Ikhlas Attarwala / MSAI 371
01-17-19 / HW 1

color(answers, red)

---

1 - OBJECTS

1. Define objects for courses in hw1.pdf:
    a. **Cs111**
        i. The constant `Cs111` represents the course `CS 111`.
    b. **Cs211**
        i. The constant `Cs211` represents the course `CS 211`.
    c. **Cs321**
        i. The constant `Cs321` represents the course `CS 321`.
    d. **Cs330**
        i. The constant `Cs330` represents the course `CS 330`.
    e. **Cs335**
        i. The constant `Cs335` represents the course `CS 335`.
    f. **Cs338**
        i. The constant `Cs338` represents the course `CS 338`.
    g. **Cs348**
        i. The constant `Cs348` represents the course `CS 348`.
    h. **Cs371**
        i. The constant `Cs371` represents the course `CS 371`.

---

2 - TYPES & ATTRIBUTES

1. Represent that a course is a CS course
    a. Name of type:
        a. `**Isa(x, y)**` represents that a variable `x` has the type `y`. From this point on, `CsCourse` represents a CS Course and `student` represents a student.
    b. Represent the following:
        a. **Isa(Cs111, CsCourse)**
        b. **Isa(Cs211, CsCourse)**
        c. **Isa(Cs348, CsCourse)**
        d. **Isa(Cs371, CsCourse)**
2. Courses can have attributes
    a. AI
        i. `**TeachesAI**` is the name of the attribute describing an AI course.
            1. **TeachesAI(Cs348)**
            2. **TeachesAI(Cs371)**
    b. Systems
        i. `**TeachesSys**` is the name of the attribute describing a Systems course.
            1. **TeachesSys(Cs321)**
    c. Theory
        i. `**TeachesThry**` is the name of the attribute describing a Theory course.
            1. **TeachesThry(Cs335)**
    d. Interfaces
        i. `**TeachesInt**` is the name of the attribute describing an Interfaces course.
            1. **TeachesInt(Cs330)**
    e. Software Development
        i. `**TeachesSD**` is the name of the attribute describing a Software Dev. course.
            1. **TeachesSD(Cs338)**

---

3 - RELATIONS

1. Name & arity of relation:

a. Name: `HasPassed(s, c)` is a relation where the student `s` has passed the course `c` (or list of courses, used in 5 – COMPLEX SENTENCES).
b. Arity: **2, because there are 2 arguments the relation takes in, `s` and `c`.**

2. 3 examples:
   a. **HasPassed(XYZ123, Cs371)** means that student `XYZ123` has passed course `Cs371`.
   b. **¬ HasPassed(XYZ123, Cs348)** means that student `XYZ123` has NOT passed course `Cs348`.
   c. **HasPassed(ABC456, Cs111)** means that student `ABC456` has passed course `Cs111`.
3. NOTE: For future reference,

   ∀s,c Isa(s, Student) ∧ Isa(c, CsCourse) ∧ HasPassed(s, c) ⇒ HasCredit(s, c),
   where `recCredFn` is a function that returns the # of credits any student `s` has earned for passing any course `c`. The function `creditsFn(c)` is defined below.

---

4 - FUNCTION

1. Name & arity of function:
   a. Name: `creditsFn(c)` is a function that returns the # of credits of any course `c` (or list of courses, used in 5 – COMPLEX SENTENCES).
   b. Arity: **1, because there is 1 argument the function takes in, `c`.**
2. Represent the following:
   a. CS 371 is a 1 credit course:
      i. **creditsFn(Cs371) = 1**
   b. CS 371 and CS 348 are worth the same # of credits:
      i. **creditsFn(Cs371) = creditsFn(Cs348)**
   c. All CS courses are worth 1 credit:
      i. **∀c Isa(c, CsCourse) ∧ creditsFn(c) = 1**

---

5 - COMPLEX SENTENCES

1. To meet the credits requirement, students need to earn 16 credits in CS courses.
   a. `CreditsMet(s)` is a property of any student `s` that means that they have met the requirement for passing 16 credits worth of courses `c`.
   b. `HasCredit(s, n)` is a relation where `s` has the number of credits `n`, the accumulation of the credits of courses `c` the student has taken. `l` is a list of courses `c` taken by the student.
   c. All other functions defined above.
   d. **∀s,c,l**
      [Isa(s, Student) ∧ Isa(c, CsCourse) ∧ HasPassed(s, c) ⇒ HasCredit(s, creditsFn(c))] ∧
      [HasCredit(s, creditsFn(l)) = HasCredit(s, 16)] ⇒ CreditsMet(s)

2. To meet the breadth requirement, students need to take at least one course from each category: AI, Systems, Theory, Interfaces, and Software Development.
   a. `BreadthMet(s)` is a property of any student `s` that has met the requirement for taking 1 course of each attribute/category (ex. TeachesAI) defined above.
   b. Explanation of formula: If `s` is a student and `c` is a CsCourse and the student has passed a class with the AI attribute & the class with the Systems attribute & …, the `BreadthMet` property applies to the student.
   c. All other functions defined above.
   d. **∀s,c Isa(s, Student) ∧ Isa(c, CsCourse) ∧**
      HasPassed(s, TeachesAI(c)) ∧
      HasPassed(s, TeachesSys(c)) ∧
      HasPassed(s, TeachesThry(c)) ∧
      HasPassed(s, TeachesInt(c)) ∧
      HasPassed(s, TeachesSD(c)) ⇒ BreadthMet(s)

3. To meet the depth requirement, students need to take six technical electives.
   ASSUMPTION: A technical elective is CS Course above the level 300, therefore
   `TechElec` is the name of the attribute describing a course.
   t = Cs321 ∨ Cs330 ∨ Cs335 ∨ Cs338 ∨ Cs348 ∨ Cs371
   a. `**DepthMet(s)**` is a property of any student `s` that means that they have met the requirement for taking at least 6 technical electives, which are any CS courses above the level 300. Explanation of formula: If `s` is a student and `t` is a tech elective CS course and the student has passed the list of 6 or more tech electives, worth 1 credit apiece anyway.
   b. `e` is the list of all elective courses `t` taken by the student.
   c. All other functions defined above.
   d. **Please read ASSUMPTION above due to vagueness of what a Technical Elective is!**

   ∀s,t,e
   [Isa(s, Student) ∧ Isa(t, CsCourse) ∧ HasPassed(s, e)] ∧
   [creditsFn(e) ≥ 6] ⇒ DepthMet(s)

4. To be granted a CS degree, a student needs to complete the credits requirement, the breadth requirement, and the depth requirement.
   a. `**RecDegree(s)**` is a property of any student `s` that has received a degree in CS.
   b. Explanation of formula: If the 3 requirements by the student are **all** met (CreditsMet, BreadthMet, and DepthMet), the student receives the degree (ie. the property `**RecDegree(s)**` applies to the student).
   c. **∀s Isa(s, Student) ∧ CreditsMet(s) ∧ BreadthMet(s) ∧ DepthMet(s) ⇒ RecDegree(s)**