MASTER IN ARTIFICIAL INTELLIGENCE (UDC - USC - UVigo)

REASONING AND PLANNING exam. June 17th, 2025

Last Name: ____

First Name: ____

INSTRUCTIONS This exam covers units 1-6 and is weighted with a maximum of **42 points (pt) from a total of 100 pt** in the whole course (Unit 7 is not covered in the exam and weights 8 pt). For the test, use the original statement sheet and avoid corrections or unclear marking (ask for a new blank sheet if needed). **Completion time = 2 hours**.

— EXAM —

Exercise 1 (20pt). Each question has at least one correct answer and its total score depends on whether you check: some incorrect answer = **-3pt**; all the correct answers = **5pt**; only correct answers, but not all = **3pt**; leaving blank = **0pt**. A total negative score in Exercise 1 counts as 0pt in the rest of the exam.

1.1) Mark those formulas below that are equivalent to $p \rightarrow q \vee \neg r$ in classical propositional logic:

- [x] $p \wedge r \rightarrow q$
- [ ] $\neg p \vee \neg q \vee \neg r$
- [x] $r \wedge \neg q \rightarrow \neg p$
- [x] $p \rightarrow (r \rightarrow q)$

1.2) Given the positive logic program P with rules `p :- q, r.    r :- p.    q :- q.    s :- p, r.` mark the correct statements about the direct consequences operator T_P.

- [x] $T_P(\{\mathtt{p},\mathtt{q},\mathtt{r}\}) = \{\mathtt{p},\mathtt{q},\mathtt{r},\mathtt{s}\}$
- [x] $T_P(\{\mathtt{p},\mathtt{q}\}) = \{\mathtt{q},\mathtt{r}\}$
- [ ] $T_P(\emptyset) = \{\mathtt{q}\}$
- [ ] $T_P(\emptyset) = \{\mathtt{p},\mathtt{q},\mathtt{r},\mathtt{s}\}$

1.3) Given the following logic program `a :- not b.    b :- not c.    c :- d, not a.`

- [x] the reduct with respect to $\{\mathtt{a},\mathtt{b}\}$ is the program `b.`
- [ ] the reduct with respect to $\{\mathtt{d}\}$ is the program `a.    b.    c.`
- [x] the reduct with respect to $\emptyset$ is the program `a.    b.    c :- d.`
- [x] the reduct with respect to $\{\mathtt{a},\mathtt{b},\mathtt{c}\}$ is the program $\boxed{}$
- [ ] the reduct with respect to $\{\mathtt{c}\}$ is the program `a :- not b.    c :- d, not a.`

1.4) The rule `a :- not b.` corresponds to the formula $\neg b \rightarrow a$ in the logic of Here-and-There (HT). In classical logic, this formula is equivalent to $\neg a \rightarrow b$, but in HT they have different models. Mark those HT interpretations that are HT models of $\neg b \rightarrow a$ but not of $\neg a \rightarrow b$.

- [ ] $H = \{b\}$, $T = \{b\}$
- [x] $H = \emptyset$, $T = \{b\}$
- [ ] $H = \{a,b\}$, $T = \{a\}$
- [ ] $H = \{a\}$, $T = \{a,b\}$

Explanations for the test

1.1) The formula $p \to q \; \vee \neg r$ is equivalent to $\neg p \vee q \vee \neg r$

- $p \wedge r \to q \equiv \neg(p \wedge r) \vee q \equiv \neg p \vee \neg r \vee q$ so the formula is equivalent
- $\neg p \vee \neg q \vee \neg r$ is not equivalent: the interpretation $I = \{p, q, r\}$ satisfies the original but makes $\neg p \vee \neg q \vee \neg r$ false
- $r \wedge \neg q \to \neg p \equiv \neg(r \; \wedge \neg q) \vee \neg p \equiv \neg r \vee \neg\neg q \vee \neg p$ so the formula is equivalent
- $p \to (r \to q) \equiv \neg p \vee (\neg r \vee q) \equiv \neg p \vee \neg r \vee q$ so, again, the formula is equivalent

1.4) The first case $H = T = \{b\}$ is a model of both formulas: it satisfies $\neg b \to a$ because the antecedent $\neg b$ is false since $b \in T$; it also satisfies $\neg a \to b$ because the consequent $b \in H$ is “provedïn H.

The second case is a model of $\neg b \to a$ because the antecedent $\neg b$ is false, but is not a model of $\neg a \to b$ because $\neg a$ holds $(a \notin T)$ whereas the consequent does not hold $b \notin H$.

The third case is not a valid HT interpretation, since $H \not\subseteq T$.

The fourth case is a model of both formulas: it satisfies $\neg b \to a$ becasue the antecedent $\neg b$ is false $(b \in T)$ and also $\neg a \to b$ because, again, the antecedent is false $(a \in T)$.

Exercise 2 (10pt). A lottery ticket in Spain consists of 5 digits, covering the interval from 00000 to 99999. Write an ASP program that generates one answer set per each lottery number that contains a strictly increasing sequence of (different) digits. Use predicate `ticket(N,D)` to represent that the `N`-th digit in the ticket is `D`.

```
% Solution using auxiliary predicates
digit(0..9).
position(1..5).
1 {ticket(N,D):digit(D)} 1 :- position(N).
:- ticket(N,D), ticket(N+1,D'), D'<=D.
#show ticket/2.
```

Exercise 3 (8pt). A traffic light changes from green to red when a pedestrian pushes a button. The light returns from red to green when an internal clock sends the signal `release`. Model this system in `telingo` using the fluent `light` whose values can be `red` or `green` and the actions `push` (the button), `release` (when the clock sends the signal) and `wait` (that has no effect). Include the inertia law so that the light stays unchanged unless there is evidence on the contrary.

```
#program initial.
h(light,green).
action(release;push;wait).

#program dynamic.
h(light,red) :- o(push), 'h(light,green).
h(light,green) :- o(release), 'h(light,red).
1 {o(A): _action(A)} 1.
% Inertia
h(F,V) :- 'h(F,V), not c(F).
c(F) :- h(F,V), 'h(F,W), V!=W.

#program initial.  % Example of execution: push, wait 2 situations and release
&tel{  &true ;>  o(push) ;> o(wait) ;> o(wait) ;>  o(release)   }.
```

Exercise 5 (4pt). Write a formula in Description Logic (DL) that describes the set of actors (*Actor*) that have always acted (*acted*) in Spanish movies but have also acted in at least one terror movie (*Terror*).

$$Actor \sqcap \forall acted.Spanish \sqcap \exists acted.Terror$$