Knowledge & Data Final Exam:

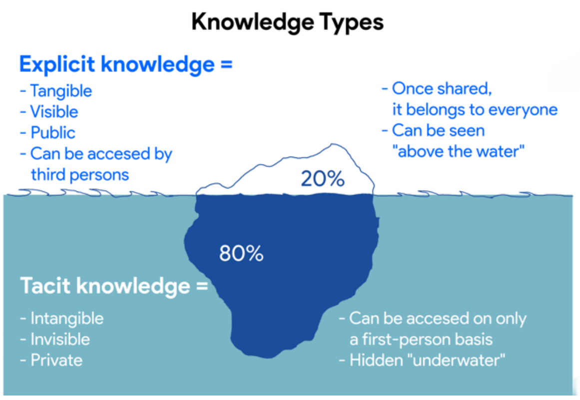
Module 1, Quiz 1:

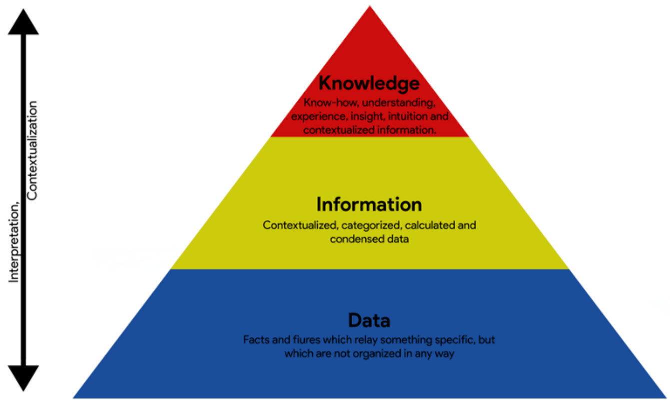
Question 1:

Which statement is true?

- Knowledge is always tacit; information is always explicit.
- Data is information that has been retained with an understanding of the significance of that information.
- Formal knowledge is an alternative for Machine Learning in most AI tasks.
- Explicit knowledge allows for predictable inferencing, thus increasing (re)usability of data.

Tacit knowledge (also known as implicit knowledge) is the knowledge that a person retains in their mind. Explicit knowledge (also known as formal knowledge) is knowledge that has been formalized, codified and stored. Formal Knowledge is necessary to efficiently interpret and reuse data, because can help us to interpret and reuse data and make it reusable for other purposes.





Possible answer	*Correct/ Wrong*
"Knowledge is always tacit; information is always explicit."	Wrong, because tacit and explicit are both types of knowledges.
"Data is information that has been retained with an understanding of the significance of that information."	Wrong, because data is information without context, categorization, calculation or condensation.
"Formal knowledge is an alternative for Machine Learning in most AI tasks."	Wrong, because formal knowledge is a type of data, not specifically relating to AI.
"Explicit knowledge allows for predictable inferencing, thus increasing (re)usability of data."	Correct, because explicit knowledge, being formalized and codified, enables and predictable inferencing, ultimately enhancing the efficiency and reusability of data.

Question 2:
Consider these two statements:
1. Tacit knowledge is knowledge that someone has in their mind.
2. Explicit knowledge is knowledge about how to process data (for example through Machine Learning)

- both 1 and 2 are correct.
- 1 is correct; 2 is incorrect.
- 1 is incorrect; 2 is correct.
- both 1 and 2 are incorrect.

Possible answer	***Correct/ Wrong***
both 1 and 2 are correct.	Wrong, because explicit knowledge is knowledge that is formalized, not knowledge about how to process data (2 is incorrect).
1 is correct; 2 is incorrect.	Correct, tacit knowledge is the knowledge that a person retains in their mind (1 is correct).
1 is incorrect; 2 is correct.	Wrong, because 1 is correct and 2 is incorrect.
both 1 and 2 are incorrect.	Wrong, because 1 is correct and 2 is incorrect.

Question 3:
Consider the following formula, which is in Infix syntax:
¬(A → (¬B ∨ C)) v (¬D v ¬E))

What is an equivalent representation in Prefix syntax?

- ☐ ¬(v(→ (A,v(¬(B),C)),v(¬(D),¬(E)))
- ☐ ¬(v(→(v(v(v(¬(¬(¬(A, B,C,D,E))))))
- ☐ ((((E)¬,(D)¬)v,((C,(B)¬)v,)→)v)¬
- ☐ ¬(v(→ (E,v(¬(B),D)),v(¬(C),¬(A)))
- ☐ v(¬(v(A,→(¬(B),C)),v(¬(D),¬(E))))

Step	***Explanation***
→ ¬(v(A -> (¬B v C)), (¬D v ¬E))	Add the disjunction between ABC and DE, negation is already good for prefix syntax.
→ ¬(v(->(A, (¬B v C)), (¬D v ¬E)))	Add the A implies BC to the formula in prefix syntax.
→ ¬(v(->(A, v(¬B, C)), (¬D v ¬E)))	Add the disjunction between B and C.
→ ¬(v(->(A, v(¬(B), C)), (¬(D) v –(E))))	Isolate all negations.
→ ¬(v(->(A, v(¬(B), C)), v(¬(D), –(E))))	Ads the disjunction between D and E.

Question 4:

Consider the following propositional formula: $p \rightarrow (q \vee \neg r)$. Which of the following interpretations is NOT a model?

- I(p)=True, I(q)=False, I(r)=True.
- I(p)=True, I(q)=False, I(r)=False.
- I(p)=False, I(q)=False, I(r)=False.
- It's a tautology, so all are models.
- I(p)=True, I(q)=True, I(r)=False.

p	q	r	¬r	q v ¬r	p→(q v ¬r)
T	T	T	F	T	T
T	T	F	T	T	T
T	F	T	F	F	F
T	F	F	T	T	T
F	T	T	F	T	T
F	T	F	T	T	T
F	F	T	F	F	T
F	F	F	T	T	T

There is only one interpretation of p -> (q v ¬r) that is not a model, p = True, q = False, and r = True.

Question 5:

Let KB be a knowledge base, and F a formula. Which of the following answers about entailment is true?

- F is entailed by KB if, and only, if every model of KB is also a model of F.
- F is entailed by KB if, and only, if there is a model of KB that is also a model of F.
- F is entailed if, and only, if every interpretation of KB is also a model of KB and thus also of F.
- If F has no model, it is always entailed by KB.
- If the knowledge base has no models, nothing is entailed.

Possible answer	***Correct/ Wrong***
F is entailed by KB if, and only, if every model of KB is also a model of F.	Correct, because formally, F is said to be entailed by KB, denoted as KB ⊨ F, if and only if, in every interpretation or model where KB is true, F is also true.
F is entailed by KB if, and only, if there is a model of KB that is also a model of F.	Wrong, because if there is a model of KB that is also a model of F, there could be a model of KB that is not a model of F.
F is entailed if, and only, if every interpretation of KB is also a model of KB and thus also of F.	Wrong, because 1 is correct and 2 is incorrect.
If F has no model, it is always entailed by KB	Wrong, if there is not a model of F, and there is not a model of KB, F is entailed by KB. If there is a model of KB and not of F, F is not entailed by KB.
If the knowledge base has no models, nothing is entailed	Wrong, if the knowledge base has no models, and F has no models, everything is entailed.

Question 6:
In Propositional Logic, which of the following formulas is entailed by the knowledge base containing two formulas p v (¬q & r) and (¬r v q)?

- p
- q
- None of the above
- r

p	q	r	¬r	¬q	¬q & r	p v (¬q & r)	¬r v q	(p v (¬q & ¬r)) & (¬r v q)
T	T	T	F	F	F	T	T	T
T	T	F	T	F	F	T	T	T
T	F	T	F	T	T	T	F	F
T	F	F	T	T	F	T	T	T
F	T	T	F	F	F	F	T	F
F	T	F	T	F	F	F	T	F
F	F	T	F	T	T	T	F	F
F	F	F	T	T	F	F	T	F

Possible answer	***Correct/ Wrong***
p	Correct, because for every model that the KB is true, p is true.
q	Wrong, because for the model p=True, q=False, r=False, the KB is true, but q is False.
None of the above	Wrong, because for every model that the KB is true, p is true.
r	Wrong, because for the model p=True, q=False, r=False, the KB is true, but r is False.

Question 7:
In Propositional Logics check whether the following entailment holds:

p v (¬q & r) ⊨ (q v ¬r) → p

- Yes.
- No.
- Cannot be decided.
- It's a Tautology.

p	q	r	¬r	¬q	¬q & r	p v (¬q & r)	¬r v q	¬r v q → p
T	T	T	F	F	F	T	T	T
T	T	F	T	F	F	T	T	T
T	F	T	F	T	T	T	F	T
T	F	F	T	T	F	T	T	T
F	T	T	F	F	F	F	T	F
F	T	F	T	F	F	F	T	F
F	F	T	F	T	T	T	F	T
F	F	F	T	T	F	F	T	F

For each model of p v (¬q & r), the formula (q v ¬r) → p is also true, so the entailment holds.

Question 8:
In Propositional Logics, which of the following entailments is true, and which is false?

(1) $p \rightarrow (p \rightarrow (p \rightarrow (q \rightarrow r))) \vDash p \rightarrow (r \rightarrow q)$
(2) $p \rightarrow (\neg q \vee r), \neg r \vDash \neg q \rightarrow \neg p$
(3) $q \vDash p \rightarrow q$

- (a) is false, (b) is true and (c) is true
- (a) is true, (b) is true and (c) is true
- (a) is false, (b) is false and (c) is true
- (a) is true, (b) is false and (c) is true

a:

p	q	r	q → r	p → (q → r)	p → (p → (q → r))	p → (p → (p → (q → r)))	r → q	p → (r → q)
T	T	T	T	T	T	T	T	T
T	T	F	F	F	F	F	T	T
T	F	T	T	T	T	T	F	F
T	F	F	T	T	T	T	T	T
F	T	T	T	T	T	T	T	T
F	T	F	F	T	T	T	T	T
F	F	T	T	T	T	T	F	T
F	F	F	T	T	T	T	T	T

b:

p	q	r	¬q	¬q v r	p → (¬q v r)	(p → (¬q v r)) & ¬r	¬p	¬q -> ¬p
T	T	T	F	T	T	F	F	T
T	T	F	F	F	F	F	F	T
T	F	T	T	T	T	F	F	F
T	F	F	T	T	T	T	F	F
F	T	T	F	T	T	F	T	T
F	T	F	F	F	T	T	T	T
F	F	T	T	T	T	F	T	T
F	F	F	T	T	T	T	T	T

c:

p	q	r	p → q
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	F
F	T	T	T
F	T	F	T
F	F	T	T
F	F	F	T

For each model of the formula before the entailment sign, the formula after the entailment sign should also be a model for the entailment to hold. Counterexamples for each entailment are:
a: If p=T, q=F and r=F, $p \rightarrow (p \rightarrow (p \rightarrow (q \rightarrow r)))$ is True, so this is a model. But $p \rightarrow (r \rightarrow q)$ is False. (entailment doesn't hold)
b: If p=T, q=T and r=F, q -> ¬p is True, so this is a model. But ¬q -> ¬p is False. (entailment doesn't hold)
c: For each model of q, $p \rightarrow q$ is also true, so the entailment holds.
(a) is false, (b) is false and (c) is true is the correct answer.

Module 1, Quiz 2:

Question 1:

Which of these four principles/proposals of the Semantic Web is *incorrect*?

1. Give things a name.
2. Names are addresses on the Web.
3. Information about things is stored in tables.
4. Make the meaning of things explicit.

- 1 is incorrect. It should be: "Define document templates".
- 2 is incorrect. It should be "Names are hashed identifiers".!
- 3 is incorrect. It should be: "Relations (triples) form a graph between things".
- 4 is incorrect. It should be: "Store meaning in metadata tables".

To achieve a data landscape as a network of data access points, we must solve two problems:

• Integrate (semantically) heterogenous information
• Integrate (physically) distributed information

The solution of this is found in Linked (open) Data, with Knowledge Graphs as the main vehicle. Some aspects of this semantic web of data are that websites publish their information in a machine-readable format, the data published by different sources is linked, enough domain knowledge is available to machines to make use of the information, machines can find and combine published information in appropriate ways to answer the user's information needs.

This can be achieved by 4 proposals:

1. Give all things (that you want to/ can talk about) a name.
2. The names are addresses on the Web.
3. Relations form a graph between things.
4. Make explicit the meaning (semantics) of things.

The first 3 proposals result in a global graph of linked data. Proposal 4 results in the graph having an underlying meaning by assigning types to things and relations,

Possible answer	*Correct/ Wrong*
1 is incorrect. It should be: "Define document templates".	Wrong, because 1 is indeed give things a name.
2 is incorrect. It should be "Names are hashed identifiers".!	Wrong, because 2 is indeed make names addresses on the web.
3 is incorrect. It should be: "Relations (triples) form a graph between things".	Correct, because 3 is relations form a graph between things.
4 is incorrect. It should be: "Store meaning in metadata tables".	Wrong, because 4 is make the meaning of things explicit.

Question 3:

Consider the following formula, which is in Prefix syntax:

&(v(A ,v(¬(B),C)) , ¬(&(¬(D) ,¬(E))))

What is an equivalent representation in Infix syntax?

- (A v (¬B v C)) & ¬(¬D & ¬E)
- (A & (¬B v C)) v ¬(¬D & ¬E)
- (A v (¬B & C)) → ¬(¬D v ¬E)
- (A v (B v C)) & ¬(D & E)

Step	***Explanation***
→(v(A ,v(¬(B),C)) & ¬(&(¬(D) ,¬(E))))	Move the disjunctions between ABC and DE inward.
→(A v (¬(B) v C)) & ¬(&(¬(D) ,¬(E))))	Move the disjunctions between A and BC and B and C inward.
→(A v (¬(B) v C)) & ¬((¬(D) & ¬(E)))	Move the conjunction between D and E inward.
→(A v (¬B v C)) & ¬(¬D & ¬E)	Move the negations inward

So the correct answer is (A v (¬B v C)) & ¬(¬D & ¬E).

Question 4:

In Propositional Logic, which of the following formulas is entailed by the knowledge base containing two formulas q v (¬p & r) and (¬q → p)?

- q
- p
- both p and q are entailed
- neither p nor q are entailed

p	q	r	¬p	¬q	¬p & r	q v (¬p & r)	¬q → p	q v (¬p & r) and (¬q → p)
T	T	T	F	F	F	T	T	T
T	T	F	F	F	F	T	T	T
T	F	T	F	T	F	F	T	F
T	F	F	F	T	F	F	T	F
F	T	T	T	F	T	T	T	T
F	T	F	T	F	F	T	T	T
F	F	T	T	T	T	T	F	F
F	F	F	T	T	F	F	F	F

As we can see, q is entailed by the knowledge base.

Module 1, Quiz 3:

Question 1:

Consider the following formula, which is in Infix syntax:

¬(A → (¬B ∨ C)) & ¬(¬D v ¬E)

What is an equivalent representation in Prefix syntax?

- &(¬(→(A,v(¬(B),C))),¬(v(¬(D),¬(E))))
- ¬(&(→(A,v(¬(B),C))),¬(v(¬(D),¬(E))))
- &(¬(→(C,v(¬(B),A))),¬(v(¬(D),¬(E))))
- &(¬(v(A,→(¬(B),C))),¬(→(¬(D),¬(E))))

Step	*Explanation*
→ &(¬(A → (¬B ∨ C)), ¬(¬D v ¬E))	Moved the conjunction outward.
→ &(¬(→(A, (¬B ∨ C)), ¬(¬D v ¬E))	Moved the implication between A and BC outward.
→ &(¬(→(A, (¬(B) ∨ C)), ¬(¬(D) v ¬(E)))	Isolate negations B, D and E.
→ &(¬(→(A, (v(¬(B), C))), ¬(¬(D) v ¬(E)))	Move disjunction between B and C outward.
→ &(¬(→(A, v(¬(B), C))), ¬(v(¬(D), ¬(E))))	Move disjunction between D and E outward

So &(¬(→(A,v(¬(B),C))),¬(v(¬(D),¬(E)))) is the correct answer.

Question 2:

Consider the following propositional formula: p → (¬q & r). Which of the following interpretations is a model?

- I(p)=True, I(q)=False, I(r)=True
- I(p)=True, I(q)=False, I(r)=False
- I(p)=True, I(Q)True, I(R)=True
- I(p)=True, I(Q)=True, I(R)=False
- There is no model.

p	q	r	¬q	¬q & r	p → (¬q & r)
T	T	T	F	F	F
T	T	F	F	F	F
T	F	T	T	T	T
T	F	F	T	F	F
F	T	T	F	F	T
F	T	F	F	F	T
F	F	T	T	T	T
F	F	F	T	F	T

If p is True, the only model is q = False, r=True, so I(p)=True, I(q)=False, I(r)=True is correct.

Question 3:
In Propositional Logic, which of the following formulas is entailed by the knowledge base containing two formulas p → (¬q & r) and (¬r v q)?

- neither p nor q are entailed
- p
- q
- both p and q are entailed

p	q	r	¬r	¬q	¬q & r	p → (¬q & r)	¬r v q	p → (¬q & r) & (¬r v q)
T	T	T	F	F	F	F	T	F
T	T	F	T	F	F	F	T	F
T	F	T	F	T	T	T	F	F
T	F	F	T	T	F	F	T	F
F	T	T	F	F	F	T	T	T
F	T	F	T	F	F	T	T	T
F	F	T	F	T	T	T	F	F
F	F	F	T	T	F	T	T	T

Possible answer	*Correct/ Wrong*
p	Wrong, because for the model p=False, q=True, r=True, the KB is true, but p is False.
q	Wrong, because for the model p=False, q=False, r=False, the KB is true, but q is False.
neither p nor q are entailed	Correct, because neither are entailed.
both q and p are entailed	Wrong, because neither are entailed.

Question 4:
In Propositional Logics check whether the following entailment holds: p v (¬r & q) ⊨ (q v ¬r) → p

- No.
- Yes.
- Cannot be decided.
- It's a tautology.

p	q	r	¬r	¬q	¬r & q	p v (¬r & q)	q v ¬r	q v ¬r → p
T	T	T	F	F	T	T	T	T
T	T	F	T	F	T	T	T	T
T	F	T	F	T	F	T	F	T
T	F	F	T	T	T	T	T	T
F	T	T	F	F	T	T	T	F
F	T	F	T	F	T	T	T	F
F	F	T	F	T	F	F	F	T
F	F	F	T	T	T	F	T	F

For each model of p v (¬r & 1), the formula (¬r v q) → p is not true, so the entailment doesn't hold. A counterexample is p=False, q=False, r=True where p v (¬r & q) is true but q v ¬r → p is false.

Module 1, Quiz 4:

Question 1:

In Propositional Logics check whether the following entailment holds:

p → (q → r) ⊨ p → (r → q)

- Yes.
- No.
- Entailment is not defined for Propositional Logic.
- There is not enough information available.

p	q	r	q → r	p → (q → r)	r → q	p → (q → r)
T	T	T	T	T	T	T
T	T	F	F	F	T	T
T	F	T	T	T	F	F
T	F	F	T	T	T	T
F	T	T	T	T	T	T
F	T	F	F	T	T	T
F	F	T	T	T	F	T
F	F	F	T	T	T	T

For each model of p → (q → r) the formula p → (q → r) is not also true. A counterexample; for the model where p=True, q=False, r=True, the formula p → (q → r) is false. The entailment does not hold.

Module 1, Worksheet:

Example question 1 (Entailment in Arithmetic):

It is easy for humans to see that a formula 3*x < 12 + 6*y and y = 1 implies (or entails) that x < 10.

Give the formal argument why this is the case according to the model theoretic semantics given in the lecture (and thus NOT using your calculus skills!). 4 rows should suffice. Use the words assignment or interpretation and model.

Truth is defined in terms of an assignment for variables. Let V be the set of variables, then I v: V → N is an assignment; a function that assigns natural numbers to each variable in V. Function I v is a model of a formula f if I v (f) is true. Formula f entails g , f ⊨ g , iff g is true for all models of f .

Answer:

(1) x < 10 is entailed by the other two formulae if h is true for all the models of f and g
(2) These models are assignments of variables x and y for which f and g are true
(3) That is, I v (3x < 12 + 6y , y = 1) and I v (x < 10) are true for all possible assignments
(4) If I v (3x < 12 + 6y , y = 1) is true, then also I v (3x < 18) is true, or I v (x < 6)
(5) Therefore, also x < 10

Example question 2 (Entailments in PL):

In Propositional Logics, which of the following entailments is true, and which is false?

(a) ¬p ∨ (q → p) ⊨ ¬p ∧ q
(b) ¬p ∨ (q → p) ⊨ ¬p ∨ q

Check the right option:

(1) (a) is true, and (b) is true
(2) (a) is true, and (b) is false
(3) (a) is false, and (b) is true
(4) (a) is false, and (b) is false

a, entailment does not hold (counterexample: when p=T and q=T ¬p ∨ (q → p) is true but ¬p ∧ q is false:

p	q	¬p	q → p	¬p ∨ (q → p)	¬p ∧ q
T	T	F	T	T	F
T	F	F	T	T	F
F	T	T	F	T	T
F	F	T	T	T	F

b, entailment does not hold (counterexample: when p=T and q=T ¬p ∨ (q → p) is true but ¬p ∨ q is false:

p	q	¬p	q → p	¬p ∨ (q → p)	¬p ∨ q
T	T	F	T	T	T
T	F	F	T	T	F
F	T	T	F	T	T
F	F	T	T	T	T

So, the correct answer is 4: (a) is false, and (b) is false.

Example question 3 (Models in Arithmetic):
In the lecture the notion of a model was not explicitly mentioned, it is explicit in the slides. A model of a knowledge base is an assignment (or interpretation) that satisfies all its axioms/formulas. Take as an example a formula x<6. An interpretation that assigns a value 3 to x satisfies the formula, and is thus a model for it. An interpretation that gives it a value 8 is not.
In propositional logic P & ¬Q, the models are those interpretations of the propositions that make the entire formula true, so only the assignment that interprets P as true and Q as false is a model. Which of the following assignments I are models for the following two formulas:
2*x + (4 + y) – z = 10 and x + y < 7,
according to the semantics of our formal system of Arithmetic?

- I(x)=3, I(y)=4,I(z)=4
- I(x)=3, I(y)=3,I(z)=4
- I(x)=3, I(y)=3,I(z)=3
- I(x)=4, I(y)=4,I(z)=3

a) 6 + 8 – 4 = 10, b) 3 + 4 < 7
a) 6 + 7 – 4 = 10, b) 3 + 3 < 7
a) 6 + 7 – 3 = 10, b) 3 + 3 < 7
a) 8 + 8 – 3 = 10, b) 4 + 4 < 7

a) 6 + 8 – 4 = 10, b) 3 + 4 < 7 is the only answer where both a and b are correct, so are models for the formulas.

Example question 4 (Formal Systems: Syntax):
The task is to design a very simple new Formal System. The only thing you can say in this language is "Oeps" and "Whow". Define the syntax of this language L. You can either give a regular expression (not introduced in this course), or an inductive definition (as we did with arithmetic and concept logic). This means, define a set of basic propositions P and an inductive definition to define your language L, i.e. the set of well-formed sentences.
You will need exactly 3 lines for this:
1) the basic propositions
2) the base-case, i.e. the most simple sentences
3) the inductive case, i.e how to construct complex sentences

1. The basic propositions are P = {“Oeps”, “Whow”}
2. The most simple sentences are p ∈ P → [p] ∈ L. This means p where p is an element of the set P, implies that the expression [p] is a part of the language L
3. The inductive case is p, q ∈ L → [p, q] ∈ L. This means that if p and q are elements in the language, the expression [p,q] is also in the language.

Example question 5 (Formal Systems: Semantics):
Let's continue with your simple formal systems about Oeps and Whow. Your friends know that if you say more "Oeps" than "Whow", you are unhappy, otherwise you are happy. Can you give a definition of the Semantics of your language, so that for any well-formed sentence, you can determine whether you are happy or unhappy.
Here you need exactly two lines:
1) Let $p \in P$, what is the interpretation I(p)?
2) If $s \in L$ is a sentence, what is the interpretation of I(s)? Make use of the way you defined the language in the previous question.

1) The interpretation of I(p) could be I (whow) = 1, I (oeps) = −1, the choice of numbers is arbitrary but I(whow) has to be higher than I(oeps).
2) The interpretation of I(s) could be I ([whoops, tail]) = −1 + I (tail), I ([wow , tail]) = 1 + I (tail) where 'tail' denotes the remainder of the sentence.

Example question 6 (Facts and reproduction):
Which of the following sentences is true (at least according to Forbes)?
1. The most time-consuming problem for data scientists is to find the right algorithm for pattern recognition.
2. Without sufficient data, tacit knowledge is useless in practices.
3. There is usually more time spend on preparing, linking and cleaning data than on building the.
   datasets in the first place.
4. Knowledge graphs are the only way to make tacit knowledge explicit.

Possible answer	***Correct/ Wrong***
The most time-consuming problem for data scientists is to find the right algorithm for pattern recognition.	Wrong, because the most time-consuming problem for data scientists is not always finding the right algorithm.
Without sufficient data, tacit knowledge is useless in practices.	Wrong, because while data and information can be valuable in decision-making and problem-solving, there are many situations where tacit knowledge plays a crucial role and can be highly useful even in the absence of extensive data. Tacit knowledge can inform judgment, decision-making, and actions, especially in complex and nuanced situations.
There is usually more time spend on preparing, linking and cleaning data than on building the. datasets in the first place.	Correct, because usually linking and cleaning the data takes most time for data scientists.
Knowledge graphs are the only way to make tacit knowledge explicit.	Wrong, there are a lot of other ways to make tacit knowledge explicit; like for example writing a book.

Module 2, Quiz 1:
Question 1:
Give the different reasons why data and knowledge reuse on the Web is difficult:

- Publishers use different syntaxes, such as XML, csv, tsf, etc.
- There are no shared identifiers between datasets.
- There is too much data to be stored on one machine, so data can never be reused.
- There is no protocol to access data from remote servers.
- Information may mean different things in different contexts.
- People may disagree, causing inconsistencies.
- Not all data is in XML.

Possible answer	***Correct/ Wrong***
Publishers use different syntaxes, such as XML, csv, tsf, etc.	Correct, because the Resource Description Framework (RDF) and other standards, while providing a common data model, do not enforce a uniform syntax. Publishers often use various syntaxes to represent their data (XML, CSV, etc.), making it challenging for systems to universally interpret and reuse the information.
There are no shared identifiers between datasets.	Correct, because this explains the absence of a standardized Uniform Resource Identifier (URI) system. Without shared identifiers, datasets may use different naming conventions or schemes, hindering effective linking and reuse of data across diverse sources
There is too much data to be stored on one machine, so data can never be reused.	Wrong, because the difficulty in data and knowledge reuse is not primarily due to the volume of data. Rather, the challenge lies in the diverse formats, lack of standardized identifiers, varying meanings in different contexts, and potential disagreements regarding interpretation and consistency, as highlighted in the correct answers.
There is no protocol to access data from remote servers.	Wrong, because the Hypertext Transfer Protocol (HTTP) is explicitly mentioned as a standard for accessing and moving data on the Web. The existence of HTTP as a protocol demonstrates that there is indeed a mechanism to access data from remote servers.
Information may mean different things in different contexts.	Correct, because this aligns with the need for a standardized data model, such as the Resource Description Framework (RDF), which provides a common structure. Different contexts or domains may interpret the same data differently, making it essential to establish a standardized representation to bridge these variations and facilitate meaningful reuse.
People may disagree, causing inconsistencies.	Correct, because this is explaining the importance of having formal semantics, as specified by the Resource Description Framework (RDF). Disagreements or differing interpretations can lead to inconsistencies in understanding and using the data. Formal semantics help in agreeing upon the meaning of data and knowledge, ensuring consistency and accuracy.
Not all data is in XML.	Wrong, because this answer, while true, is not directly related to the difficulty of data and knowledge reuse. The challenge is more about the lack of standardized data models (like RDF) and shared identifiers (URIs) rather than the specific data format (XML or others) being used.

Question 2:
Modeling a DB as a graph, Part 1: Instances, Relations and Objects. Consider the following relational databases with tables Players and Clubs::

Player	**Name**	**Age**	**Country of Birth**	**Club**
p1	Toni Kroos	28	Germany	c1
p2	Timo Werner	22	Germany	c2
p3	Frenkie de Jong	21	Netherlands	c4
p4	Virgil van Dijk	26	Netherlands	c3

Club	**Name**	**Country of Competition**
c1	Real Madrid	Spain
c2	RB Leibzig	Germany
c3	FC Liverpool	England
c4	Ajax Amsterdam	Netherlands

The ultimate goal is to represent (some of) the information as a knowledge graph as faithfully as possible. In order to do so, first determine what you want to model as entities, literals, and properties. The following lists are not complete, but which of them are correct:

- Entities: p1, c1, "RB Leibzig", Netherland, 21
- Entities: p1, c1, Netherlands, Competition, Club
- Entities: p1, c1, Spain, Club, 26
- Literals: c1, "FC Liverpool", 26, England
- Literals: "FC Liverpool", "Ajax Amsterdam", 26, "Frenkie de Jong"
- Literals: "FC Liverpool", Netherlands, c1, p2
- Properties: a, hasAge, playsInCompetition, hasLabel
- Properties: a, hasAge, Player, 26
- Properties: Club, hasAge, playsInCompetition, Netherlands

Possible answer	***Correct/ Wrong***
Entities: p1, c1, "RB Leibzig", Netherlands, 21	Wrong, because "RB Leipzig" and 21 are literals, not an entity.
Entities: p1, c1, Netherlands, Competition, Club	Correct, because these are all entities.
Entities: p1, c1, Spain, Club, 26	Wrong, because 26 is a literal, not an entity.
Literals: c1, "FC Liverpool", 26, England	Wrong, because c1 and England are entities, not literals.
Literals: "FC Liverpool", "Ajax Amsterdam", 26, "Frenkie de Jong"	Correct, because these are all literals.
Literals: "FC Liverpool", Netherlands, c1, p2	Wrong, because c1, Netherlands and p2 are entities, not literals.
Properties: a, hasAge, playsInCompetition, hasLabel	Correct, because these are all properties.
Properties: a, hasAge, Player, 26	Wrong, because 26 is a literal, not a propery.
Properties: Club, hasAge, playsInCompetition, Netherlands	Wrong, because Netherlands is an entity, not a property.

Question 3:

Modeling a DB as a graph, Part 2: Relational Data as a Graph

Consider the relational database from the previous question. Which of the following triples are faithful representations of the information in the database as a Knowledge Graph? At least 4 triples are correct, but there could be more.

- (c4 label "Ajax Amsterdam")
- (p4 label "Frenkie de Jong" hasAge 21)
- (p4 playsAtClub c4)
- (p4 "Club" c4)
- (p4 born_In Netherlands)
- (p4 a Player)
- ("c4 label Ajax Amsterdam")
- ("Ajax Amsterdam" playsIn "Netherlands")
- (c4 playsInCountry Netherlands)
- (Virgil_vav_Dijk playsAtClub FC_Liverpool)

Possible Answers	*Correct/ Wrong*
(c4 label "Ajax Amsterdam")	Correct, because c4 is Ajax Amsterdam.
(p4 label "Frenkie de Jong" hasAge 21)	Wrong, because the triple does not have correct formatting.
(p4 playsAtClub c4)	Wrong, p4 plays at club c3.
(p4 "Club" c4)	Wrong, because "Club" is not a property.
(p4 born_In Netherlands)	Correct, because p4 was born in the Netherlands.
(p4 a Player)	Correct , because p4 is of type Player.
("c4 label Ajax Amsterdam")	Wrong , because the content of the triple should not be in a string.
("Ajax Amsterdam" playsIn "Netherlands")	Wrong, because "Ajax Amsterdam" is the label, it should be c4.
(c4 playsInCountry Netherlands)	Correct , because c4 (Ajax Amsterdam) plays in the Netherlands.
(Virgil_vav_Dijk playsAtClub FC_Liverpool)	Wrong, because Virgil_vav_Dijk and FC_Liverpool are the labels, not the entities.

Question 4:
Is the following RDF turtle file syntactically correct or not?

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix ex: <http://example.org/stuff/1.0/> .

<http://rijksmuseum.nl/data/nachtwacht>
dc:title "Nachtwacht"@nl ;
ex:painted_by [
   foaf:name "Rembrandt"
   ] .
```

- Yes, this turtle file is syntactically correct
- No, dc:title is not a correct property
- No, "Nachtwacht" should not have a language tag
- No, the blank node should use { } instead of [ ] brackets

Possible answer	*Correct/ Wrong*
Yes, this turtle file is syntactically correct	Correct, because the turtle has correct syntax.
No, dc:title is not a correct property	Wrong, because dc:title is a correct property.
No, "Nachtwacht" should not have a language tag	Wrong, because “Nachtwacht” should have a language tag as it is a string.
No, the blank node should use { } instead of [ ] brackets	Wrong, because blank node should use [] brackets.

Question 5:

Is the following RDF turtle file syntactically correct or not?

```
@prefix : <http://example.org/data/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

:mycat :name "Truffel"@nl ;
       :likes :food ,
              :me .

:me foaf:name "Victor".
```

- Yes, this turtle file is syntactically correct
- No, the empty prefix declaration is not correct
- No, ":me" should be a blank node
- No, there should me more triples for ":me"

Possible answer	***Correct/ Wrong***
Yes, this turtle file is syntactically correct	Correct, because the turtle has correct syntax.
No, the empty prefix declaration is not correct	Wrong, because the empty prefix declaration is correct
No, ":me" should be a blank node	Wrong, because :me has a URI reference.
No, there should me more triples for ":me"	Wrong, because there is no need for more triples for “:me”

Question 6:

Let KG={(a,p,b),(b,q,a)} be a knowledge graph.
Which of the following interpretations is a model of KG?

- (U,I) with U={1,2}, I(a)=1,I(b)=2, I(p)={(1,2)}, I(q)={(1,1)}
- (U,I) with U={1,2}, I(a)=1,I(b)=2, I(p)={(1,2)}, I(q)={(2,2)}
- (U,I) with U={1,2}, I(a)=2,I(b)=2, I(p)={(1,2)}, I(q)={(2,2)}
- (U,I) with U={1,2}, I(a)=2,I(b)=2, I(p)={(1,2)}, I(q)={(2,1)}
- (U,I) with U={1,2}, I(a)=2,I(b)=1, I(p)={(2,1)}, I(q)={(1,2)}

Possible answer	***Correct/ Wrong***
(U,I) with U={1,2}, I(a)=1,I(b)=2, I(p)={(1,2)}, I(q)={(1,1)}	Wrong, because if (a p b) and (b q a) are in the knowledge graph, if p = {(1,2)}; q has to be {(2,1)}, not {(1,1)}, which is if it was (a q a).
(U,I) with U={1,2}, I(a)=1,I(b)=2, I(p)={(1,2)}, I(q)={(2,2)}	Wrong, because if (a p b) and (b q a) are in the knowledge graph, if p = {(1,2)}; q has to be {(2,1)}, not {(2,2)}, which is if it was (b q b).
(U,I) with U={1,2}, I(a)=2,I(b)=2, I(p)={(1,2)}, I(q)={(2,2)}	Wrong, because if (a p b) and (b q a) are in the knowledge graph, p can't be {(2,2)}; if a = 2, p has to be {(2, b)}.
(U,I) with U={1,2}, I(a)=2,I(b)=2, I(p)={(1,2)}, I(q)={(2,1)}	Wrong, because if (a p b) and (b q a) are in the knowledge graph, p can't be {(2,2)}; if a = 2, p has to be {(2, b)}.
(U,I) with U={1,2}, I(a)=2,I(b)=1, I(p)={(2,1)}, I(q)={(1,2)}	Correct, because if (a p b) and (b q a) are in the knowledge graph, a = 2, b = 1. p has to be {(2,1)} and q has to be {(1,2)}

Question 7:

Let KG = {(stefan,teaches,kad),(klaas,isa,teacher),(kad,isa,course),(carl,participates,kad),(carl,isa ,student)}

Which of the following knowledge graphs is entailed by entailed by KG in RDF?

- {(klaas,isa,teacher),(carl,isa,student),(kad,isa,course)}
- {(stefan,teaches,kad),(stefan,isa,teacher)}
- {(stefan,teaches,kad),(klaas,isa,teacher),(kad,isa,course),(carl,participates,kad),(carl,isa,student),(carl,participates,course)}
- {(carl,particpartes,course)}

A knowledge graph A is entailed by another knowledge graph B if all triples from A are found in B.

Possible answer	***Correct/ Wrong***
{(klaas,isa,teacher),(carl,isa,student),(kad,isa,course)}	Correct, this knowledge graph is entirely found in the KG.
{(stefan,teaches,kad),(stefan,isa,teacher)}	Wrong, (stefan, isa, teacher) is not in the KG.
{(stefan,teaches,kad),(klaas,isa,teacher),(kad,isa,course), (carl,participates,kad),(carl,isa,student),(carl,participates,course)}	Wrong, (carl,participates,course) is not in the KG.
{(carl,particpartes,course)}	Wrong, (carl,participates,course) is not in the KG.

Question 8:
Why is it useful to use HTTP URIs, instead of other identifiers, such as Apples App ID, or the Social Security Number of a person?

- HTTP URIs can be dereferenced, ie they can be accessed via http, and provide useful information about a resource.
- HTTP is the only way to access data from a triple-store, so it is critical that your URIs are HTTP addresses.
- Chances are very high that HTTP URIs are unique, whereas other identifiers can be the same.
- HTTP is the only way to provide information about resources that is human readable.
- HTTP uses hypertext, and is therefore far more secure than standard identifiers.
- It is illegal to use Social Security Numbers, as they are protected under US laws.
- HTTP URIs are more elegant, as they force people to write human readable identifiers, which is socially desired.
- HTTP URIs can easily be merged, which is crucial when merging datasets.
- HTTP URIs are unique, as one has to buy them using a domain service provider (DSP).
- HTTP URIs encode the graph structure of a knowledge graph, and thus allow for more efficient retrieval.

Possible Answer	**Correct / Wrong**
HTTP URIs can be dereferenced, ie they can be accessed via http, and provide useful information about a resource.	Correct, while HTTP URIs were designed to be dereferenced, Apples App ID or the SSN of a person cannot be dereferenced.
HTTP is the only way to access data from a triple-store, so it is critical that your URIs are HTTP addresses.	Wrong, URIs for identifying resources aren't limited to HTTP; various other schemes are also valid. Triple-store data access is facilitated via protocols like SPARQL and RDF interfaces.
Chances are very high that HTTP URIs are unique, whereas other identifiers can be the same.	Correct, because HTTP URIs often ensure global uniqueness, reducing the likelihood of conflicts, while other identifiers might not guarantee uniqueness, potentially leading to collisions.
HTTP is the only way to provide information about resources that is human readable.	Wrong, HTTP is a protocol for data transfer, not inherently for human-readable information. HTML is often used with HTTP for human-readable data, but other formats can also achieve this (JSON, XML).
HTTP uses hypertext, and is therefore far more secure than standard identifiers.	Wrong, HTTP and hypertext are not inherently linked to security. HTTP is a protocol for transferring data, while hypertext is a method for organizing and displaying information with hyperlinks.
It is illegal to use Social Security Numbers, as they are protected under US laws.	Wrong, while using Social Security Numbers for certain purposes may be restricted by law due to privacy concerns, their use is not illegal but regulated to protect individuals' personal information.
HTTP URIs are more elegant, as they force people to write human readable identifiers, which is socially desired.	Wrong, HTTP URIs serve a different purpose—they uniquely identify resources on the web and enable web-based retrieval. The elegance of an identifier is subjective; each has its own use case and context.
HTTP URIs can easily be merged, which is crucial when merging datasets.	Wrong, HTTP URIs are designed for web resource identification, not dataset merging. App-specific identifiers like App IDs are used for unique identification within their respective contexts and aren't intended for dataset merging.
HTTP URIs are unique, as one has to buy them using a domain service provider (DSP).	Wrong, HTTP URIs are unique due to their domain structure, not because they are purchased. App IDs and Social Security Numbers are distinct identifiers used for different purposes.
HTTP URIs encode the graph structure of a knowledge graph, and thus allow for more efficient retrieval.	Wrong, HTTP URIs are used for resource identification on the web, not encoding graph structures. Efficiency in retrieval is a function of proper data modeling and indexing within a knowledge graph.

Question 9:

Consider the small knowledge graph:

```
@prefix dc: <http://ntnu.edu/dc/elements/1.1/> .
@prefix stock: <http://ntnu.edu/stock#> .
@prefix inv: <http://ntnu.edu/inventory#> .
@prefix dbo: <http://dbpedia.org/ontology/>

stock:book1 dc:title "SPARQL Introduction" .
stock:book1 rdf:type dbo:ComputerScience .
stock:book1 inv:price 10 .
stock:book1 inv:quantity 3 .
stock:book2 dc:title "World War II" .
stock:book2 rdf:type dbo:History .
stock:book2 inv:price 20 .
stock:book2 inv:quantity 5 .
stock:book3 dc:title "SpaceX" .
stock:book3 rdf:type dbo:ScienceFiction .
stock:book3 inv:price 20 .
stock:book3 inv:quantity 8 .
stock:book4 dc:title "Novels from Rome" .
stock:book4 rdf:type dbo:Novel.
stock:book4 inv:price 14 .
stock:book4 inv:quantity 2 .
```

What would the following SPARQL query return?

```
PREFIX dc: <http://ntnu.edu /dc/elements/1.1/>
PREFIX stock: <http://ntnu.edu /stock#>
PREFIX inv: <http://ntnu.edu /inventory#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?title ?price ?num
WHERE {
   ?book dc:title ?title .
   ?book inv:price ?price . FILTER ( ?price > 15 )
   ?book inv:quantity ?num . FILTER ( ?num > 0 ) }
```

- "Sparql Introduction" 10 3
- "SpaceX" 20 8
- "World War II" 20 5
- "World War II" 5 20
- "Lost in Space" 5 0
- "SpaceX"

The SPARQL query will return 3 variables: ?title, ?price and ?num. From this we can already conclude ("SpaceX") is false. The query selects all books with a title ?title. It then selects books that has the price ?price, where it filters to only include books where the price is above 15. From this we can also remove ("Sparql Introduction" 10 3), ("Lost in Space" 5 0) and ("World War II" 5 20). The SPARQL query then selects books that has a quantity ?num, but only if the quantity is higher than 0. This line removes no triples, so the final results that remain are ("SpaceX" 20 8) and ("World War II" 20 5).

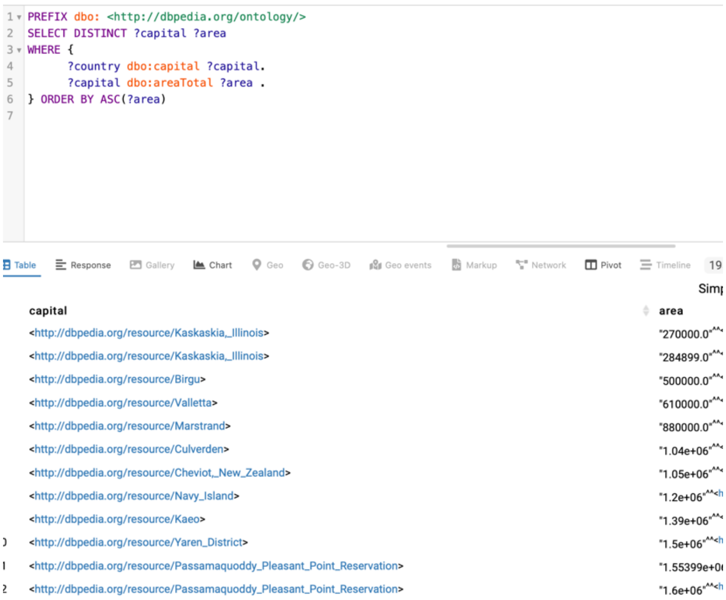
Question 10:

Remember the following query from the working group (executed using Yasgui.org against the dbpedia sparql endpoint):

```
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT DISTINCT ?capital ?area
WHERE {
      ?country dbo:capital ?capital.
      ?capital dbo:areaTotal ?area .
}
```

Modify the query to arrange the results by area in ascending order. What is the binding to ?capital for result number 4? Give the full URI (without angled brackets, and without namespace prefix).

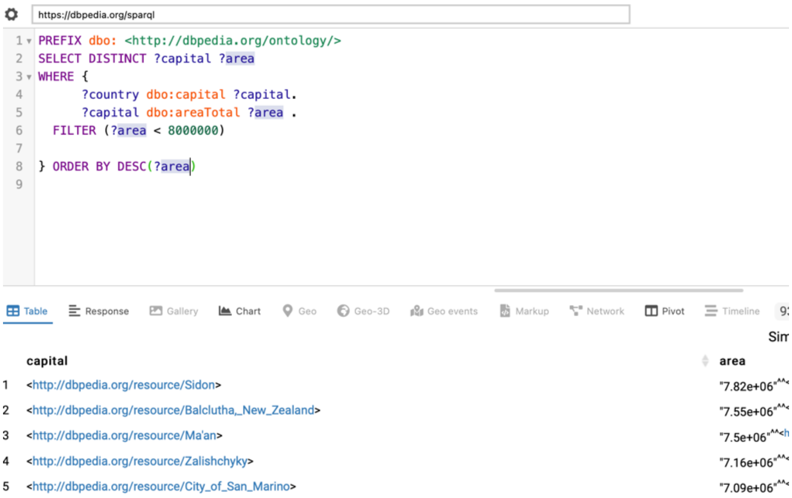
To do this, we first need to modify the SPARQL query and add a ORDER statement, we need to sort the results by area in ascending order. To do this add "ORDER BY ASC(?area)" to the SPARQL query after the where {}. After running the query we can see the 4th result is http://dbpedia.org/resource/Valletta



Question 11:

Change the query to include only the results where the area is under 8000000, now ordered by area from large to small. What is the binding to ?capital for the fifth (5th) result?

To do this, we first need to add a FILTER statement that filters out any capitals where capital area is larger than 8000000 (FILTER (?area < 8000000)). Then we change the ORDER by statement to sort descending (ASC → DESC).



http://dbpedia.org/resource/City_of_San_Marino

Question 12:

Consider the following RDF graph:

```
@prefix foaf:      <http://xmlns.com/foaf/0.1/> .
_:a  foaf:name       "Alice" .
_:a  foaf:homepage   <http://work.example.org/alice/> .
_:b  foaf:name       "Bob" .
_:b  foaf:mbox       <mailto:bob@work.example> .
```

Which of the queries below produces the following result:

name	mbox	hpage
"Alice"		<http://work.example.org/alice/>
"Bob"	<mailto:bob@work.example>	

- PREFIX foaf: <http://xmlns.com/foaf/0.1/>
  SELECT ?name ?mbox ?hpage
  WHERE {
  ?x foaf:name ?name .
  OPTIONAL { ?x foaf:mbox ?mbox } .
  OPTIONAL { ?x foaf:homepage ?hpage }
  }

- ```
  PREFIX foaf: <http://xmlns.com/foaf/0.1/>
  SELECT ?name ?mbox ?hpage
  WHERE {
      ?x foaf:name ?name .
      ?x foaf:mbox ?mbox .
      OPTIONAL { ?x foaf:homepage ?hpage }
  }
  ```
- ```
  PREFIX foaf: <http://xmlns.com/foaf/0.1/>
  SELECT ?name ?mbox ?hpage
  WHERE {
      ?x foaf:name ?name .
      OPTIONAL { ?x foaf:mbox ?mbox } .
      ?x foaf:homepage ?hpage
  }
  ```
- ```
  PREFIX foaf: <http://xmlns.com/foaf/0.1/>
  SELECT ?name ?mbox ?hpage
  WHERE {
      OPTIONAL { ?x foaf:name ?name } .
      OPTIONAL { ?x foaf:mbox ?mbox } .
      OPTIONAL { ?x foaf:homepage ?hpage }
  }
  ```

Query	Output		

PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox ?hpage
WHERE {
?x foaf:name ?name .
OPTIONAL { ?x foaf:mbox ?mbox } .
OPTIONAL { ?x foaf:homepage ?hpage }
}
	name	**mbox**	**hpage**

	"Alice"		<http://work.example.org/alice/>
	"Bob"	<mailto:bob@work.example>	

PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox ?hpage
WHERE {
?x foaf:name ?name .
?x foaf:mbox ?mbox .
OPTIONAL { ?x foaf:homepage ?hpage }
}
	name	**mbox**	**hpage**

	“Bob”	<mailto:bob@work.example>	

PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox ?hpage
WHERE {
?x foaf:name ?name .
OPTIONAL { ?x foaf:mbox ?mbox } .
?x foaf:homepage ?hpage
}
	name	**mbox**	**hpage**

	"Alice"		<http://work.example.org/alice/>

PREFIX foaf: <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox ?hpage
WHERE {
OPTIONAL { ?x foaf:name ?name } .
OPTIONAL { ?x foaf:mbox ?mbox } .
OPTIONAL { ?x foaf:homepage ?hpage }
}
	name	**mbox**	**hpage**

	"Alice"		<http://work.example.org/alice/>
	"Bob"	<mailto:bob@work.example>	

While query 1 and query 4 provide the same results, query 1 is deemed correct.
  ```

Module 2, Quiz 2:
Question 1:

Is the following RDF turtle file syntactically correct or not?

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix people: <http://data.example.org/people/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

people:joe
a foaf:Person ;
foaf:name "Joe Raad" ;
foaf:knows people:victor ;
           people:stefan.

people:stefan
a foaf:Person .
```

- Yes, this turtle file is syntactically correct
- No, the prefixes declarations are not well-formed
- No, the semicolon after "people:victor" should be a comma
- No, the indentation is incorrect

Possible answer	*Correct/ Wrong*
Yes, this turtle file is syntactically correct	Wrong, because the turtle does not have correct syntax.
No, the prefixes declarations are not well-formed	Wrong, because the prefix declarations are well-formed.
No, the semicolon after "people:victor" should be a comma	Correct, because the semicolon after "people:victor" should be a comma in turtle form.
No, the indentation is incorrect	Wrong, because the indentation is correct.

Question 2:

Consider the small knowledge graph:

```
@prefix dc: <http://ntnu.edu/dc/elements/1.1/> .
@prefix stock: <http://ntnu.edu/stock#> .
@prefix inv: <http://ntnu.edu/inventory#> .
@prefix dbo: <http://dbpedia.org/ontology/>

stock:book1 dc:title "SPARQL Introduction" .
stock:book1 rdf:type dbo:ComputerScience .
stock:book1 inv:price 10 .
stock:book1 inv:quantity 3 .
stock:book2 dc:title "World War II" .
stock:book2 rdf:type dbo:History .
stock:book2 inv:price 20 .
stock:book2 inv:quantity 5 .
stock:book3 dc:title "SpaceX" .
stock:book3 rdf:type dbo:ScienceFiction .
stock:book3 inv:price 20 .
stock:book3 inv:quantity 8 .
stock:book4 dc:title "Novels from Rome" .
stock:book4 rdf:type dbo:Novel.
stock:book4 inv:price 14 .
stock:book4 inv:quantity 2 .
```

What would the following SPARQL query return?

```
PREFIX dc: <http://ntnu.edu /dc/elements/1.1/>
PREFIX stock: <http://ntnu.edu /stock#>
PREFIX inv: <http://ntnu.edu /inventory#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?price ?num ?title
WHERE {
   ?book dc:title ?title .
   ?book inv:price ?price . FILTER ( ?price < 15 )
   ?book inv:quantity ?num . FILTER ( ?num < 4 ) }
```

- 14 2 "Novels from Rome"
- "Novels from Rome" 14 2
- "Sparql Introduction" 10 3
- "Sparql Introduction"
- "Lost in Space" 5 0
- 10 3 "SPARQL Introduction"

The SPARQL query will return 3 variables: ?price, ?num and ?title. From this we can already conclude ("Novels from Rome" 14 2), ("Sparql Introduction"), ("Lost in Space" 5 0) and ("Sparql Introduction" 10 3) are false. The query selects all books with a title ?title. It then selects books that has the price ?price, where it filters to only include books where the price is above 15. The SPARQL query then selects books that has a quantity ?num, but only if the quantity is higher than 0. This line removes no triples, so the final results that remain are (10 3 "SPARQL Introduction") and (14 2 "Novels from Rome").

Question 3:
Let KG={(a,p,a),(b,q,a)} be a knowledge graph.

Which of the following interpretations is a model of KG?

- (U,I) with U={1,2}, I(a)=1,I(b)=2, I(p)={(1,1)}, I(q)={(2,1)}
- (U,I) with U={1,2}, I(a)=1,I(b)=2, I(p)={(1,2)}, I(q)={(2,2)}
- (U,I) with U={1,2}, I(a)=2,I(b)=2, I(p)={(1,2)}, I(q)={(2,2)}
- (U,I) with U={1,2}, I(a)=2,I(b)=2, I(p)={(1,2)}, I(q)={(2,1)}
- (U,I) with U={1,2}, I(a)=2,I(b)=1, I(p)={(2,1)}, I(q)={(1,2)

Possible answer	***Correct/ Wrong***
(U,I) with U={1,2}, I(a)=1,I(b)=2, I(p)={(1,1)}, I(q)={(2,1)}	Correct, because if (a p a) and (b q a) are in the knowledge graph, if p = {(1,1)}; if b = 2 and p = {(2,1)} everything is semantically correct
(U,I) with U={1,2}, I(a)=1,I(b)=2, I(p)={(1,2)}, I(q)={(2,2)}	Wrong, because if (a p a) and (b q a) are in the knowledge graph, p can't be {(2,2)}; if a = 1.
(U,I) with U={1,2}, I(a)=2,I(b)=2, I(p)={(1,2)}, I(q)={(2,2)}	Wrong, because if (a p a) and (b q a) are in the knowledge graph, p can't be {(1,2)}; if a = 2, p has to be {(2, 2)}.
(U,I) with U={1,2}, I(a)=2,I(b)=2, I(p)={(1,2)}, I(q)={(2,1)}	Wrong, because if (a p a) and (b q a) are in the knowledge graph, p can't be {(1,2)}; if a = 2, p has to be {(2, 2)}.
(U,I) with U={1,2}, I(a)=2,I(b)=1, I(p)={(2,1)}, I(q)={(1,2)}	Wrong, because if (a p a) and (b q a) are in the knowledge graph, p can't be {(2,1)}; if a = 2, p has to be {(2, 2)}.

Module 2, Worksheet

Example question 1 (Modelling tabular information as a graph):

Consider the following relational databases: Courses and StudyInformation

Study	Name	Language
IMM	Information, Management and Multimedia	Dutch
IK	Informatiekunde	Dutch

Student	vunetID	Prov	Studies
s1	as344	UvA	IK
s2	ex444	VU	IMM

With tables Courses and StudyInformation. Please write down a knowledge graph to represent the knowledge of this graph as faithfully as possible. In order to do so, first determine what you want to model as resources, literals and properties. There is no need to model the entire dataset: give 10 triples. Those triples should include 1) triples that model implicit knowledge to enrich the data, e.g. about types, and you 2) should make a distinction between objects and literals. Use the simple notation (s p o) from module 1.

s	p	o
s1	hasVunetID	as344
s2	hasstudy	IMM
s1	hasstudy	IK
IMM	hasName	“Information, Management and Multimedia”
IK	hasName	Informatiekunde
IMM	hasLanguage	Dutch
IK	hasLanguage	Dutch
s1	studiesAt	UvA
s2	studiesAt	VU
s1	a	student
s2	a	student

Example question 2 (RDF and Turtle)

Consider the knowledge graph you just built. Transform this set of triples into real RDF in the Turtle serialisation. Concretely, make sure you use URIs to define namespaces, and abbreviate them with prefixes. Also make sure you use at least one of the syntax shortcuts Turtle provides.

```
@prefix ex: <http://example.org/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

ex:s1 ex:hasVunetID "as344" ;
     ex:hasstudy ex:IK,
                 ex:IMM ;
     ex:studiesAt ex:UvA ;
     a ex:student .

ex:s2 ex:hasstudy ex:IMM ;
     ex:studiesAt ex:VU ;
     a ex:student .

ex:IMM ex:hasName "Information, Management and Multimedia" ;
     ex:hasLanguage "Dutch" .

ex:IK ex:hasName "Informatiekunde" ;
     ex:hasLanguage "Dutch" .

ex:UvA rdfs:label "University of Amsterdam" .

ex:VU rdfs:label "VU University Amsterdam" .
```

Example question 3 (Models for knowledge graphs):

Let KG={(a,p,b),(b,p,c)} be a knowledge graph. Which of the following interpretations is a model of KG? (for simplicy reasons, there is no difference made between the interpretation functions for properties and resources).

- (U,I) with U={1,2}, I(a)=2,I(b)=1,I(c)=1, I(p)={(1,2)}
- (U,I) with U={1,2}, I(a)=2,I(b)=1,I(c)=2, I(p)={(1,2)}
- (U,I) with U={1,2}, I(a)=2,I(b)=1,I(c)=2, I(p)={(2,2)}
- (U,I) with U={1,2}, I(a)=2,I(b)=1,I(c)=1, I(p)={(2,2),(1,1)}
- (U,I) with U={1,2}, I(a)=2,I(b)=2,I(c)=2, I(p)={(2,2)}

Possible answer	***Correct/ Wrong***
(U,I) with U={1,2}, I(a)=2,I(b)=1,I(c)=1, I(p)={(1,2)}	Wrong, because if (a p b) and (b q c) are in the knowledge graph, p can't be {(1,2)}; if a = 2, p has to be {(2, b)}.
(U,I) with U={1,2}, I(a)=2,I(b)=1,I(c)=2, I(p)={(1,2)}	Wrong, because if (a p b) and (b q c) are in the knowledge graph, p can't be {(1,2)}; if a = 2, p has to be {(2, b)}.
(U,I) with U={1,2}, I(a)=2,I(b)=1,I(c)=2, I(p)={(2,2)}	Wrong, because if (a p b) and (b q c) are in the knowledge graph, p can't be {(2,2)}; if b = 1, p has to be {(a, 1)}.
(U,I) with U={1,2}, I(a)=2,I(b)=1,I(c)=1, I(p)={(2,2),(1,1)}	Wrong, because if a = 2, b = 1, and p connects a and b, p cannot have to tuples of 2 numbers.
(U,I) with U={1,2}, I(a)=2,I(b)=2,I(c)=2, I(p)={(2,2)}	Correct, because if (a p b) and (b q c) are in the knowledge graph, a = 2, b = 2. p has to be {(2,2)}.

Example question 5 (URIs and Resources):
Describe in your own words what the difference is between a URI and a Resource.

Resources can be physical objects ("things") (like people, cars, and molecules), non-physical objects (like concepts and dreams), attributes of other resources (literals). URIs/IRIs uniquely identify physical and non-physical resources.

Example question 6 (Exploring DBPedia)
Go to DBPedia and find which triples below form a path consisting of three steps between

- http://dbpedia.org/resource/Amsterdam and
- http://dbpedia.org/resource/Kingdom_of_the_Netherlands

A path is any connection between resources that are connected via some property, independent of the directionality. So, if you have (s p o) and (o p q) there is a path between s and q. But also if you have (s p o) and (q p o).
We use the following namespace prefixes:

- dbr: http://dbpedia.org/resource/
- dbo: http://dbpedia.org/ontology/
- dbp: http://dbpedia.org/property/

Select the triples that together make the path.

- dbr:Netherlands dbo:leader dbr:Willem-Alexander_of_the_Netherlands
- dbr:Kingdom_of_the_Netherlands dbo:language dbr:Dutch_language
- dbr:Amsterdam rdf:type dbo:Place
- dbr:Kingdom_of_the_Netherlands dbo:leader dbr:Willem-Alexander_of_the_Netherlands
- dbr:Netherlands dbo:part_of dbr:Kingdom_of_the_Netherlands
- dbr:Willem-Alexander_of_the_Netherlands rdf:type dbo:Person
- dbr:Amsterdam dbo:country dbr:Netherlands

The capitals that form this path are:
dbr:Netherlands dbo:leader dbr:Willem-Alexander_of_the_Netherlands
dbr:Kingdom_of_the_Netherlands dbo:leader dbr:Willem-Alexander_of_the_Netherlands
dbr:Amsterdam dbo:country dbr:Netherlands

Example question 7:
Try out the following SPARQL query by executing it against the DBPedia SPARQL endpoint
(http://dbpedia.org/sparql) in YASGUI (http://yasgui.org)

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?capital ?area
WHERE {
?country dbo:capital ?capital.
?capital dbo:areaTotal ?area .
}
```

This query returns 3832 results.

Example question 8 (Eliminate Duplicates with SPARQL):

Adapt the query to eliminate duplicate lines from the results. The query now returns 1932 results.

“DISTINCT” was added.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT DISTINCT ?capital ?area
WHERE {
?country dbo:capital ?capital.
?capital dbo:areaTotal ?area .
}
```

Example question 9 (SPARQL queries):

Restrict the query to include only bindings to ?capital that have a `rdfs:label` relation with the English language literals "Amsterdam" or "Berlin". Provide the SPARQL query as the answer.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT DISTINCT ?capital ?area
WHERE {
?country dbo:capital ?capital.
?capital dbo:areaTotal ?area .
?capital rdfs:label ?label
FILTER(?label = "Amsterdam"@en || ?label = "Berlin"@en)
}
```

Example question 10 (SPARQL queries):

Extend the query for previous question to also retrieve the value of the dbo:areaWater property, and bind it to a variable ?areaWater. Make sure it still returns both dbpedia:Amsterdam and dbpedia:Berlin as bindings to ?capital. Provide the SPARQL query as the answer.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT DISTINCT ?capital ?area ?areaWater
WHERE {
?country dbo:capital ?capital.
?capital dbo:areaTotal ?area .
?capital rdfs:label ?label .
  OPTIONAL{?capital dbo:areaWater ?areaWater}
FILTER(?label = "Amsterdam"@en || ?label = "Berlin"@en)
}
```

Module 3, Quiz 1:
Question 1:
Given the following RDF in Turtle format:

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#>
@prefix ex: <http://example.com/>

ex:S rdfs:subClassOf ex:T .
ex:T owl:disjointWith ex:O .
ex:r rdfs:domain ex:S .
ex:r rdfs:range ex:T .
ex:z rdfs:subPropertyOf ex:r .
ex:r rdf:type owl:ObjectProperty .
ex:z rdf:type owl:ObjectProperty .
ex:s rdf:type ex:S .
ex:t rdf:type ex:T .
ex:o rdf:type ex:O .
ex:t ex:z ex:s .
ex:t ex:z ex:o .
```

If I run an RDFS reasoner on this, what statement(s) should it infer?
- ex:r rdf:type rdf:Property .
- rdf:Property rdf:type rdfs:Class .
- Nothing, the ontology contains a contradiction.
- ex:s rdf:type ex:T .
- rdfs:domain rdf:type rdf:Property .
- ex:O rdfs:subClassOf ex:T .
- rdf:type rdf:type rdf:Property .
- ex:c owl:differentFrom ex:b .

The reasoning rules for inferring with RDFS are the following:
- (rdfs2) If G contains the triples (aaa rdfs:domain xxx.) and (uuu aaa yyy.) then infer the triple (uuu rdf:type xxx.)
- (rdfs3) If G contains the triples (aaa rdfs:range xxx.) and (uuu aaa vvv.) then infer the triple (vvv rdf:type xxx .)
- (rdfs5) If G contains the triples (uuu rdfs:subPropertyOf vvv.) and (vvv rdfs:subPropertyOf xxx.) then infer the triple (uuu rdfs:subPropertyOf xxx.)
- (rdfs7) If G contains the triples (aaa rdfs:subPropertyOf bbb.) and (uuu aaa yyy.) then infer the triple (uuu bbb yyy)
- (rdfs9) If G contains the triples (uuu rdfs:subClassOf xxx.) and (vvv rdf:type uuu.) then infer the triple (vvv rdf:type xxx.) -> this one was not mentioned in the lecture, but is a very important one.
- (rdfs11) If G contains the triples (uuu rdfs:subClassOf vvv.) and (vvv rdfs:subClassOf xxx.) then infer the triple (uuu rdfs:subClassOf xxx.)

OR:

1.	s	p	o	→	s	rdf:type	rdfs:Resource
					p	rdf:type	rdf:Property
					o	rdf:type	rdfs:Resource
						OR	
					o	rdf:type	rdfs:Literal
2.	s	rdf:type	o		o	rdf:type	rdfs:Class
3.	s	rdf:type	rdfs:Class	→	s	rdfs:subClassOf	s
4.	p	rdf:type	rdf:Property	→	p	rdfs:subPropertyOf	p
5.	s	rdfs:subClassOf	y				
	y	rdfs:subClassOf	z	→	s	rdfs:subClassOf	z
6.	p	rdfs:subPropertyOf	q				
	q	rdfs:subPropertyOf	r	→	p	rdfs:subPropertyOf	r
7.	s	p	o				
	p	rdfs:subPropertyOf	q	→	s	q	o
8.	s	p	o				
	p	rdfs:domain	x	→	s	rdf:type	x
9.	s	p	o				
	p	rdfs:range	x	→	o	rdf:type	x
10.	s	rdf:type	o				
	o	rdfs:subClassOf	t	→	s	rdf:type	x

Possible answer	**Correct / wrong**
ex:r rdf:type rdf:Property .	Correct, because ex:z rdfs:subPropertyOf ex:r . and ex:t ex:z ex:s . we can use rule 7 to infer that ex:t ex:r ex:s. And with rule 4 that ex:r rdf:type rdf:Property .
rdf:Property rdf:type rdfs:Class .	Correct, because ex:r rdf:type rdf:Property . we can use rule 1 to infer that rdf:Property rdf:type rdfs:Class .
Nothing, the ontology contains a contradiction.	Wrong, the ontology does not contain a contradiction.
ex:s rdf:type ex:T .	Correct, because ex:s rdf:type ex:S . and ex:S rdfs:subClassOf ex:T, we can use rule 9 to infer that ex:s rdf:type ex:T
rdfs:domain rdf:type rdf:Property .	Wrong, there are no inference rules that can be used to infer this.
ex:O rdfs:subClassOf ex:T .	Wrong, there are no inference rules that can be used to infer this.
rdf:type rdf:type rdf:Property .	Correct, because ex:s rdf:type ex:S, we can use rule 1 to infer that rdf:type rdf:type rdf:Property.
ex:c owl:differentFrom ex:b .	Wrong, there are no inference rules that can be used to infer this.

Question 2:

What is the function of vocabularies such as foaf, dublin core or skos? (multiple correct answers possible, select them all)

- It is illegal to use RDFS and foaf in one document, as the uncertainty and underspecification of semantics of foaf would destroy the formal semantics of RDFS.
- While they do not provide a lot of formal semantics (and are thus not very powerful in inferencing new facts) vocabularies are crucial to the Linked Data initiative, as one cannot publish valid RDF without using these vocabularies.
- It is highly recommended to use extra vocabularies in RDF and RDFS documents in order to make reuse of the data easier.
- While they do not provide a lot of formal semantics (and are thus not very powerful in inferencing new facts) these vocabularies help people share the same terms when talking about things. They provide some kind of social semantics to a formal language.
- While they do not provide a lot of formal semantics (and are thus not very powerful in inferencing new facts) vocabularies help because people are forced to use the same terms for domains in order to adhere to Tim Berners Lees 5-Star principle.
- These vocabularies, foaf, skos and dc, come with strict formal semantics that allow for rich inferencing.
- There is no need to use dublin core and foaf vocabularies, as they are already included in RDFS.

Possible answer	**Correct / wrong**
It is illegal to use RDFS and foaf in one document, as the uncertainty and underspecification of semantics of foaf would destroy the formal semantics of RDFS.	Wrong, because it is not wrong to use RDFS and foaf if this is necessary for your purposes.
While they do not provide a lot of formal semantics (and are thus not very powerful in inferencing new facts) vocabularies are crucial to the Linked Data initiative, as one cannot publish valid RDF without using these vocabularies.	Wrong, while vocabularies like FOAF, Dublin Core, and SKOS are valuable in describing resources, they are not mandatory for publishing valid RDF. RDF can use custom vocabularies and still be valid. Standardized vocabularies enhance interoperability but aren't a strict requirement.
It is highly recommended to use extra vocabularies in RDF and RDFS documents in order to make reuse of the data easier.	Correct. Vocabularies like FOAF, Dublin Core, and SKOS enhance RDF and RDFS documents by providing standardized terms, facilitating data reuse, interoperability, and alignment with common data structures.
While they do not provide a lot of formal semantics (and are thus not very powerful in inferencing new facts) these vocabularies help people share the same terms when talking about things. They provide some kind of social semantics to a formal language.	Correct, these vocabularies help establish a common understanding by providing shared terms. While they may not offer extensive formal semantics, they enhance data interoperability and encourage standardization.
While they do not provide a lot of formal semantics (and are thus not very powerful in inferencing new facts) vocabularies help because people are forced to use the same terms for domains in order to adhere to Tim Berners Lees 5-Star principle	Wrong, vocabularies like FOAF, Dublin Core, and SKOS are useful not just for adhering to the 5-Star Linked Data principle but also for providing structured and standardized descriptions of resources, improving data interoperability. They do not force people but encourage the use of common terms.
These vocabularies, foaf, skos and dc, come with strict formal semantics that allow for rich inferencing.	Wrong, while FOAF, SKOS, and DC provide structured and standardized ways to describe resources, they do not typically come with strict formal semantics for rich inferencing. They focus on data interoperability and metadata.
There is no need to use dublin core and foaf vocabularies, as they are already included in RDFS.	Wrong, Dublin Core and FOAF vocabularies provide additional, specialized terms and concepts not included in the more generic RDF Schema (RDFS). Their usage depends on specific data needs.

Question 3:
This question is about the role of an index in a database, more specifically in TripleStores. Which of the following statements are correct, and which are not?

- Triplestores use indices to answer queries faster.
- There is a single way of indexing a database .
- A triple-store can only be indexed by the subject of the triples .
- There are several ways of indexing data in a triple-store, by subject, by object and even first by subject and then by predicate.
- Triplestores use indices to save memory.

TripleStores are purpose-build graph databases to deal with RDF data. Data can be stored on disk (more scalable) or in memory (faster). TripleStores often come with GUIs and a SPARQL interface. Some examples are GraphDB, Stardog, SWI Prolog. The queries from triplestores execute faster because of dictionary use, indexing and statistics.

Possible answer	**Correct / wrong**
Triplestores use indices to answer queries faster.	Correct, triplestores employ various types of indices to enhance query performance. Indices help in rapidly accessing specific RDF data within the triplestore, significantly improving query response times. Do the s
There is a single way of indexing a database.	Wrong, there isn't a single way to index a database. Different databases, including triplestores, can use various indexing techniques based on their data structures and query patterns.
A triple-store can only be indexed by the subject of the triples.	Wrong, a triple-store can be indexed in different ways, including by subject, predicate, and object, allowing for versatile query optimization based on the specific needs of the data and queries.
There are several ways of indexing data in a triple-store, by subject, by object and even first by subject and then by predicate.	Correct, there are several ways of indexing data in a triple-store, including indexing by subject, object, and predicate. These multiple indexing options enhance query performance.
Triplestores use indices to save memory.	Wrong, triplestores use indices primarily for query performance and data retrieval speed, not memory conservation. Indices aid in rapid data access and query optimization.

Question 4:
What happens when you send a query:

SELECT * WHERE {?s ?p ?o } LIMIT 10

to the following adress via Yasgui:

http://dbpedia.org/sparql

- Yasgui has a local copy of DBPedia, ready downloaded and indexed for efficient processing. It returns the first 10 triples in the database.
- Yasqui sends an http GET request to the URL which denotes a SPARQL API. The specified query is ran against the DBPedia triple store, and the first 10 triples in that store are returned to Yasgui, which renders it nicely.
- Yasgui will attempt to send a query to a database http://dbpedia.org/sparql, but there will be an error message, as it is not a valid http address
- Yasgui runs an instance of a triple store server, which it calls "http://dbpedia.org/sparql". This instance returns the first 10 triples it has indexed and Yasgui renders them nicely.

Possible answer	Correct / wrong
Yasgui has a local copy of DBPedia, ready downloaded and indexed for efficient processing. It returns the first 10 triples in the database.	Wrong, Yasgui does not have a local copy of DBPedia as this would be inefficient storage usage.
Yasqui sends an http GET request to the URL which denotes a SPARQL API. The specified query is ran against the DBPedia triple store, and the first 10 triples in that store are returned to Yasgui, which renders it nicely.	Correct, this is how Yasgui and similar SPARQL gui's operate.
Yasgui will attempt to send a query to a database http://dbpedia.org/sparql, but there will be an error message, as it is not a valid http address	Wrong, when you send the query "SELECT * WHERE {?s ?p ?o } LIMIT 10" to http://dbpedia.org/sparql via Yasgui, it should work without any error. The provided address is a valid SPARQL endpoint for querying the DBpedia dataset.
Yasgui runs an instance of a triple store server, which it calls "http://dbpedia.org/sparql". This instance returns the first 10 triples it has indexed and Yasgui renders them nicely.	Wrong, Yasgui does not run an instance of a triple store server called dbpedia.

Question 5:

Suppose (only) the following rule is given.

rdfs9	xxx `rdfs:subClassOf` yyy `.` zzz `rdf:type` xxx `.`	→	zzz `rdf:type` yyy `.`

and an RDFS graph: { (ex:A rdfs:subClassOf ex:D), (ex:B rdfs:subClassOf ex:C),(ex:B rdfs:subClassOf ex:D), (ex:i1 rdf:type ex:A), (ex:i2 rdf:type ex:B)}

Which of the following (sets of) triples can be derived? There might be other triples that are entailed beyond those listed.

- (ex:i2 rdf:type ex:A) and (ex:i1 rdf:type ex:D)
- (ex:i1 rdf:type ex:C) and (ex:i2 rdf:type ex:D)
- (ex:i1 rdf:type ex:D) and (ex:i2 rdf:type ex:D)
- (ex:i1 rdf:type ex:C) and (ex:i2 rdf:type ex:C)

Possible answer	**Correct / wrong**
(ex:i2 rdf:type ex:A) and (ex:i1 rdf:type ex:D)	Wrong, ex:i2 is of rdf:type ex:B, and ex:B is not a subclass of ex:A.
(ex:i1 rdf:type ex:C) and (ex:i2 rdf:type ex:D)	Wrong, ex:i1 is of rdf:type ex:A, and ex:A is not a subclass of ex:C.
(ex:i1 rdf:type ex:D) and (ex:i2 rdf:type ex:D)	Correct, ex:i1 is of rdf:type ex:A, and ex:A is a subclass of ex:D; ex:i2 is of rdf:type ex:B, and ex:B a subclass of ex:D.
(ex:i1 rdf:type ex:C) and (ex:i2 rdf:type ex:C)	Wrong, ex:i1 is of rdf:type ex:A, and ex:A is not a subclass of ex:C.

Question 6:

Now consider the RDFS graph { (ex:D rdfs:subClassOf ex:C), (ex:C rdfs:subClassOf ex:B),(ex:B rdfs:subClassOf ex:A), (ex:i1 rdf:type ex:A), (ex:i2 rdf:type ex:B)}, but now only the rule:

rdfs11	xxx `rdfs:subClassOf` yyy `.` yyy `rdfs:subClassOf` zzz `.`	→	xxx `rdfs:subClassOf` zzz `.`

Which of the following (sets of) triples can be derived? There might be other triples that are entailed beyond those listed.

- ex:i1 rdf:type ex:C) and (ex:i2 rdf:type ex:D)}
- (ex:A rdfs:subClassOf ex:C) and (ex:i1 rdf:type ex:C)
- (ex:A rdfs:subClassOf ex:C) and (ex:A rdfs:subClassOf ex:D) and (ex:B rdfs:subClassOf ex:D)
- (ex:C rdfs:subClassOf ex:A) and (ex:D rdfs:subClassOf ex:A) and (ex:C rdfs:subClassOf ex:A)

Possible answer	**Correct / wrong**
(ex:i1 rdf:type ex:C) and (ex:i2 rdf:type ex:D)	Wrong, with the inference rule, only subclasses can be derived.
(ex:A rdfs:subClassOf ex:C) and (ex:i1 rdf:type ex:C)	Wrong, with the inference rule, only subclasses can be derived.
(ex:A rdfs:subClassOf ex:C) and (ex:A rdfs:subClassOf ex:D) and (ex:B rdfs:subClassOf ex:D)	Wrong, ex:A is not a subclass of another class, which means the inferring cannot happen.
(ex:C rdfs:subClassOf ex:A) and (ex:D rdfs:subClassOf ex:A)	Correct, ex:C is a subclass of ex:B, which is a subclass of ex:A, so C is a subclass of A. ex:D is a subclass of ex;B, which means with the same reasoning of before, ex:D is also a subclass of ex:A.

Question 7:

Consider the rules for domain and range:

rdfs2	aaa `rdfs:domain` xxx `.` yyy aaa zzz `.`	→	yyy `rdf:type` xxx `.`
rdfs3	aaa `rdfs:range` xxx `.` yyy aaa zzz `.`	→	zzz `rdf:type` xxx `.`

Given a Knowledge Graph {(ex:x ex:y ex:z), (ex:y rdfs:range ex:w), (rdfs:range rdfs:range rdfs:Class), (ex:u a ex:v)}.

Which triple(s) can be derived?

- Solution: (ex:z a ex:v) and (ex:u a rdfs:Class)
- Solution: (ex:x a ex:w) and (ex:w a rdfs:Class)
- Solution: (ex:x a ex:w) and (ex:w a rdfs:Class)
- Solution: (ex:z a ex:w) and (ex:w a rdfs:Class)

Possible answer	**Correct / wrong**
Solution: (ex:z a ex:v) and (ex:u a rdfs:Class)	Wrong, with the inference rule, the object after a range predicate (xxx rdfs:range yyy) is the object of (bbb a yyy) when another triple has yyy as the predicate (aaa yyy bbb). ex:v isn't the object after a range predicate.
Solution: (ex:x a ex:w) and (ex:w a rdfs:Class)	Wrong, with the inference rule, the object after a range predicate (xxx rdfs:range yyy) is the object of (bbb a yyy) when another triple has yyy as the predicate (aaa yyy bbb). ex:x is in this case aaa, so not of rdf:type ex:w.
Solution: (ex:x a ex:w) and (ex:w a rdfs:Class)	Wrong, with the inference rule, the object after a range predicate (xxx rdfs:range yyy) is the object of (bbb a yyy) when another triple has yyy as the predicate (aaa yyy bbb). ex:x is in this case aaa, so not of rdf:type ex:w.
Solution: (ex:z a ex:w) and (ex:w a rdfs:Class)	Correct, ex:y rdfs:range ex:w, so if ex:x ex:y ex:z, we can infer ex:z is of type ex:w. And ex:w is a rdfs:Class along with this.

Question 8:

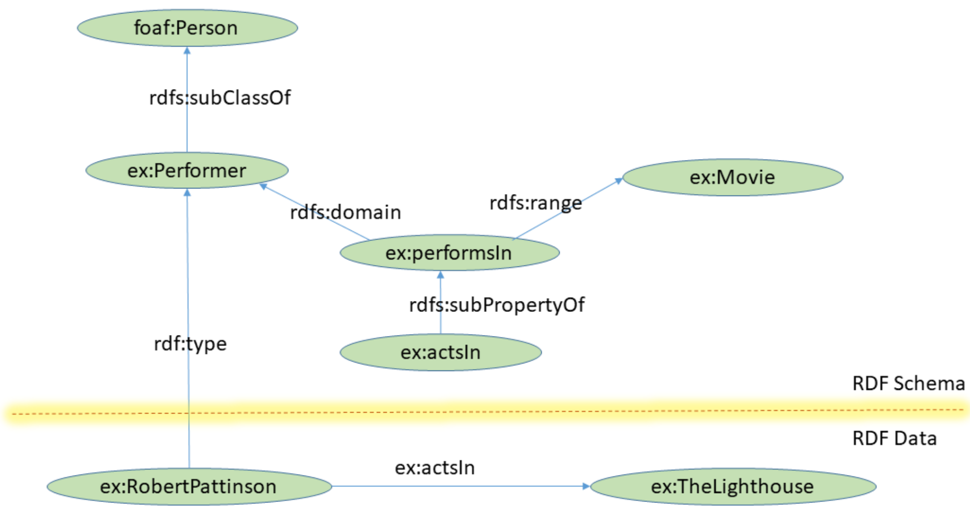
Consider the knowledge graph: {(ex:Peter ex:likes ex:John), (ex:likes rdfs:domain ex:Human), (ex:likes rdfs:range ex:Human),(ex:Peter ex:likes ex:Chocolate)}.

There is one triple you can derive which is undesirable. Which one?

- ex:John ex:likes ex:Chocolate .
- ex:Human ex:likes ex:Chocolate .
- ex:Chocolate ex:likes ex:Human .
- ex:Chocolate rdf:type ex:Human .

Possible answer	**Correct / wrong**
ex:John ex:likes ex:Chocolate .	Correct, "ex:Chocolate rdf:type ex:Human" is the undesirable triple that can be derived from the given knowledge graph. It's undesirable because it implies that chocolate is a type of human, which is incorrect.
ex:Human ex:likes ex:Chocolate .	Wrong, "ex:John ex:likes ex:Chocolate" is not undesirable as it's a valid relationship based on the provided graph.
ex:Chocolate ex:likes ex:Human .	Wrong, "ex:Human ex:likes ex:Chocolate" is not undesirable either, as it follows the graph's structure and doesn't imply any incorrect relationships.
ex:Chocolate rdf:type ex:Human .	Wrong, "ex:Chocolate ex:likes ex:Human" is also not undesirable, as it follows the graph's structure and represents the relationship between chocolate and humans, which is acceptable in this context.

Question 9:
Take the following graph:



Which triples are not explicitly stated in the graph but can be inferred by the rdf and rdfs inference rules.

- ex:RobbertPattinson rdf:type foaf:Person.
- ex:TheLightHouse rdf:type ex:Performer.
- ex:RobbertPattinson ex:actsIn ex:Movie.
- ex:RobbertPattinson rdfs:range ex:Movie.
- ex:RobbertPattinson ex:performsIn ex:TheLighthouse.
- ex:RobbertPattinson rdfs:subClassOf foaf:Person.
- ex:Movie rdf:type rdfs:Class.

Possible answer	**Correct / wrong**
ex:RobbertPattinson rdf:type foaf:Person.	Correct, rule 10 says if s rdf:type o and o:rdfs:subClassOf t, s rdf:type t. So ex:RobertPattinson is of type person.
ex:TheLightHouse rdf:type ex:Performer.	Wrong, there is no connection from the ex:theLighthouse to ex:Performer.
ex:RobbertPattinson ex:actsIn ex:Movie.	Wrong, there is no connection here. ex:RobertPattinson ex:actsIn ex:theLighthouse, acts:in is a rdfs:subPropertyOf ex:performsIn, wh
ex:RobbertPattinson rdfs:range ex:Movie.	Wrong, there is no connection here.
ex:RobbertPattinson ex:performsIn ex:TheLighthouse.	Correct, ex:actsIn is a subProperty of ex:performsIn.
ex:RobbertPattinson rdfs:subClassOf foaf:Person.	Wrong, there is no connection here.
ex:Movie rdf:type rdfs:Class.	Correct, there is no connection here. ex:RobertPattinson ex:actsIn ex:theLighthouse, acts:in is a rdfs:subPropertyOf ex:performsIn, which has rdfs:range ex:Movie, which means we can infer ex:theLighthouse rdf:type ex:Movie, and ex:Movie rdf:type rdfs:Class.

Module 3, Quiz 2:

Question 1:

Given the following RDF in Turtle format:

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix ex: <http://example.com/> .

ex:A rdfs:subClassOf ex:B .
ex:B owl:disjointWith ex:C .
ex:p rdfs:domain ex:A .
ex:p rdfs:range ex:B .
ex:q rdfs:subPropertyOf ex:p .
ex:p rdf:type owl:ObjectProperty .
ex:q rdf:type owl:ObjectProperty .
ex:a rdf:type ex:A .
ex:b rdf:type ex:B .
ex:c rdf:type ex:C .
ex:b ex:q ex:a .
ex:b ex:q ex:c .
```

If I run an RDFS reasoner on this, what statement(s) should it infer?

- ex:a rdf:type ex:B.
- rdf:type rdf:type rdf:Property.
- rdfs:domain rdf:type rdf:Property.
- ex:C rdfs:subClassOf ex:B.
- ex:p rdf:type rdf:Property.
- rdf:Property rdf:type rdfs:Class.
- ex:c owl:differentFrom ex:b.
- Nothing, the ontology contains a contradiction.

Possible answer	**Correct / wrong**
ex:a rdf:type ex:B.	Correct, because of rule 10, we can infer ex:a rdf:type ex:B,
rdf:type rdf:type rdf:Property.	Correct, because of rule 1, we can infer this.
rdfs:domain rdf:type rdf:Property.	Correct, because of rule 1, we can infer this.
ex:C rdfs:subClassOf ex:B.	Wrong, there are no inference rules that can be used to infer this.
ex:p rdf:type rdf:Property.	Correct, because of rule 7 and 1, we can infer this.
rdf:Property rdf:type rdfs:Class.	Correct, because of rule 1, we can infer this.
ex:c owl:differentFrom ex:b.	Wrong, there are no inference rules that can be used to infer this.
Nothing, the ontology contains a contradiction.	Wrong, there ontology doesn't contain a contradiction.

Question 2:
Now consider the RDFS graph { (ex:A rdfs:subClassOf ex:B), (ex:B rdfs:subClassOf ex:C),(ex:C rdfs:subClassOf ex:D), (ex:i1 rdf:type ex:A), (ex:i2 rdf:type ex:B)}, but now only the rule:

rdfs11	xxx `rdfs:subClassOf` yyy `.` yyy `rdfs:subClassOf` zzz `.`	→	xxx `rdfs:subClassOf` zzz `.`

Which of the following (sets of) triples can be derived? There might be other triples that are entailed beyond those listed.

- (ex:C rdfs:subClassOf ex:A) and (ex:D rdfs:subClassOf ex:A) and (ex:D rdfs:subClassOf ex:A)
- (ex:A rdfs:subClassOf ex:C) and (ex:A rdfs:subClassOf ex:D) and (ex:B rdfs:subClassOf ex:D)
- (ex:A rdfs:subClassOf ex:C) and (ex:i1 rdf:type ex:C)
- (ex:i1 rdf:type ex:C)

Possible answer	**Correct / wrong**
(ex:C rdfs:subClassOf ex:A) and (ex:D rdfs:subClassOf ex:A) and (ex:D rdfs:subClassOf ex:A)	Wrong, with the inference rule, C is subClassOf D, and D isn't the subclass of another class.
(ex:A rdfs:subClassOf ex:C) and (ex:A rdfs:subClassOf ex:D) and (ex:B rdfs:subClassOf ex:D)	Correct, A is subclass of B, and B is subclass of C, C is subclass of D.
(ex:A rdfs:subClassOf ex:C) and (ex:i1 rdf:type ex:C)	Wrong, with the inference rule, only subclasses can, be derived.
(ex:i1 rdf:type ex:C)	Wrong, with the inference rule, only subclasses can be derived.

Question 3:
Consider the knowledge graph: {(ex:Stefan ex:works_with ex:Klaas), (ex:works_with rdfs:domain ex:Human), (ex:works_with rdfs:range ex:Human),(ex:Stefan ex:works_with ex:LapTop)}.

There is one triple you can derive which is undesirable. Which one? (1 line)

- ex:LapTop rdf:type ex:Human .
- ex:Klaas rdf:type ex:Stefan .
- ex:Klaas ex:works_with ex:Laptop .
- ex:Klaas rdf:type ex:LapTop .

Possible answer	**Correct / wrong**
ex:LapTop rdf:type ex:Human .	Correct, "ex:LapTop rdf:type ex:Human" is the undesirable triple that can be derived from the given knowledge graph. It's undesirable because it implies that laptop is a type of human, which is incorrect.
ex:Klaas rdf:type ex:Stefan .	Wrong, "ex:Klaas rdf:type ex:Stefan" is not undesirable as it cannot be derived.
ex:Klaas ex:works_with ex:Laptop .	Wrong, "ex:Klaas ex:works_with ex:Laptop" is not undesirable either, as it follows the graph's structure and doesn't imply any incorrect relationships.
ex:Klaas rdf:type ex:LapTop .	Wrong, "ex:Klaas rdf:type ex:LapTop" is also not undesirable, as it cannot be derived.