Question 1 (30%)

You have two predicates p and q. Express the following situations in first order logic:

(a) there is no p that is not also q.

> **Solution:** $\forall x.px \rightarrow qx$

(b) there is exactly one p.

> **Solution:** $\exists x.px \wedge \forall y.py \rightarrow y = x$

(c) there are exactly two ps.

> **Solution:** $\exists x \exists y.px \wedge py \wedge \neg(x = y) \wedge \forall z.pz \rightarrow z = x \vee z = y$

(d) there is at most one p.

> **Solution:** $(\neg \exists x.px) \vee \exists x.px \wedge \forall y.py \rightarrow y = x$

Question 2 (20%)

Render the following sentence in first order logic.

(1) Every farmer who owns a donkey beats it.

If you cannot come up with a satisfactory formula, briefly comment on why your efforts have failed.

> **Solution:** One possible attempt
>
> $$\forall x.farmer'x \wedge (\exists y.donkey'y \wedge owns'yx) \rightarrow beats'yx$$
>
> fails because the last occurrence of y is left unbound.
>
> Another one
>
> $$\forall x \forall y.farmer'x \wedge donkey'y \wedge owns'yx \rightarrow beats'yx$$
>
> is fine with respect to binding, but implies that expressions like *a donkey* can be interpreted as involving universal quantification. We are left with no explanation why we do not have a universally quantified reading for
>
> (2) A donkey sleeps.

Question 3 (50%)

Specify a lexicon for all the items in the following sentences and drive their meaning specifying their order of combination, syntactic categories, semantic interpretations and semantic types in each step:

Solution: Lexicon:

Let G be an abbreviation for $(S/(S\backslash NP))$

$$\begin{aligned}
&donkey := & N: & \quad \lambda x.donkey'x :: & \langle e,t\rangle & \quad (1)\\
&student := & N: & \quad \lambda x.student'x :: & \langle e,t\rangle & \quad (2)\\
&John := & S/(S\backslash NP): & \quad \lambda p.pjohn' :: & \langle\langle e,t\rangle,t\rangle & \quad (3)\\
&sleeps := & S\backslash NP: & \quad \lambda x.sleeps'x :: & \langle e,t\rangle & \quad (4)\\
&walk(s) := & S\backslash NP: & \quad \lambda x.walk'x :: & \langle e,t\rangle & \quad (5)\\
&lazy := & N/N: & \quad \lambda p\lambda x.p'x \wedge lazy'x :: & \langle\langle e,t\rangle,\langle e,t\rangle\rangle & \quad (6)\\
&slowly := & S\backslash NP\backslash(S\backslash NP): & \quad \lambda p\lambda x.slowly'px :: & \langle\langle e,t\rangle,\langle e,t\rangle\rangle & \quad (7)\\
&and := & G\backslash G/G: & \quad \lambda p\lambda q\lambda r.pr \wedge qr :: & \langle\langle\langle e,t\rangle,t\rangle,\langle\langle\langle e,t\rangle,t\rangle,\langle\langle e,t\rangle,t\rangle\rangle\rangle & \quad (8)\\
&every := & S/(S\backslash NP)/N: & \quad \lambda p\lambda q\forall x.px \rightarrow qx :: & \langle\langle e,t\rangle,\langle\langle e,t\rangle,t\rangle\rangle & \quad (9)\\
&a := & S/(S\backslash NP)/N: & \quad \lambda p\lambda q\exists x.px \wedge qx :: & \langle\langle e,t\rangle,\langle\langle e,t\rangle,t\rangle\rangle & \quad (10)
\end{aligned}$$

(a) Every donkey sleeps.

Solution: D1: apply (9) to (1)
D2: apply (the result of) D1 to (4)

(b) John walks slowly.

Solution: D1: apply (7) to (5)
D2: apply (3) to D1

(c) A lazy donkey walks.

Solution: D1: apply (6) to (1)
D2: apply (10) to D1
D3: apply D2 to (5)

(d) Every student and a lazy donkey walk.

Solution: D1: apply (6) to (1)
D2: apply (10) to D1
D3: apply (9) to (2)
D4: apply (8) to D2
D5: apply D4 to D3
D6: apply D5 to (5)