    P is a list of pairs (p,p') of products that are incompatible: we cannot buy p
and also p'.
    R is a list of rules of the form "S needs p", indicating that, if we buy
          all products in the set of products S, then we must also buy the product
p.
    Given M,P,R, can we buy a set of products satisfying the requirements of M,P,R?

...
Variables: x_p meaning "product p is bought".
Clauses: for each p in M,                         one clause:   x_p
         for each (p,p') in P,                    one clause:  -x_p  v  -x_p'
         for each rule  {p1...pn} needs p  in R,  one clause:  -x_p1 v...v -x_pn v
p.
This is Horn-SAT which is polynomial.



------------------------------------------------------------------------------------------

Problem 4. (1 point ).

4) UNIQUE-SAT is the problem of determining whether a given set of clauses S has
exactly one model.
Explain very briefly how you would use a SAT solver to decide UNIQUE-SAT.

...
We need at most two calls to the SAT solver:
Call the solver with input S:
 -if unsatisfiable, output "no".
 -if   satisfiable with model I,
     Add to S the clause forbidding just that model I, and call the solver again:
```

```
       -if satisfiable,   output "no".
       -if unsatisfiable, output "yes".

The clause forbidding I has the form  x_1 v...v x_n  v  -y_1 v...v -y_m   where
the x_i are
all variables x_i with I(x_i)=0, and the y_j are the variables with I(y_j)=1.
-----------------------------------------------------------------------------------------------
```