p229278-M.SHAFEEN

May 28, 2024

[ ]:
```
percept=["smoke","normal",30]
state=["alarm","normal","Alarm"]
rules=["sprinkler","normal","call"]

def getState(percept_value,value):
    index = -1
    for x in percept:
        # print(x)
        index = index+1
        if x == percept_value:
                return state[index]
        if value==True:
            continue
        elif int(percept_value) > percept[2]:
                return state[2]
        # elif percept_value:
        #     return state[int[index]]
def getRules(state_value):
    index = -1
    for i in state:
        index = index+1
        if i == state_value:
            # if i.lower() == percept[0] and state_value == percept[0]:
            #     return rules[0]
            # elif i == percept[1] and int(state_value) > percept[1]:
            #     return rules[1]
            # elif i.lower() == percept[2] and state_value == percept[2]:
            #     return rules[2]
            return rules[index]

def simpleReflexAgent(percep,value):
    return getRules(getState(percep,value))
while (1):
    print("Choose from the following : ")
    # visual_input = input("What situation is it ? 1 ) Smoke , 2 ) High␣
 ↪Temperaure 3 )  Normal")
```

```
    choice=input("What situation is it ? 1 ) Smoke , 2 ) High Temperaure 3 ) Normal")
    if choice=="1":
        print(simpleReflexAgent(percept[0],True))
    elif choice=="2":
        choice1=input("Enter Temperature")
        print(simpleReflexAgent(choice1,False))
    elif choice == "3":
        print(simpleReflexAgent(percept[1],True))
    elif choice=="4":
        break
    # rule = simpleReflexAgent(visual_input)
    # print(rule)
    # print("Do you want to exit? Press 1")
    # choice=input("Enter For Exit , any other key if you want to continue")
    # if choice=="1":
    #     break
```

```
Choose from the following :
sprinkler
Choose from the following :
call
Choose from the following :
call
Choose from the following :
normal
Choose from the following :
```

0.1 PROBLEM 2

[ ]:
```
percept=["Dry soil","Moist soil","Wet soil"]
state=["watering ON","watering OFF","deactivates"]
rules=["water","avoid overwatering","prevent waterlogging"]

def getState(percept_value,value):
    index = -1
    for x in percept:
        # print(x)
        index = index+1
        if x == percept_value:
                return state[index]
        # if value==True:
        #     continue
        # elif int(percept_value) > percept[2]:
        #         return state[2]
        # elif percept_value:
        #     return state[int[index]]
```

```
def getRules(state_value):
    index = -1
    for i in state:
        index = index+1
        if i == state_value:
            # if i.lower() == percept[0] and state_value == percept[0]:
            #     return rules[0]
            # elif i == percept[1] and int(state_value) > percept[1]:
            #     return rules[1]
            # elif i.lower() == percept[2] and state_value == percept[2]:
            #     return rules[2]
            return rules[index]

def simpleReflexAgent(percep,value):
    return getRules(getState(percep,value))
while (1):
    print("Choose from the following : ")
    # visual_input = input("What situation is it ? 1 ) Smoke , 2 ) High␣
 ↪Temperaure 3 )  Normal")
    choice=input("What situation is it ? 1 ) dry soil , 2 ) moist soil 3 )  wet␣
 ↪soil")
    if choice=="1":
        print(simpleReflexAgent(percept[0],True))
    elif choice=="2":
        # choice1=input("Enter Temperature")
        print(simpleReflexAgent(percept[2],False))
    elif choice == "3":
        print(simpleReflexAgent(percept[1],True))
    elif choice=="4":
        break
    # rule = simpleReflexAgent(visual_input)
    # print(rule)
    # print("Do you want to exit? Press 1")
    # choice=input("Enter For Exit , any other key if you want to continue")
    # if choice=="1":
    #     break
```

```
Choose from the following :
water
Choose from the following :
prevent waterlogging
Choose from the following :
prevent waterlogging
Choose from the following :
avoid overwatering
Choose from the following :
avoid overwatering
```

Choose from the following :