Homework 3

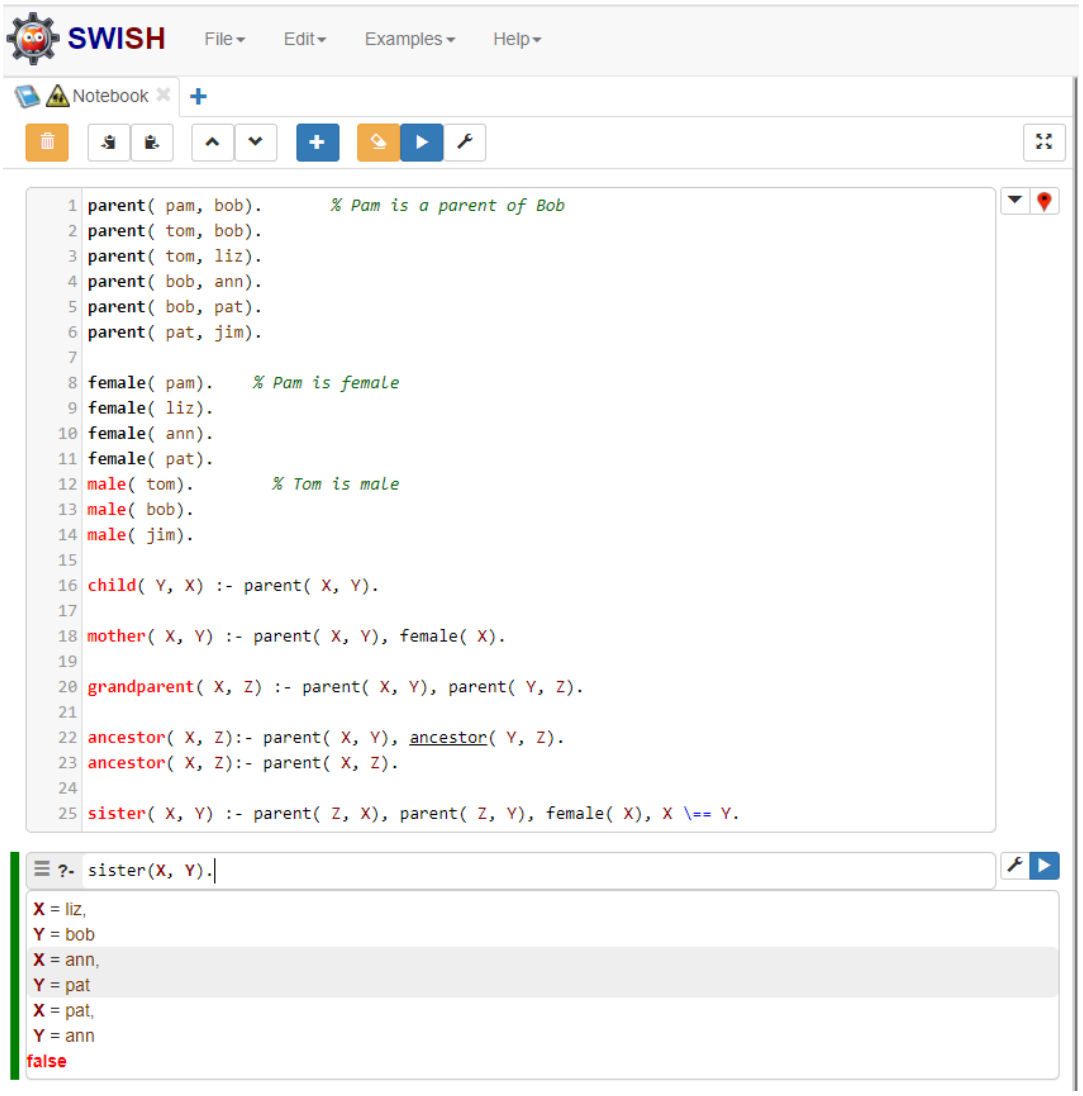
FULYA KOCAMAN
CWID: 803023878

Due: **Check date on Canvas**. Prepare your answers as a **single PDF file**.

1. The attached `family.pl` Prolog program describes a set of facts and rules to identify some family relationships. `parent(pam, bob)` means that pam is the parent of bob.

   a. Add one new rule to family.clp to print a list of all (sister, person) pairs. X is a sister of Y if they have the same parent and X is female. [Show rule and output of the program]

sister(X, Y) :- parent(Z, X), parent(Z, Y), female(X), X \== Y.

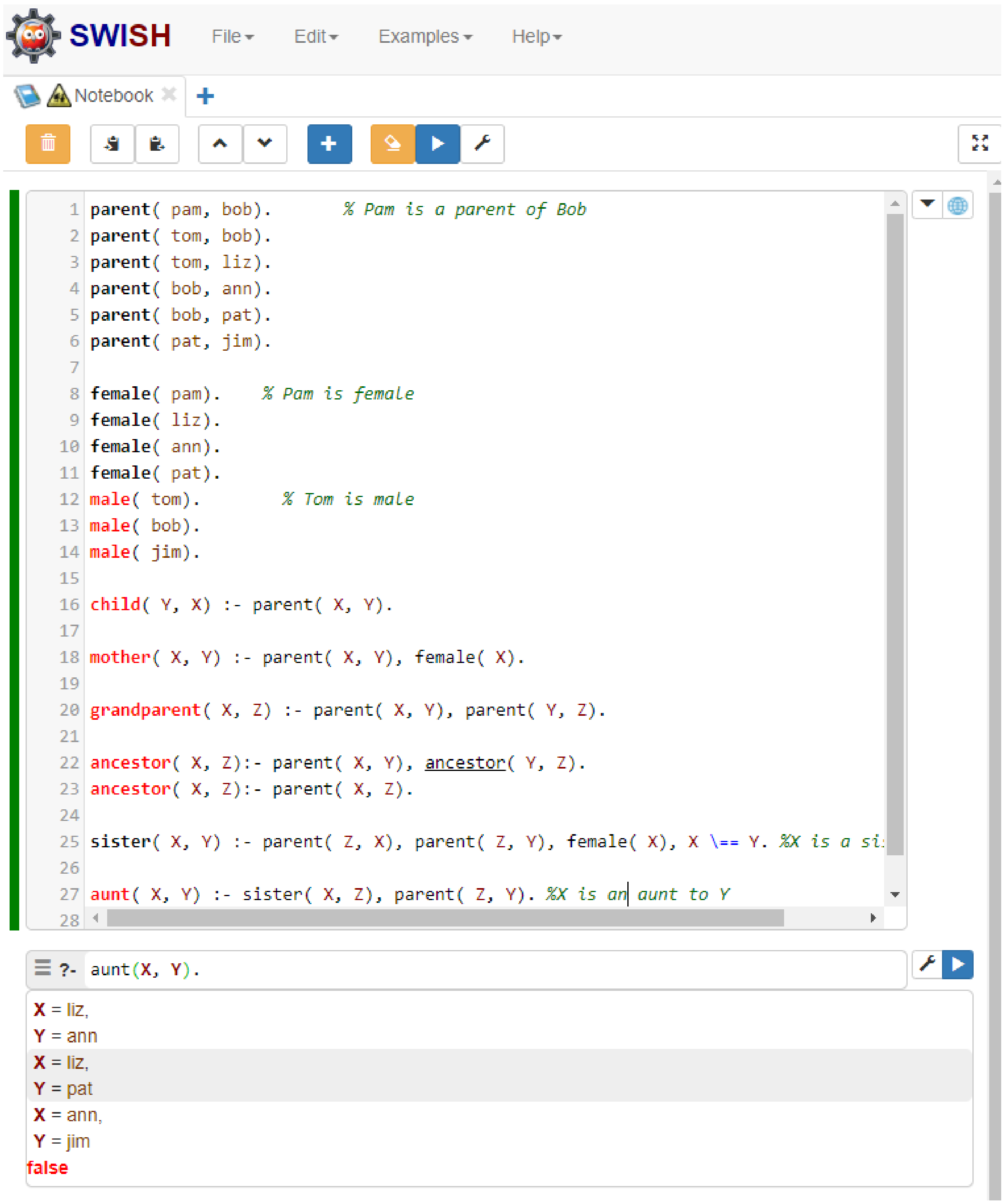


```
parent( pam, bob).       % Pam is a parent of Bob
parent( tom, bob).
parent( tom, liz).
parent( bob, ann).
parent( bob, pat).
parent( pat, jim).

female( pam).    % Pam is female
female( liz).
female( ann).
female( pat).
male( tom).        % Tom is male
male( bob).
male( jim).

child( Y, X) :- parent( X, Y).

mother( X, Y) :- parent( X, Y), female( X).

grandparent( X, Z) :- parent( X, Y), parent( Y, Z).

ancestor( X, Z):- parent( X, Y), ancestor( Y, Z).
ancestor( X, Z):- parent( X, Z).

sister( X, Y) :- parent( Z, X), parent( Z, Y), female( X), X \== Y.
```

```
?- sister(X, Y).
X = liz,
Y = bob
X = ann,
Y = pat
X = pat,
Y = ann
false
```

b. Add one new rule to family.clp to print a list of all (aunt, nephew/niece) pairs. [Show rule and output of the program]

aunt(X, Y) :- sister(X, Z), parent(Z, Y).   %X is an aunt to Y



```
parent( pam, bob).        % Pam is a parent of Bob
parent( tom, bob).
parent( tom, liz).
parent( bob, ann).
parent( bob, pat).
parent( pat, jim).

female( pam).    % Pam is female
female( liz).
female( ann).
female( pat).
male( tom).        % Tom is male
male( bob).
male( jim).

child( Y, X) :- parent( X, Y).

mother( X, Y) :- parent( X, Y), female( X).

grandparent( X, Z) :- parent( X, Y), parent( Y, Z).

ancestor( X, Z):- parent( X, Y), ancestor( Y, Z).
ancestor( X, Z):- parent( X, Z).

sister( X, Y) :- parent( Z, X), parent( Z, Y), female( X), X \== Y. %X is a si

aunt( X, Y) :- sister( X, Z), parent( Z, Y). %X is an aunt to Y
```

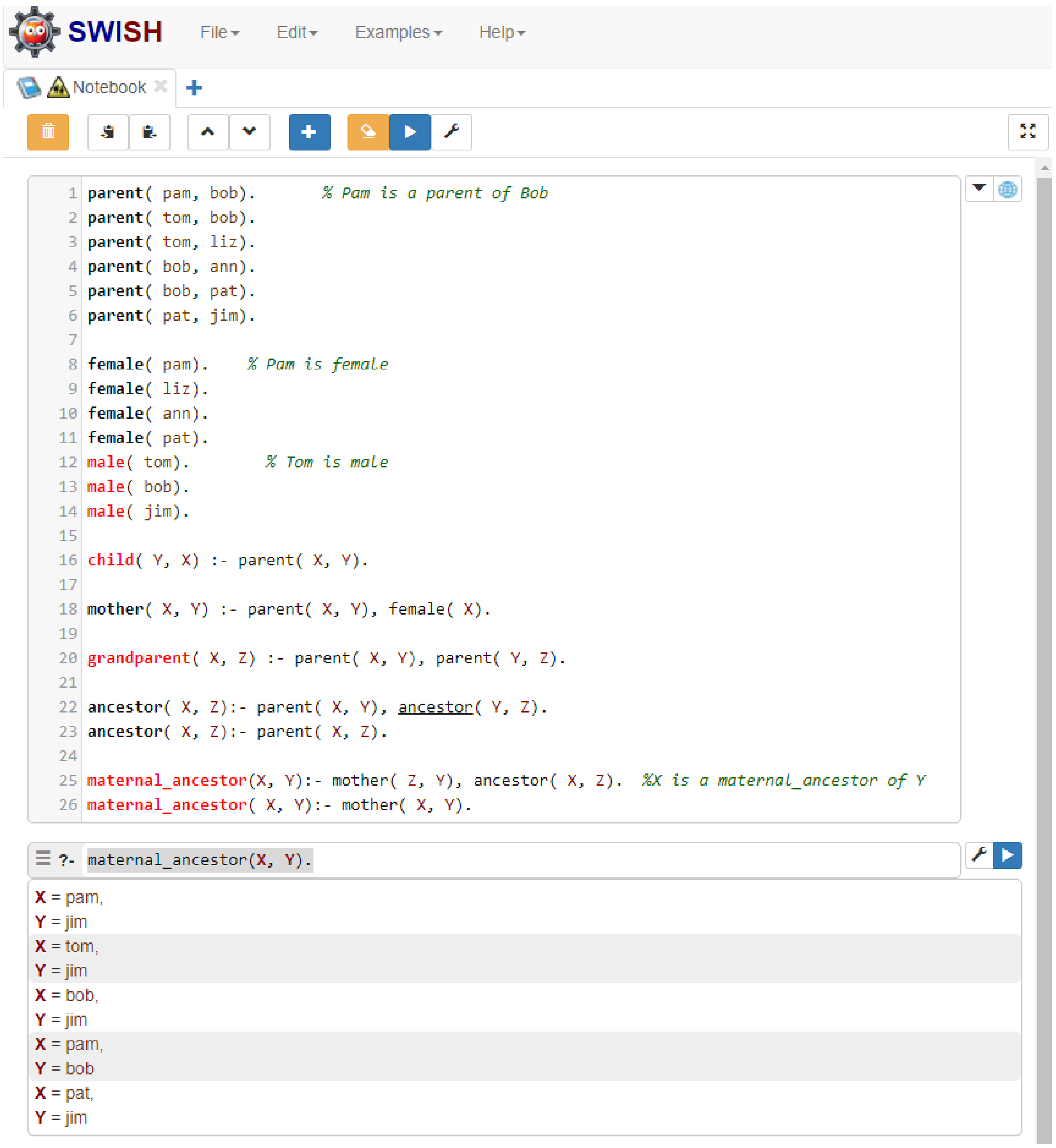
```
?- aunt(X, Y).
```

```
X = liz,
Y = ann
X = liz,
Y = pat
X = ann,
Y = jim
false
```

c. Add one new rule to family.clp to print a list of all (maternal_ancestor, person) pairs. A maternal ancestor is an ancestor via the person's mother's side, e.g., mother, mother's father, mother's grandfathers, mother's grandmothers, ... [Show rules and output of the program]

maternal_ancestor(X, Y):- mother(Z, Y), ancestor(X, Z). %X is a maternal_ancestor of Y

maternal_ancestor(X, Y):- mother(X, Y).



```
parent( pam, bob).      % Pam is a parent of Bob
parent( tom, bob).
parent( tom, liz).
parent( bob, ann).
parent( bob, pat).
parent( pat, jim).

female( pam).    % Pam is female
female( liz).
female( ann).
female( pat).
male( tom).        % Tom is male
male( bob).
male( jim).

child( Y, X) :- parent( X, Y).

mother( X, Y) :- parent( X, Y), female( X).

grandparent( X, Z) :- parent( X, Y), parent( Y, Z).

ancestor( X, Z):- parent( X, Y), ancestor( Y, Z).
ancestor( X, Z):- parent( X, Z).

maternal_ancestor(X, Y):- mother( Z, Y), ancestor( X, Z).  %X is a maternal_ancestor of Y
maternal_ancestor( X, Y):- mother( X, Y).
```

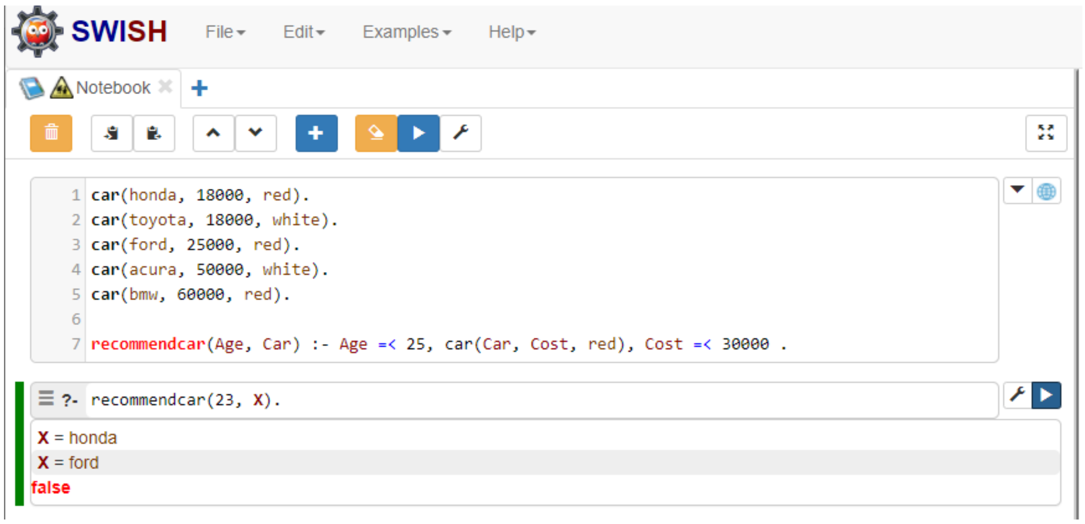
```
?- maternal_ancestor(X, Y).
```

```
X = pam,
Y = jim
X = tom,
Y = jim
X = bob,
Y = jim
X = pam,
Y = bob
X = pat,
Y = jim
```

2. The attached `cars.pl` Prolog program describes a list of cars (brand, price, color) and a rule that recommends a car that costs less than $30,000 for a person younger than 25 years of age [same problem as in the CLIPS question from the last homework]. Note that you "run" this program by posing a query: `recommendcar(23, X).`

   a. Modify the rule such that the recommendation for a person younger than 25 years is a car that costs less than $30,000 and red in color. [Show modified rule and output of the program]

recommendcar(Age, Car) :- Age =< 25, car(Car, Cost, red), Cost =< 30000 .

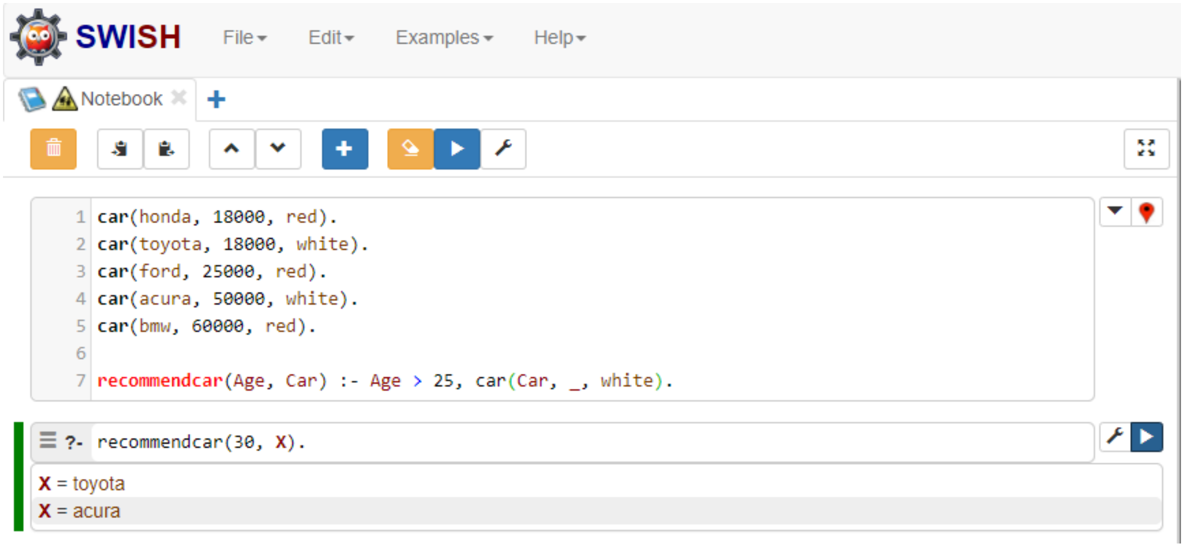


```
car(honda, 18000, red).
car(toyota, 18000, white).
car(ford, 25000, red).
car(acura, 50000, white).
car(bmw, 60000, red).

recommendcar(Age, Car) :- Age =< 25, car(Car, Cost, red), Cost =< 30000 .
```

```
?- recommendcar(23, X).
X = honda
X = ford
false
```

   b. Change the program such that the recommendation for a person older than 25 years is a white car. [Show rules and output of the program]

recommendcar(Age, Car) :- Age > 25, car(Car, _, white).



```
car(honda, 18000, red).
car(toyota, 18000, white).
car(ford, 25000, red).
car(acura, 50000, white).
car(bmw, 60000, red).

recommendcar(Age, Car) :- Age > 25, car(Car, _, white).
```

```
?- recommendcar(30, X).
X = toyota
X = acura
```