A picture containing logo

Description automatically generated

**Comp 348**

**Principles of Programming Languages**

**Fall 2020**

Assignment 1

**Amr Hefny - 40082583**

**Philippe Carrier - 40153985**

**Ryan Leyland - 40015165**

**Sobhan Mehrpour Kevishahi - 40122438**

Date: 6th October 2020

Section: U

Professor: Dr. Ali Jannatpour

**Question 1 –** see Employee.java, Person.java

**Question 2 –** see Driver.java, SalaryRange.java

**Question 3**

**1. food(bread, X) = Food(Y, soup)**

Error;

Capital “F” Food is not a valid functor.

**2. Bread = soup**

Unify;

Bread = soup.

**3. Bread = Soup**

Unify;

1. **food(bread, X, milk) = food(Y, salad, X)**

Does Not Unify;

X can’t be both milk and salad.

1. **manager(X) = Y**

Unify;

The entire thing will be unified with Y.

1. **meal(healthyFood(bread), drink(milk)) = meal(X,Y)**

Unify;

X= healthyFood(bread)

Y = drink(milk)

1. **meal(eat(Z), drink(milk)) = [X]**

Does Not Unify;

LHS isn’t a list

1. **[eat(Z), drink(milk)] = [X, Y | Z]**

Unify;

X = eat(Z) = eat([])

Y = drink(milk)

Z = []

1. **f(X, t(b, c)) = f(l, t(Z, c))**

Unify;

X=l

Z=b

1. **ancestor(french(jean), B) = ancestor(A, scottish(joe))**

Unify;

A = french(jean)

B = scottish(joe)

1. **meal(healthyFood(bread), Y) = meal(X, drink(water))**

Unify;

X = healthyFood(bread);

Y = drink(water)

1. **[H|T] = [a, b, c]**

Unify;

H = a

T = [b,c]

1. **[H, T] = [a, b, c]**

Does Not Unify

LHS has 2 terms, RHS has 3 terms

1. **breakfast(healthyFood(bread), egg, milk) = breakfast(healthyFood(Y), Y, Z)**

Does Not Unify.

Y cannot be both bread and egg

1. **dinner(X, Y, Time) = dinner(jack, cook( egg, oil), Evening)**

Unify;

X = jack

Y = cook(egg, oil)

Time = Evening

1. **k(s(g), Y) = k(X, t(k))**

Unify

X = s(g)

Y = t(k)

1. **equation(Z, f(x, 17, M), L\*M, 17) = equation(C, f(D, D, y), C, E)**

Does Not Unify

D cannot be both x and 17

1. **a(X, b(c, d), [H|T]) = a(X, b(c, X), b)**

Does Not Unify

b is not a list, so cannot unify with [H|T]

**Question 4**

**1. ? field(hit\_transfer, engineering).**

Ground

field(hit\_transfer, engineering) = field(X,Y) : - course (X,Z), field(Z,Y).

X = hit\_transfer; Y = engineering

course(hit\_transfer, Z) = course(hit\_transfer, mechanical).

Z = mechanical

field(mechanical, engineering) = field(mechanical, engineering).

True.

**2. ? lab\_number(fine\_arts, X).**

Non-ground

lab\_number(fine\_arts, X) = lab\_number(fine\_arts, 10).

X = 10.

**3. ? field(computer, literature).**

Ground

field(computer, literature) = field(X,Y) : - course (X,Z), field(Z,Y).

X = computer ; Y = literature

course(computer, Z) does not unify.

False.

4**. ? course(X,Y).**

Non-ground

course(X,Y) = course(hit\_transfer, mechanical).

X = hit\_transfer;

Y = mechanical.

**5. ? student(adrian).**

Ground

student(adrian) = student(X) :- student(X,\_).

X = adrian.

student(adrian,\_) = student(adrian, web\_design).

True.

**6. ? student(anna, engineering)**.

Ground

student(anna, engineering) = student(X,Y) :- field(Z,Y), student(X,Z).

X = anna; Y = engineering.

…

field(Z, engineering) = field(X,Y) :- course(X,Z), field(Z,Y).

…

Z = hit\_transfer

student(anna, hit\_transfer) = student(anna, hit\_transfer).

True.

**7. ? student(X, engineering)**.

Non-ground

student(X, engineering) = student(X,Y):- field(Z,Y), student(X,Z).

Y = engineering; X = X;

field(Z, engineering) = … = field(mechanical, engineering) = … = field(hit\_transfer, engineering).

Z = hit\_transfer;

student(X, hit\_transfer) = student(anna, hit\_transfer).

X = anna;

**8. ? student(X, fine-arts), course(fine\_arts, Y).**

Non-ground

student(X, fine-arts) = student(X, Y) :- field(Z,Y), student(X,Z).

Y = fine-arts;

field(Z, fine-arts) = field(X,Y):- course(X,Z), field(Z,Y).

…

field(Z, fine-arts) = field(Z, Y) => fails

False; (No matches found).

**9. ? field(\_,X).**

Non-ground

field(\_,X) = field(mechanical, engineering).

X = engineering.

**10. ? lab\_number(\_,X), field(X,Y).**

Non-ground

lab\_number(\_,X) = lab\_number(mechanical, 15) => False

lab\_number(\_,X) = lab\_number(fine\_arts, 10) => False

lab\_number(\_,X) = lab\_number(X,Z) => False

False; (No matches found).

**11. ? lab\_number(X,15), field(X,Y).**

Non-ground

lab\_number(X,15) = lab\_number(mechanical, 15).

X = mechanical;

field(mechanical, Y) = field(mechanical, engineering).

Y = engineering;

Output:

X = mechanical.

Y = engineering.

**12. ? student(X), !, student(X,\_).** % note to cut here

Non-ground

student(X) = student(X) :- student(X,\_)

student(X,\_) = student(anna, hit\_transfer).

X = anna;

!

Output:

X = anna;

**13. ? student(X), student(X,\_), !.**

Non-ground

student(X) = student(X) :- student(X,\_)

student(X,\_) = student(anna, hit\_transfer).

X = anna;

student(anna,\_) = student(anna, hit\_transfer).

!

Output :

X = anna ;

**14. ? course(X,\_), \+ student(\_,X).** % \+ is for negation (not)

non-ground

course(X,\_) = course(hit\_transfer, mechanical).

X = hit\_transfer;

student(\_,hit\_transfer) = student(anna, hit\_transfer) => True

\+ True => False

…

X = web\_design; (True for adrian) => False

X = design\_methods; (True for ava) => False

X = poetry; (True for jack) => False

X = leadership; (True for lee) => False

X = biology;

student(\_,biology) = student(X,Y) :- field(Z,Y), student(X,Z).

Y = biology;

field(Z, biology) = field(X,Y) : - course(X,Z), field(Z,Y).

…

student(\_,biology) => false

\+ false => True.

Output:

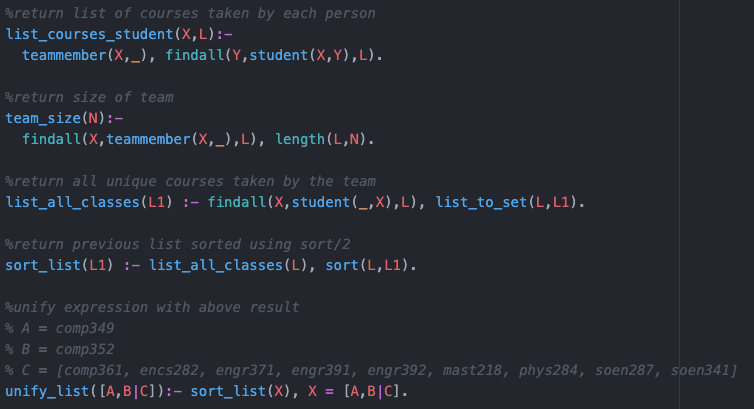
X = biology.

**Question 5 –** see question5.pl

Text, chat or text message

Description automatically generatedText

Description automatically generated



A = comp348

B = comp352

C = [comp361, encs282, engr371, engr391, engr392, mast218, phys284, soen287, soen341]

**Question 6 –** see question6.pl

Text

Description automatically generated

?- accept([a, a, b]).

accept([a, a, b]). will return true

It can be seen from the diagram that the FSM starts at s1.

Traversing along the paths a, a and then b will result in the return to s1, which is a final state

Therefore, will return true

**Question 7 –** see question7.plText

Description automatically generatedText

Description automatically generatedText

Description automatically generatedA picture containing text

Description automatically generated

?- circuit(0,1,0,1). 🡪 “a=1 b=0 c=1 d=1 e=0 f=1 g=1 Output is 5.”

**Question 8 –** see question8.pl

Text

Description automatically generated

**Question 9 –** see question9.pl

Text

Description automatically generated