Ismael njihia

20/03701

Question One

Internet games have become very popular. Designing a good computer game needs to use 3D graphics and artificial intelligence technologies. Search the literature to specify what AI techniques are able to make computer entertainment more exciting and challenging

1. **Adaptive Difficulty:** Adaptive difficulty refers to the ability of a game to dynamically adjust its difficulty level based on the player's skill level. AI can be used to analyze the player's behavior and performance in the game and adjust the difficulty level accordingly, making the game more challenging and engaging.
2. **Intelligent NPCs:** Non-Player Characters (NPCs) are characters in the game that are controlled by the computer. AI techniques such as rule-based systems, decision trees, and machine learning algorithms can be used to create intelligent NPCs that behave realistically and adapt to the player's actions, making the game more challenging and immersive.
3. **Procedural Content Generation:** Procedural content generation refers to the use of algorithms to generate game content, such as levels, maps, and items, dynamically. AI techniques such as genetic algorithms and neural networks can be used to generate content that is unique, unpredictable, and challenging, making the game more exciting.
4. **Natural Language Processing:** Natural Language Processing (NLP) is a branch of AI that deals with the interaction between computers and human language. NLP can be used in games to create more realistic and engaging dialogues between characters and to enable players to interact with the game world using natural language, making the game more immersive and challenging.
5. **Reinforcement Learning:** Reinforcement learning is a type of machine learning that involves an agent learning to interact with an environment by trial-and-error. Reinforcement learning can be used in games to create intelligent opponents that learn from their mistakes and improve their strategies over time, making the game more challenging and dynamic.
6. **Emotion Recognition:** Emotion recognition is an AI technique that involves analyzing facial expressions, voice tone, and other cues to identify a person's emotional state. In games, emotion recognition can be used to create more realistic and nuanced character interactions and to adjust the game's difficulty level based on the player's emotional state, making the game more engaging and challenging.
7. **Generative Adversarial Networks (GANs):** GANs are a type of neural network that can be used to generate realistic and novel content, such as images, music, and sound effects. In games, GANs can be used to create procedurally generated content, such as enemies and weapons, that are unique and challenging, making the game more exciting and dynamic.

Question Two

Find applications of artificial intelligence and expert systems. Identify an organization with which at least one member of your group has a good contact who has a decision-making problem that requires some expertise (but is not too complicated). Understand the nature of its business and identify problems that have been supported or can potentially be supported by intelligent systems.

One organization with which I have a good contact is a mid-sized marketing agency that focuses on digital marketing campaigns for small and medium-sized businesses. The company has expressed interest in using AI and expert systems to improve their campaign planning and optimization processes.

Some potential applications of AI and expert systems in this context could include:

1. Campaign Planning: AI can be used to analyze data from past campaigns, as well as demographic and behavioral data from target audiences, to help identify the most effective messaging and targeting strategies. This can help the agency improve the effectiveness of their campaigns and increase client satisfaction.
2. Ad Optimization: Expert systems can be used to analyze data from ad performance metrics, such as click-through rates and conversion rates, to identify patterns and trends that can be used to optimize ad targeting and messaging. This can help the agency improve the ROI of their campaigns and reduce costs.
3. Content Creation: AI can be used to generate or suggest content for social media and other marketing channels based on analysis of past performance and target audience demographics and preferences. This can help the agency create more effective and engaging content for their clients.
4. Market Research: Expert systems can be used to analyze data from social media and other online sources to identify trends and consumer preferences in target markets. This can help the agency develop more effective marketing strategies and increase client satisfaction.

Question Three

Consider the decision-making situation defined by the following rules: 1. If it is a nice day and it is summer, then I go to the golf course. 2. If it is a nice day and it is winter, then I go to the ski resort. 3. If it is not a nice day and it is summer, then I go to work. 4. If it is not a nice day and it is winter, then I go to class. 5. If I go to the golf course, then I play golf. 6. If I go to the ski resort, then I go skiing. 7. If I go skiing or I play golf, then I have fun. 8. If I go to work, then I make money. 9. If I go to class, then I learn something. a. Follow the rules for the following situations (what do you conclude for each one?):

1. **It is a nice day and it is summer.**

According to rule 1, if it is a nice day and it is summer, then I go to the golf course. Following rule 5, if I go to the golf course, then I play golf, and following rule 7, if I play golf, then I have fun. Therefore, I conclude that if it is a nice day and it is summer, then I go to the golf course and have fun playing golf.

2. **It is not a nice day and it is winter.**

According to rule 4, if it is not a nice day and it is winter, then I go to class. Following rule 9, if I go to class, then I learn something. Therefore, I conclude that if it is not a nice day and it is winter, then I go to class and learn something.

3. **It is a nice day and it is winter.**

According to rule 2, if it is a nice day and it is winter, then I go to the ski resort. Following rule 6, if I go to the ski resort, then I go skiing, and following rule 7, if I go skiing, then I have fun. Therefore, I conclude that if it is a nice day and it is winter, then I go to the ski resort and have fun skiing.

4. **It is not a nice day and it is summer.**

According to rule 3, if it is not a nice day and it is summer, then I go to work. Following rule 8, if I go to work, then I make money. Therefore, I conclude that if it is not a nice day and it is summer, then I go to work and make money.

b. Are there any other combinations that are valid? Explain.

No, there are no other combinations that are valid based on the given rules.

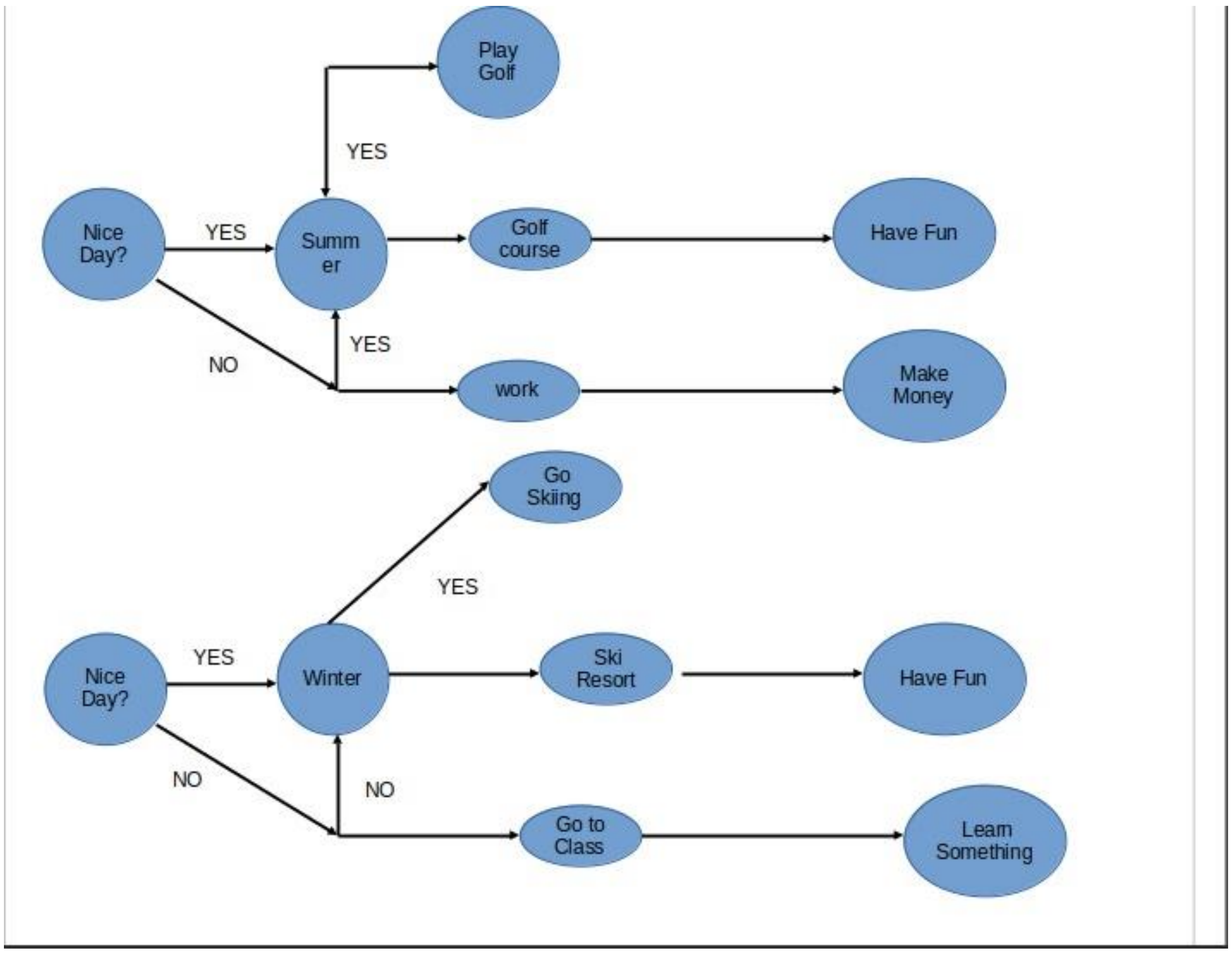
All possible combinations of a nice day/not a nice day and summer/winter have been covered by the rules, and the corresponding actions and outcomes have been defined. Therefore, any other combination that is not defined by the rules does not have a corresponding action or outcome, and thus is not valid.

c. What needs to happen for you to "learn something" in this knowledge universe? Start with the conclusion "learn something" and identify the rules used (backward) to get to the needed facts.

To get to learn something conclusion from the rules, I have used backward chaining as follows :

1. If I go to class, then I learn something (rule 9)
2. It is not a nice day and it is winter, so I go to class (rule 4)
3. Therefore, I learn something.

d. Encode the knowledge into a graphical diagram (like an influence diagram). Use a circle to represent a fact such as: The day is nice or The day is not nice and an arrow to indicate influence.



e. Write a L.C. (or other third-generation language) program to execute this knowledge. Use IF- THEN (ELSE) statements in your implementation. How many lines long is it? How hard would it be to modify the program to insert new facts and a rule such as:

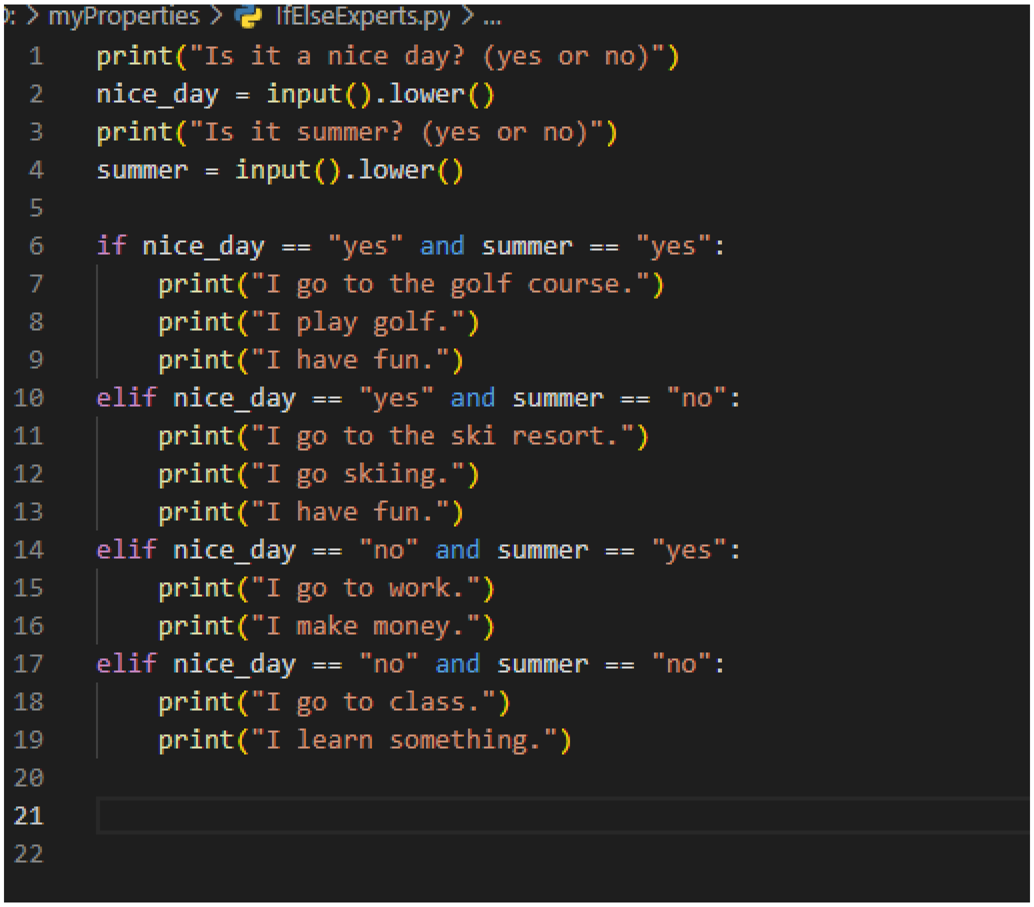
1. If it is cloudy and it is warm

2. and it is not raining

3. and it is summer

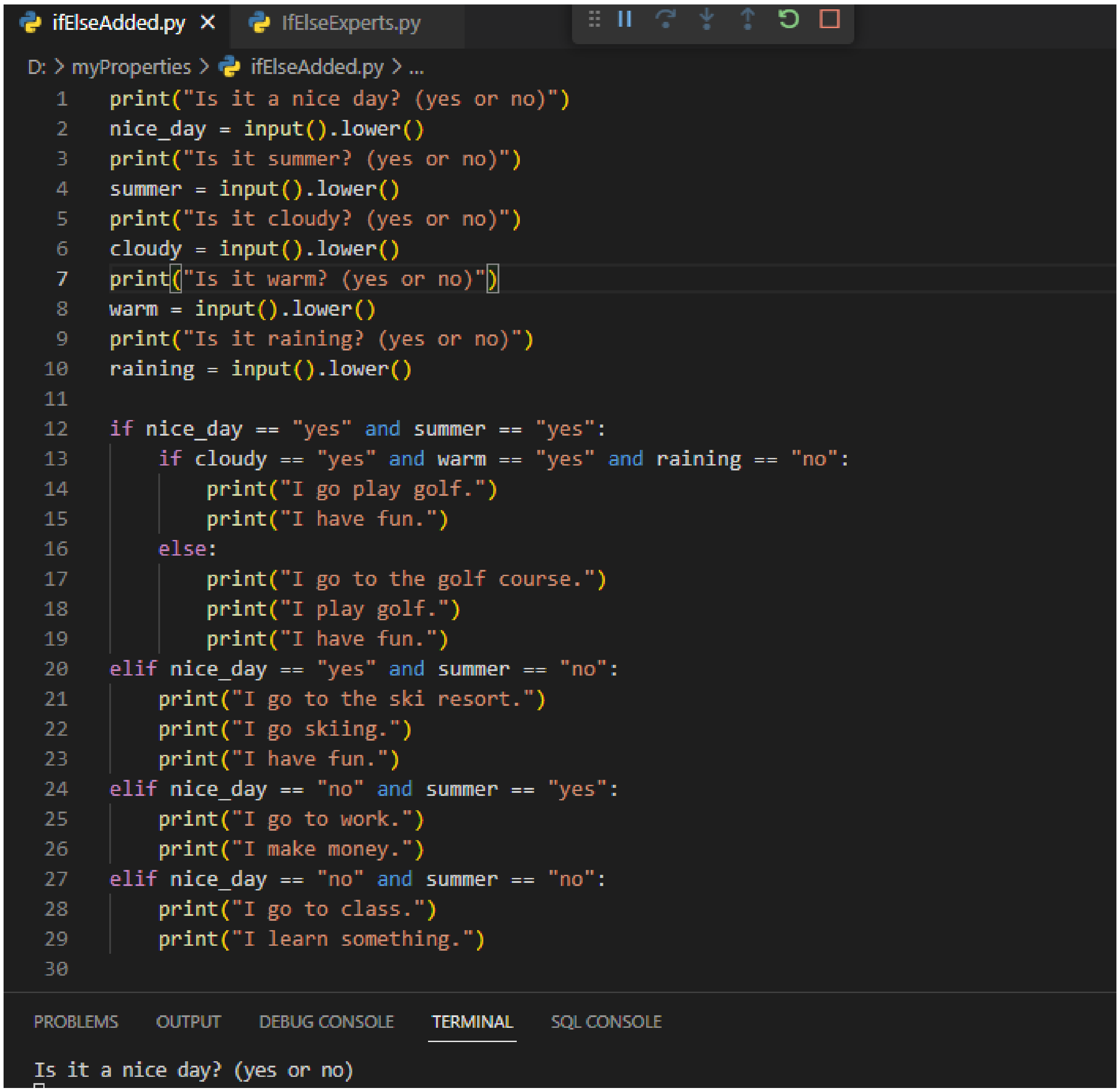
4. then I go play golf

Here is a Python program to execute the give knowledge.



```python
print("Is it a nice day? (yes or no)")
nice_day = input().lower()
print("Is it summer? (yes or no)")
summer = input().lower()

if nice_day == "yes" and summer == "yes":
    print("I go to the golf course.")
    print("I play golf.")
    print("I have fun.")
elif nice_day == "yes" and summer == "no":
    print("I go to the ski resort.")
    print("I go skiing.")
    print("I have fun.")
elif nice_day == "no" and summer == "yes":
    print("I go to work.")
    print("I make money.")
elif nice_day == "no" and summer == "no":
    print("I go to class.")
    print("I learn something.")
```

The program has 20 lines of code


ifElseAdded.py × IfElseExperts.py

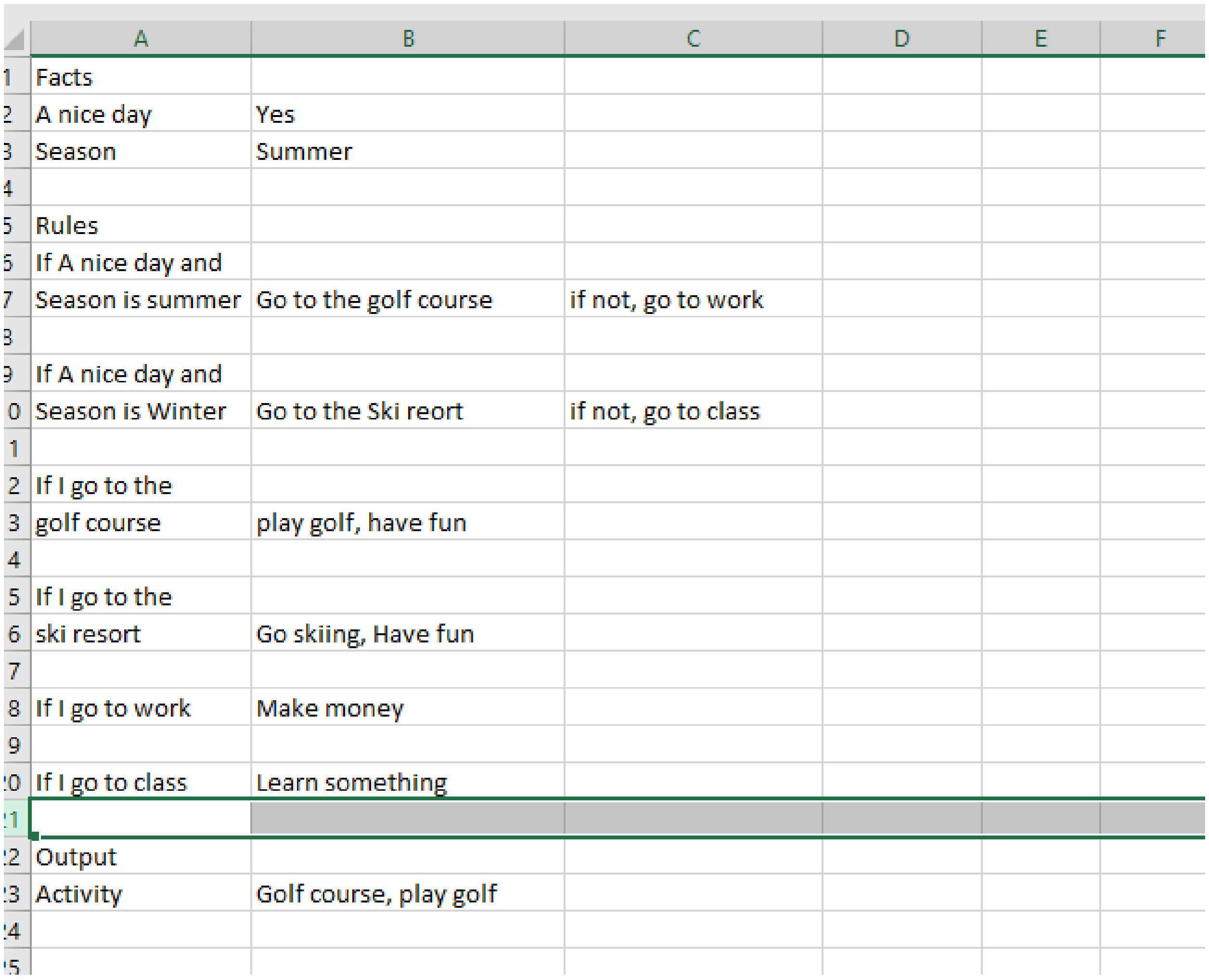
D: > myProperties > ifElseAdded.py > ...

```python
print("Is it a nice day? (yes or no)")
nice_day = input().lower()
print("Is it summer? (yes or no)")
summer = input().lower()
print("Is it cloudy? (yes or no)")
cloudy = input().lower()
print("Is it warm? (yes or no)")
warm = input().lower()
print("Is it raining? (yes or no)")
raining = input().lower()

if nice_day == "yes" and summer == "yes":
    if cloudy == "yes" and warm == "yes" and raining == "no":
        print("I go play golf.")
        print("I have fun.")
    else:
        print("I go to the golf course.")
        print("I play golf.")
        print("I have fun.")
elif nice_day == "yes" and summer == "no":
    print("I go to the ski resort.")
    print("I go skiing.")
    print("I have fun.")
elif nice_day == "no" and summer == "yes":
    print("I go to work.")
    print("I make money.")
elif nice_day == "no" and summer == "no":
    print("I go to class.")
    print("I learn something.")

```

```
Is it a nice day? (yes or no)
```

It is not hard to add new facts. The modified code is 30 lines of code

f. Implement the knowledge in a spreadsheet or database package on a PC.

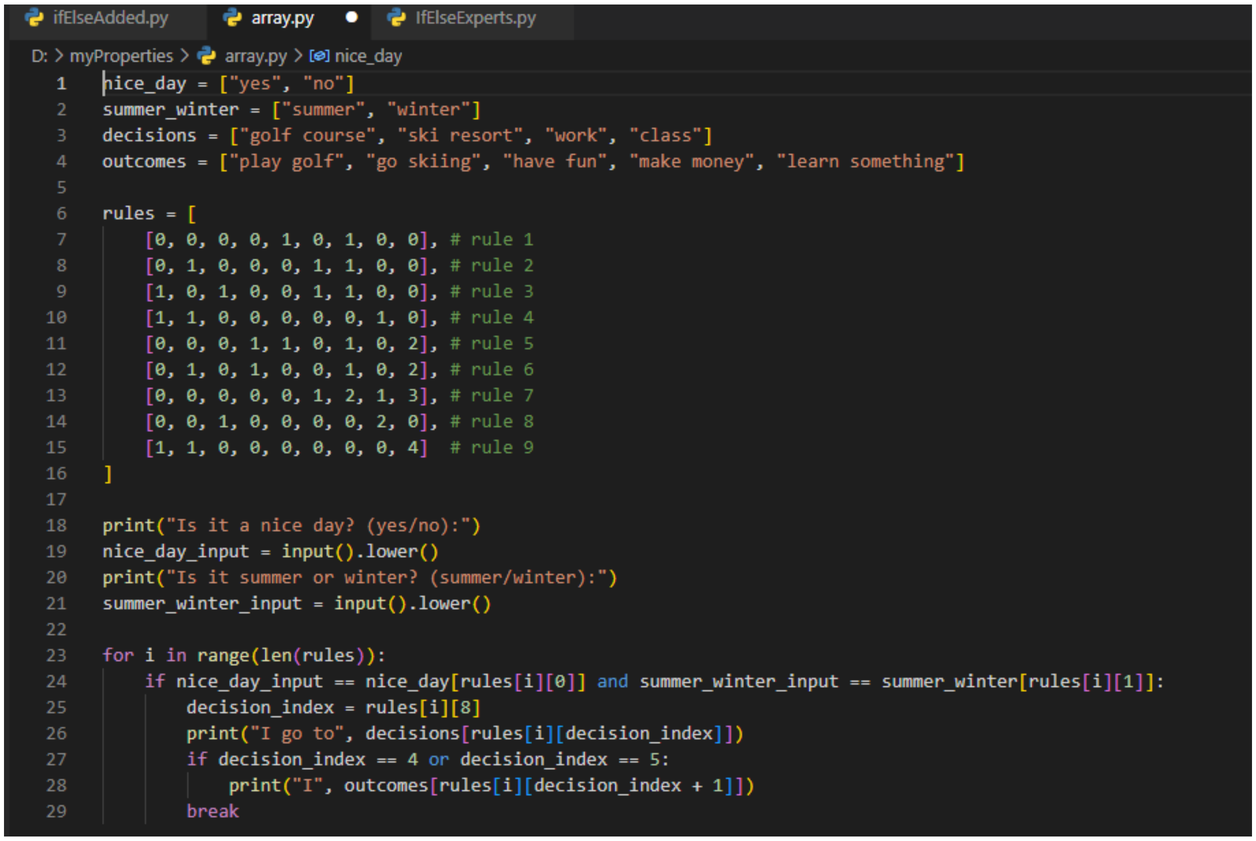


	A	B	C	D	E	F
1	Facts					
2	A nice day	Yes				
3	Season	Summer				
4						
5	Rules					
6	If A nice day and					
7	Season is summer	Go to the golf course	if not, go to work			
8						
9	If A nice day and					
10	Season is Winter	Go to the Ski reort	if not, go to class			
11						
12	If I go to the					
13	golf course	play golf, have fun				
14						
15	If I go to the					
16	ski resort	Go skiing, Have fun				
17						
18	If I go to work	Make money				
19						
20	If I go to class	Learn something				
21						
22	Output					
23	Activity	Golf course, play golf				
24						
25						

To modify the program to include the new rule, we need to add a new row to the Rules table:

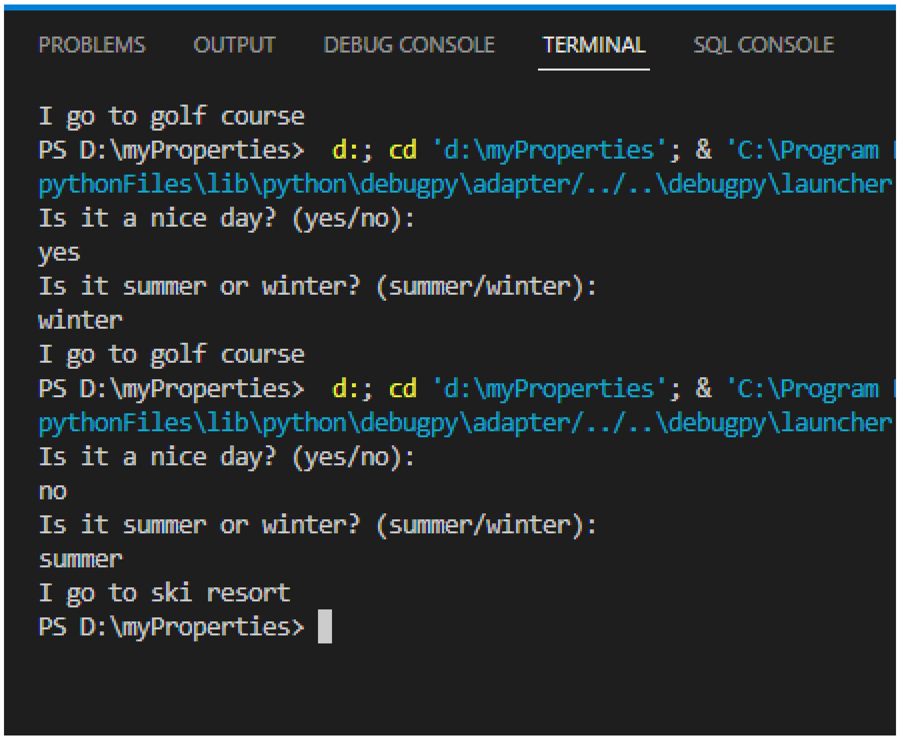
| If it is cloudy and it is warm | and it is not raining | and it is summer | Go play golf |

g. Advanced exercise. In an implementation similar to the one in part (d), write a new implementation but store the knowledge in variables. Let the program search the arrays to make decisions.



```python
nice_day = ["yes", "no"]
summer_winter = ["summer", "winter"]
decisions = ["golf course", "ski resort", "work", "class"]
outcomes = ["play golf", "go skiing", "have fun", "make money", "learn something"]

rules = [
    [0, 0, 0, 0, 1, 0, 1, 0, 0], # rule 1
    [0, 1, 0, 0, 0, 1, 1, 0, 0], # rule 2
    [1, 0, 1, 0, 0, 1, 1, 0, 0], # rule 3
    [1, 1, 0, 0, 0, 0, 0, 1, 0], # rule 4
    [0, 0, 0, 1, 1, 0, 1, 0, 2], # rule 5
    [0, 1, 0, 1, 0, 0, 1, 0, 2], # rule 6
    [0, 0, 0, 0, 0, 1, 2, 1, 3], # rule 7
    [0, 0, 1, 0, 0, 0, 0, 2, 0], # rule 8
    [1, 1, 0, 0, 0, 0, 0, 0, 4]  # rule 9
]

print("Is it a nice day? (yes/no):")
nice_day_input = input().lower()
print("Is it summer or winter? (summer/winter):")
summer_winter_input = input().lower()

for i in range(len(rules)):
    if nice_day_input == nice_day[rules[i][0]] and summer_winter_input == summer_winter[rules[i][1]]:
        decision_index = rules[i][8]
        print("I go to", decisions[rules[i][decision_index]])
        if decision_index == 4 or decision_index == 5:
            print("I", outcomes[rules[i][decision_index + 1]])
        break
```

Output



```
I go to golf course
PS D:\myProperties>  d:; cd 'd:\myProperties'; & 'C:\Program
pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher
Is it a nice day? (yes/no):
yes
Is it summer or winter? (summer/winter):
winter
I go to golf course
PS D:\myProperties>  d:; cd 'd:\myProperties'; & 'C:\Program
pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher
Is it a nice day? (yes/no):
no
Is it summer or winter? (summer/winter):
summer
I go to ski resort
PS D:\myProperties>
```