## Question 1 - 15 points

**Resolution**

**1a. (7 points)** Which sentences, if any, do you obtain by applying the resolution inference rule to the following pair of sentences? **Do not do any simplifications to either the input or the output sentences, just blindly apply the resolution rule.** The sentences use first-order logic, x,y, and z are variables, John is a constant, pred1, pred2 and pred3 are predicates. If resolution cannot be applied, clearly explain why.

```
not(pred1(x, y)) or pred2(x) or pred3(y)
pred1(z, John)
```

pred2(z) or pred3(John)

**1b. (8 points)** Which sentences, if any, do you obtain by applying the resolution inference rule to the following pair of sentences? **Do not do any simplifications to either the input or the output sentences, just blindly apply the resolution rule.** The sentences use first-order logic, x and y are variables, John and Mary are constants, pred1, pred2 and pred3 are predicates. If resolution cannot be applied, clearly explain why.

```
not(pred1(x, y)) or pred2(x) or pred3(x)
pred1(Mary, John) or pred3(John)
```

pred2(Mary) or pred3(Mary) or pred3(John)

## Question 2 - 15 points

**Pattern Matching**

Write the most general unifier for each of the following pairs of first-order logic sentences. The following conventions hold:
F and G are predicates.
x, y, z are variables.
John and Mary are constants.

**2a. (5 points)**

```
F(x, y) and F(y, z)
F(John, Mary) and F(Mary, John)
```

{x/John, y/Mary, z/John}

**2b. (5 points)**

```
F(x, y) and F(G(Mary), G(x))
F(John, Mary) and F(z, G(John))
```

{x/John, y/Mary, z/G(Mary)}

**2c. (5 points)**

```
F(x, y, G(z))
F(G(y), Mary, G(John))
```

{x/G(Mary), y/Mary, z/John}

## Question 3 – 15 points

### Conjunctive Normal Forms

**3a. (7 points)** Put the following propositional-logic knowledge base in conjunctive normal form:

((NOT A) AND B) => (C OR A)
A AND B AND (C OR D)

A OR (NOT B) OR C OR A
A
B
C OR D

**3b. (8 points)** Suppose that some knowledge base contains various propositional-logic sentences that utilize symbols A, B, C, D, E (connected with various connectives). There are only two cases when the knowledge base is false:
- First case: when A is true, B is true, C is false, D is false, E is false.
- Second case: when A is true, B is false, C is false, D is false, E is true.
In all other cases, the knowledge base is true. Write a conjunctive normal form for the knowledge base. (Hint: there is a much simpler and quicker way to get the right answer, compared to considering all 32 entries in the truth table).

(NOT A) OR (NOT B) OR C OR D OR E
(NOT A) OR B OR C OR D OR (NOT E)

## Question 4 - 15 points

### Logical Equivalence

Determine if the following pairs of sentences are logically equivalent, meaning that one is true if and only iff the other is true. You do not have to justify your answer.

**4a. (5 points)** Propositional logic.

```
A or B or not(B) or C
A or B or (C <=> C)
```

Logically equivalent, they are both always true

**4b. (5 points)** First-order logic, x and y are variables, f is a predicate.

```
for-every x, for-every y: f(x, y)
for-every x, for-every y: f(y, x)
```

Logically equivalent, "for-every x, for-every y" is the same as for-every y, for-every x, so:

for-every x, for-every y: f(x, y) is the same as
for-every y, for-every x: f(x, y)

By consistently replacing x with y and y with x in "for-every y, for-every x: f(x, y)", we obtain "for-every x, for-every y: f(y, x)"

**4c. (5 points)** Propositional logic.

```
(A and B) => (E and G)
not(A) or not(B) or (E and G)
```

logically equivalent, the second statement is obtained from the first one by applying the rule that "X => Y" is the same as "(not X) or Y"

## Question 5 - 10 points

**Partially-Ordered Plans**

The actions for the traditional block-world problem are:

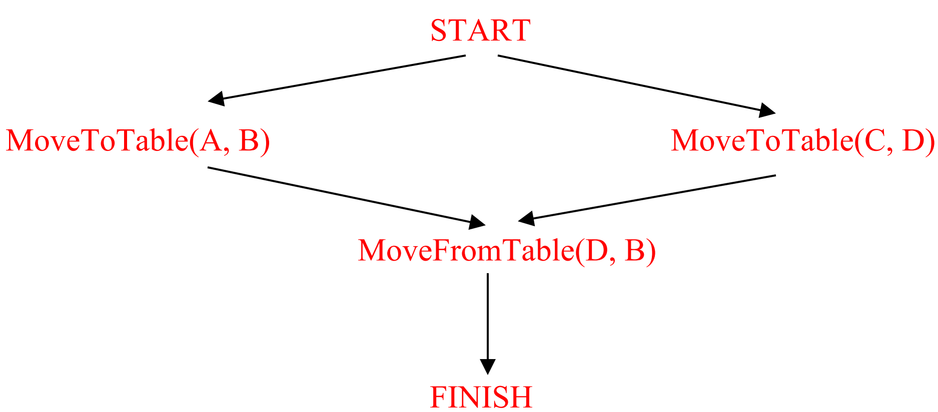
Action: **Move(p,x,y)**
Precond: on(p,x) and clear(y) and clear(p)
Effect: on(p,y) and clear (x) and not(clear(y)) and not(on(p,x))

Action: **MoveFromTable(p,y)**
Precond: on(p,Table) and clear(y) and clear(p)
Effect: on(p,y) and not(clear(y)) and not(on(p,Table))

Action: **MoveToTable(p,x)**
Precond: on(p,x) and clear(p)
Effect: on(p,Table) and clear (x) and not(on(p,x))

Consider the following initial state and goal:
- Initial state: on(A, B), on(C,D), clear(A), clear(C), on(B, Table), on(D, Table)
- Goal: on(D, B)

Give a partially ordered plan to achieve the goal from the initial state, such that the plan is minimally ordered (i.e., no action A is constrained to happen before another action B unless it is necessary for plan correctness). In specifying the plan, specify the name and parameter of every action in the plan, but do NOT list the preconditions and effects of each action.



## Question 6 - 15 points

**Conditional Planning**

We modify the traditional block-world actions, to introduce uncertainty. In particular, while actions **MoveFromTable** and **MoveToTable** remain as in **Question 5**, action **Move** is modified. Specifically, the actions are now defined as follows:

Action: **Move(p,x,y)**
Precond: on(p,x) and clear(y) and clear(p)
Effect: (on(p,y) and clear (x) and not(clear(y)) and not(on(p,x))) or
(on(p,Table) and clear (x) and not(on(p,x)))

Action: **MoveFromTable(p,y)**
Precond: on(p,Table) and clear(y) and clear(p)
Effect: on(p,y) and not(clear(y)) and not(on(p,Table))

Action: **MoveToTable(p,x)**
Precond: on(p,x) and clear(p)
Effect: on(p,Table) and clear (x) and not(on(p,x))

Consider the following initial state and goal:
- Initial state: on(A, B), on(B, C), clear(A), on(C, Table)
- Goal: on(C, B), on(B, A)

Is it possible to come up with a finite plan that guarantees success in this case? If not, why not? If yes, specify the plan in partial or complete order. In specifying the plan, specify the name and parameter of every action in the plan, but do NOT list the preconditions and effects of each action.

Yes, it is possible, here is a plan:

MoveToTable(A, B)
MoveToTable(B, C)
MoveFromTable(B, A)
MoveFromTable(C, B)

## Question 7 - 15 points

### Defining Actions

We define a new version of the block world, where we can move block **p** from **x** to **y** even if **p** is not clear. In that case, all blocks that are on top of **p** move together with **p**, and remain on top of **p** and in the same order as before (and, naturally, the topmost block of the pile remains clear). Note that **x** or **y** may be the table. Everything else is the same as in the traditional block world of question **5**, and there is NO uncertainty in the effects of any action. In other words, **y** still needs to be clear before the move, **x** is clear after the move, and the table is always clear.

**7a (10 points).** Define appropriate actions for this problem. The set of actions should be complete enough to allow optimal plans to be constructed in all possible cases. Optimal plans are the ones that consist of the smallest possible number of actions, given the initial state and goal.

Action: **Move(p,x,y)**
Precond: on(p,x) and clear(y)
Effect: on(p,y) and clear (x) and not(clear(y)) and not(on(p,x))

Action: **MoveFromTable(p,y)**
Precond: on(p,Table) and clear(y)
Effect: on(p,y) and not(clear(y)) and not(on(p,Table))

Action: **MoveToTable(p,x)**
Precond: on(p,x)
Effect: on(p,Table) and clear (x) and not(on(p,x))

**7b (5 points).** Given the specifications for this modified block world, provide (in partial or complete order) an optimal (i.e., shortest possible) plan for the following problem:

- Initial state: clear(A), on(A, B), on(B, C), on (C, D), on (D, E), on(E, Table)
- Goal: on(A, B), on(B, E)

In specifying the plan, specify the name and parameter of every action in the plan, but do NOT list the preconditions and effects of each action.

MoveToTable(D, E)
Move(B, C, E)