Q1: La différence entre un agent est un objet est que l'agent fonctionne de manière autonome: il peut s'appeler et agir au sein d'un système sans avoir besoin d'être appelé de manière externe, il est réactif et proactif, peut-être self-aware. (Un agent est en quelque sorte un objet qui en interne peut agir de lui-même). Un objet ne peut refuser l'appel de l'une de ses méthodes et n'est pas intelligent.

Q2: 2 exemples de mécanismes d'auto-organisation:
- Une colonie de fourmis qui cherche de la nourriture et la rapporte à la fourmilière. (biologique)
- Un banc de poisson qui se regroupe pour se protéger de prédateurs. (biologique)

Q3: Les différences entre Jade & Netlogo:
"NetLogo est un langage et une plateforme de développement dédiée à la simulation de modèles individus centrés (individu=agent).

Différent d'une plateforme de type JADE où les agents
sont «réels» (ils peuvent circuler sur un réseau informatique, «réels» (ils peuvent circuler sur un réseau informatique, interagir à distance sur le réseau, ...)"

"JADE is not a
simulation framework; it has no scheduler, nor a notion of a "clock",
i.e., for the agents to maintain the same view of "time", so you
would have to code any simulation infrastructure."

Platform	Primary Domain	License	Programming Language	Operating System	User Support	FIPA Compliant	GIS Capabilities	3D Capabilities
NetLogo	Social and natural sciences; Help beginning users get started authoring models	GPL	NetLogo	Any Java Virtual Machine, version 5 or later.	Documentation; FAQ; selected references; tutorials; third party extensions; defect list; mailing lists	Unknown	Yes	Yes
JADE	Distributed applications composed of autonomous entities	LGPL version 2	Java	Any Java Platform	FAQ; mailing list; defect list; tutorials; API; documentation	Yes	Unknown	Unknown

Q4: Les différences entre systèmes complexe et compliqué :
- Un système compliqué est un système difficile à comprendre, qui une fois compris et maîtrisé permet de prévoir l'évolution qu'il suivra. (Air bus)
- Un système complexe est un système qui aura un comportement impossible à prévoir à l'avance et qui est réactif à son environnement. Son fonctionnement est régi par un nombre limité de règles, simples mais auxquelles on ne peut déroger. (plat de spaghettis/nuée d'oiseau)

Q5 : Systèmes Experts :
- Fournissent des conseils ; raisonnent mais n'agissent pas en perception de…, ni en action sur … ; pas pro-actifs ; pas autonome : nécessitent des interventions et des instructions. ; pas adaptatifs ; pas distribués ; les agents peuvent être dotés de règles inférentielles utilisées dans les SE ; l'évolution des SE vers les systèmes multi-experts fut à l'origine des SMA (qui y ajoutent : le Machine learning/human-computer interaction : adaptation/personalization | l'Artificial intelligence, software engineering | l'Objet (un peu d'autonomie, de ré-utilisabilité, interaction)

Q6 : les liens qui unissent ces deux domaines sont très étroits: les SMA utilisent un grand nombre de techniques provenant des systèmes distribués, en se situant à un niveau plus conceptuel et méthodologique. Certains algorithmes qu'emploient les SMA, tels que les algorithmes d'allocation de tâches, et certains formalismes de représentation, tels que les réseaux de Petri, font partie du bagage naturel des systèmes distribués. De ce fait, les SMA et les systèmes distribuées entretiennent plus un rapport synergique qu'une véritable opposition, les seconds apportant leur rigueur et leurs algorithmes, les premiers donnant une conceptualisation et une approche plus générale, moins centrée sur les mécanismes d'exécution. Il n'est donc pas étonnant que des recherches en cours tendent vers une collaboration de plus en plus ´etroites entre les spécialistes des systèmes distribuées
et ceux des systèmes multi-agents.

;a fix mettre a 25% les voisin des voisin et evapFact et aussi je sais pas pk (set odeurs (qtit / 2)]) ca veux pas du coup j'ai fais ca voir en bas ou il y a conversion
patches-own [odeurs] *;variable pour chaque patch*
globals [evap conversion] *;variable global*
turtles-own [qtite]

to PREPARER *; le setup qui permet d'ini tout les paramètres*
  clear-all
  prepare-patches
  cree-lapin

```
end

to cree-lapin ;je crée des lapin et je mets random le qtite pour chaque lapin
  create-turtles 10
  ask turtles [
     setxy random-xcor random-ycor
      set qtite random 100
     ]
end

to prepare-patches ;Je prépare les patches chaque case je mets à 0 la variable global de l'odeur qui sera laissée par un lapin
  ask patches [
     set pcolor green
     set odeurs 0 ;  mais a 0 chaque patch
     set plabel odeurs
    ]
end

to GO ;Go sera la fonction en boucle
  lapin-move
  evapFact
end

to lapin-move
  ask turtles [
    show "Voisin :"
    show  [odeurs] of neighbors ;print test les odeurs des 8 cases
    let target max-one-of neighbors [odeurs] ;je prend les coordonnée du max
    let valuermax max [ odeurs] of neighbors ;je prend la valeur la plus grande

    ifelse(valuermax = 0) ;tout le monde a 0 random move[
      show "Move random"
       right random 360
       forward 1
        show   qtite
       set conversion qtite
      ask neighbors [set odeurs ((conversion / 2 ) + odeurs)] ;50%
    ]
    [
      show "valeur max :"
      show   valuermax
      face target
      move-to target
       ask neighbors [set odeurs ((conversion / 2 ) + odeurs)] ;50%
    ]
  ]
  ;print :
    ask patches [
     set plabel odeurs
    ]
end

to evapFact ;une erreur à fix ne pas faire -1 sur le patch ou il se trouve à fix
  ask patches [
    ifelse (odeurs = 0)[
      set odeurs 0
    ]
    [
     set odeurs (odeurs - 1)
    ]
  ]
end
```