A) Para cada una de las siguientes actividades, describa en PEAS el entorno de la tarea y caracterizarlo en términos de las propiedades enumeradas.

- Jugar al CS (o cualquier otro 3d Shooter).
- Explorar los océanos.
- Comprar y vender tokens crypto (alguno).
- Practicar el tenis contra una pared.
- Realizar un salto de altura.
- Pujar por un artículo en una subasta.

B) Implementar un simulador que determine la medida de rendimiento para el entorno del mundo de la aspiradora según las siguientes especificaciones:

1. La medida de rendimiento premia con un punto al agente por cada recuadro que limpia (aspira) en un período de tiempo concreto, a lo largo de una «vida» de 1000 acciones.
2. La «dimensión» de la grilla se conoce a priori pero la distribución de la suciedad y la localización inicial del agente no se conocen (aleatorio). Las cuadrículas se mantienen limpias y aspirando se limpia la cuadrícula en que se encuentra el agente
3. Las acciones Izquierda, Derecha, Arriba, Abajo mueven al agente en dichas direcciones, excepto en el caso en que lo pueda llevar fuera de la grilla.
4. Las acciones permitidas son:
    a. Arriba
    b. Abajo
    c. Izquierda
    d. Derecha
    e. Limpiar (aspirar)
    f. NoHacerNada
5. El agente percibe su locación y si esta contiene suciedad

Posibles interfaz a utilizar

```
class Environment:
      def __init__(self,sizeX,sizeY,init_posX,init_posY,dirt_rate)
      def accept_action(self,action):
      def is_dirty(self):
      def get_performance(self):
      def print_environment(self):
```

C) Implementar un agente reflexivo simple para el entorno de la aspiradora del ejercicio anterior.

Posible interfaz para el Agente

```
class Agent:
      def __init__(self,env): #recibe como parámetro un objeto de la clase
Environment
      def up(self):
      def down(self):
      def left(self):
      def right(self):
      def suck(self): # Limpia
      def idle(self): # no hace nada
      def perspective(self,env): #sensa el entorno
      def think(self): # implementa las acciones a seguir por el agente
```

D) Evaluar el desempeño del agente agente reflexivo (medida de desempeño y unidades de tiempo consumidas) para:

1. Entornos de : 2x2, 4x4, 8x8, 16x16, 32x32, 64x64, 128x128
2. Porcentaje de Suciedad en el ambiente: 0.1, 0,2 0,4, 0.8
3. Repetir 10 veces cada combinación.

Nota:Se recomienda elaborar una tabla en google sheets (o algo similar) en donde se presente los resultados en términos de la medida de rendimiento para cada uno de los casos. Esto luego se podrá utilizar para realizar alguna visualización de los resultados.

E) Repetir el procedimiento descrito en el punto C, para el caso de un agente con comportamiento totalmente aleatorio. En cada periodo de tiempo, el agente toma una acción al azar.

F) Responder preguntas 2.10 y 2.11 de AIMA 3era Edición.

G) Desarrollar un agente reflexivo que funcione para el entorno FrozenLake de la biblioteca Gymnasium. **(OPCIONAL)**

Descripción:
`FrozenLake` es un entorno de cuadrícula donde un agente debe navegar desde un punto de inicio hasta una meta, evitando caer en los agujeros del hielo.

La medida de rendimiento recompensa al agente con un punto por cada casilla que cruza exitosamente sin caer en un agujero durante un período de tiempo específico, a lo largo de una *vida* de 1000 acciones.

La dimensión de la cuadrícula se conoce de antemano, pero la distribución de los agujeros y la ubicación inicial del agente no se conocen (son aleatorios). Las casillas permanecen sólidas una vez cruzadas, y el objetivo del agente es alcanzar la meta sin caer en un agujero.

Las acciones "*Left*", "*Right*", "*Up*" y "*Down*" mueven al agente en esas direcciones, excepto cuando dicho movimiento sacaría al agente de la cuadrícula o lo llevaría a un agujero.

Las acciones permitidas son:

- 0: Move left
- 1: Move down
- 2: Move right
- 3: Move up

El agente percibe su ubicación y si esa casilla contiene un agujero o es la meta.

A continuación una porción de código que implementa un agente aleatorio en el entorno **frozenLake**. Se puede tomar como guía para implementar un agente reactivo que funcione sobre el mismo entorno.

```python
import gym

# Create the FrozenLake environment
env = gym.make('FrozenLake-v0')

# Number of episodes to run
num_episodes = 1000
total_reward = 0

for episode in range(num_episodes):
    state = env.reset()  # Reset environment and get initial state for the new episode
    done = False

    while not done:
        action = env.action_space.sample()  # Randomly choose an action
        next_state, reward, done, _ = env.step(action)  # Take the action in the environment

        total_reward += reward
        state = next_state

    print(f"Episode {episode + 1}: Reward received: {reward}")

print(f"Over {num_episodes} episodes, the random agent received a total reward of: {total_reward}")

env.close()
```

H) Evaluar el desempeño del agente agente reflexivo (medida de desempeño y unidades de tiempo consumidas) para: **(OPCIONAL)**

4. Entornos de : 2x2, 4x4, 8x8, 16x16, 32x32, 64x64, 128x128
5. Porcentaje de agujeros en el ambiente: 0.1, 0,2 0,4, 0.8
6. Repetir 10 veces cada combinación.

Forma de entrega:

1. La implementación de los puntos D y E se debe realizar en python 3. Pueden usarse todas las bibliotecas y/o frameworks. (en este último caso se deberá demostrar que se conoce lo que realiza el framework)
2. Dentro del repositorio **ia-uncuyo-2023** crear una carpeta con el nombre **tp2-agentes-racionales/**.

3. Dentro de dicha carpeta (**tp2-agentes-racionales/**), colocar un archivo con el nombre **tp2-peas.md** con la respuesta a la pregunta A) y en un archivo con el nombre t**p2-aima-questions.md** con las respuestas a la pregunta F)
4. Dentro de dicha carpeta (**tp2-agentes-racionales/**) crear una nueva carpeta **code/** para el proyecto desarrollado en python
5. Dentro de dicha carpeta (**tp2-agentes-racionales/**) colocar un archivo con el nombre **tp2-results.md** con la respuesta a las preguntas D) y E)

Ejemplo:

la-uncuyo-2023/
tp2-agentes-racionales/
code/*.py
tp2-peas.md
tp2-aima-questions.md
tp2-results.md
Readme.md