Mid Term Questions :

- from final 2011 :

** Question 3 :

In the answer to this question, you may wish to use some of the notation used in Question 1.

- a) It has been argued that the multi-agent systems paradigm emerged from five ongoing trends
  in computing. Explain what you understand these trends to be. ✅
  [5 marks]

** answer :

1. Ubiquity : lower cost of computing resources make it economic and allow it's existence in our life to increase more and more .
2. Delegation : now we delegate the computer to perform our tasks .. also in critical tasks .
3. interconnection : now computers are connected together in large distributes systems , no stand alone computer .
4. human orientation : programming languages continously updated to be so close to human languages .
5. intellegence : complexity of tasks that computer can do increase more and more .

- b) Explain what is meant by a predicate task specification, and how such a specification
  relates to utility functions over runs. ✅
  [5 marks]

- predicate task specification : special case of utility function that assign run to 0 or 1 , 0 mean run will fail , 1 mean run will success .

- c) Explain what is meant by an achievement goal. ✅
  [5 marks]

- Give agent good states that try to achieve while run .

- d) Explain what is meant by a maintenance goal. ✅
  [5 marks]

- give agent bad states which avoid while run .

- e) Two key problems that arise in deductive/symbolic agent architectures are transduction
  and representation/reasoning. Explain what you understand by these problems. ✅
  [5 marks]

answer :

1. transduction problem : problem to convert real world to symbols to understand it .
2. representation / reasoning : how can we symbolically represent complex real world .

- ****************************** final 2009 : *******************************

b) Shoham’s Agent Oriented Programming paradigm, as exemplified in the Agent0 language , was a first attempt to program computers using the intentional stance. Figure 1 shows a

fragment of Agent0 code. With reference to this code, describe:

```
(defagent plane
  :timegrain 10
  :beliefs '(
        (1 (at 100 100))
        (1 (max-speed 5))
        (1 (CMT plane plane
              (INFORM 2 world (2 (plane p1 100 100))))))

  :commit-rules
  '(

  ( (control REQUEST (DO ?time (be-at ?gx ?gy)))
        () ;; no mental conditions
        control
        (DO now (cap-check ?time 'be-at ?gx ?gy)) )


  ( (control UNREQUEST (DO ?time (be-at ?gx ?gy)))
        (CMT control (DO ?time2 (be-at ?gx ?gy)))
        plane
        (DO now (uncommit ?time2 'be-at ?gx ?gy)) )

  ( () ;; no message condition
        (and (B (now (lowfuel)))
              (CMT ?agent (DO ?time2 (be-at ?z1 ?z2))))
        plane
        (REQUEST now control (UNREQUEST now plane
              (DO ?time2 (be-at ?z1 ?z2)))) )
 [...]
  )          ; ends commitment rules
)       ; ends defagent
```

a. how programs in Agent0 are constructed ✅

Agent has 4 main things in Agent0 program :

1. beliefs
2. intentions - it's own motivations .
3. commitments that try to achieve .
4. messages that communicate via .

b. the overall control loop for Agent0

c. how agents in Agent0 communicate ✅

- agents in Agent0 are communicating via sending messages which is one of 3 types :

1. Request - to commit an action .
2. un-Request - refrain from action .
3. informs - pass information .

- **************************** final 2008 : *******************************

a. The following pseudo-code characterises the action selection process of a logic-based agent.✅

$$
\begin{array}{l}
\text{for each } \alpha \in Ac \text{ do} \\
\quad\quad \text{if } \Delta \vdash_\rho Do(\alpha) \text{ then} \\
\quad\quad\quad\quad \text{return } \alpha \\
\quad\quad \text{end-if} \\
\text{end-for} \\
\text{for each } \alpha \in Ac \text{ do} \\
\quad\quad \text{if } \Delta \not\vdash_\rho \neg Do(\alpha) \text{ then} \\
\quad\quad\quad\quad \text{return } \alpha \\
\quad\quad \text{end-if} \\
\text{end-for} \\
\text{return } \texttt{null}
\end{array}
$$

Describe how this loop works, explaining in particular the role of ρ, Δ, `ρ, and Do(α), and the limitations of such an approach for run-time decision-making. ✅

- at first loop agent try to get optimal action “explicit prescribed action ” .
- if agent didn’t find that optimal action , then it will enter second loop which it try to get not excluded action .
- ρ → the theory of agent “set of rules” that we choose optimal action depending on .
- Δ → agent logical database that contain all agent belifes .
- `ρ -> prove that Do alpha excluded from agent beliefes .
- Do(α) → perform action alpha “α” .
- limititions :

1. how can we convert real world seeing to logical representation .
2. decision making using first order logic is undecidable .

c) With particular reference to the outline architecture above, explain what you understand

by the term calculative rationality. ✅

- decision making assume static environment .

---

d) Explain what you understand by bounded optimality. ✅

- some agent can’t be implemented on the computer because they need more resources , so we replaces optimal agent equation to another equation called bounded optimal agent equation .

- ****************************** final 2007 : *******************************

a) Briefly define and explain, with examples where appropriate, the properties you would
expect an intelligent agent to exhibit. ✅
[10 marks]

1. Autonomy : agent ability to achieve it’s goals independently .
2. Reactive : agent ability to deal with all changes in environment .
3. Pro-active :
4. social ability : Agent ability to communicate , cooperate and negotiate with other agents .

---

b) Briefly explain what you understand by the intentional stance and an intentional system,
and explain the role that the intentional stance plays in the agent-oriented programming
paradigm, as typified by the AGENT0 programming language.
[5 marks]

---

c) The issue of telling an agent what to do (without telling it how to do it) is a central problem
in multiagent systems. A number of approaches to this problem have been proposed, chief
among them being the following:

- utility functions over states;

- : calculate utility of each state in environment individually
- adv : give utility number for each env state
- dis adv : short term and future states not depend on current value .
- utility functions over runs
- calculate utility of run .
- adv : is long term approach
- dis adv : from where we get these numbers and it’s hard to formulate these terms .
- predicates over runs;
- assign one of 2 values for run 0 or 1 . 0 mean this run will fail , 1 mean this run will success .
- adv : easier to assign values to run (0,1) .
- achievement tasks;
- Give agent good states that try to achieve while run .
- maintenance tasks.
- Give agent bad states that avoid while run .
- Explain what you understand by each of these approaches ✅ , making clear the relative advantages and disadvantages of each ✅ and how these approaches relate to one-another. ✅
- how these approaches relate to one-another :

1. first we need to tell the agent what to do with it telling him how to do
2. we used states utility function , but we found that it was short term.
3. then we used run utility function , it okay was a long term but it cause another problems as from where we get the numbers and how will we formulate these tasks into these terms .
4. the we get special case of utility function , it was task predicate specification , that assign 0 or 1 to the run . , which make the process easier .
5. then we get Achievement task which provide set of runs that agent try to achieve to success

6. and also Maintainance task that provide set or bad runs that agent must avoid to avoid faliure .

- Illustrate
  your answer with examples as appropriate.
- مش مهم .. هو عايزك تكتب المعادلات بس مفيش وقت تحفظ القرف دا

- ***************************** final 2004 : ******************************

a. The issue of telling an agent what to do (without telling it how to do it) is a central problem
   in multiagent systems. A number of approaches to this problem have been proposed, chief
   among them being the following:

- utility functions over states;
- utility functions over runs;
- predicates over runs;
- achievement tasks;
- maintenance tasks.
- Explain what you understand by each of these approaches, making clear the relative advantages and disadvantages of each and how these approaches relate to one-another. Illustrate
  your answer with examples as appropriate.
  [5 marks]
- نفس اجابه اللي فوقه

---

b. Some researchers have argued that the notion of bounded optimality is a more appropriate measure of optimality than that of simply maximising expected utility. Present these
arguments as you understand them, and formally define the notion of a bounded optimal
agent.