Richard Hayes Crowley

07/24/2021

CSC\_242\_Lab\_012

**SOURCE CODE:**

**Brainstorm.py**

*from* typing *import* TypedDict

*from* simple\_term\_menu *import* TerminalMenu

*from* tabulate *import* tabulate

*import* operations

*from* random *import* sample

class Team(TypedDict):

name: set

class CategoryCode(TypedDict):

code: str

class Operations(TypedDict):

operation: any

*# category codes for coding team member preferences*

categoryCodes: CategoryCode = {

"A": "analysis",

"Q": "prototyping",

"D": "design",

"R": "software",

"E": "maintenance",

"S": "support",

"M": "management",

"T": "technology",

"O": "organizing",

"U": "testing",

"P": "programming",

"V": "user insight"

}

*# the list of team members*

team: Team = {

"Daisy": set(["D", "M", "T", "U"]),

"Edward": set(["D", "E", "M", "R", "V"]),

"Kay": set(["E", "M", "S", "V", "U"]),

"Mirna": set(["A", "E", "O", "P", "S"]),

"Sidney": set(["Q", "R", "V"]),

"Tina": set(["A", "M", "O", "P", "S"]),

"Zachary": set(["P", "Q", "O", "T", "U"])}

*# set operations index*

operations: Operations = {

"Union": operations.union,

"Intersection": operations.intersection,

"Single Attribute Subset": operations.singleAttributeSubset,

"Set Complement": operations.symmetricDifference,

"Difference": operations.difference,

"Exit": ""

}

skillList = [[i[0], i[1]] *for* i *in* categoryCodes.items()]

teamList = [[i[0], ', '.join(j *for* j *in* i[1])] *for* i *in* team.items()]

def selectTeamMember():

teamMemberNameList = [i[0] *for* i *in* teamList]

entry = TerminalMenu(

teamMemberNameList, *title*="\nWhich team member would you like to select?").show()

memberName = teamMemberNameList[entry]

memberSkillset = team[memberName]

*return* (memberName, memberSkillset)

def selectSkill():

skillStrList = [" : ".join([i[0], i[1]])*for* i in skillList]

entry = TerminalMenu(

skillStrList, *title*="\nWhich skill would you like to select?").show()

skillCode = skillList[entry][0]

skillName = skillList[entry][1]

*return* (skillCode, skillName)

def main():

operationsList = [k *for* k *in* operations.keys()]

skillTable = tabulate(skillList, *headers*=[

"Code", "Skill"], *tablefmt*="github", *numalign*="left")

teamTable = tabulate(teamList, *headers*=[

"Name", "Skillset"], *tablefmt*="github", *numalign*="left")

print("\n~\*~\*~\* Welcome to Fake Business Brainstorm! ~\*~\*~\*~")

print("Your task is to put together various team configurations based on skillsets using set operations for a big project")

print("\nHere are the skill categories and corresponding skill code:\n")

print(skillTable)

print("\nAnd here is the team and their corresponding skill sets:\n")

print(teamTable)

*while* True:

entry = TerminalMenu(

operationsList, *title*="\nWhat set operation would you like to conduct?").show()

*if* operationsList[entry] == "Union":

print("\nSelect two team members to form union...")

memberA = selectTeamMember()

memberB = selectTeamMember()

union = operations["Union"](memberA[1], memberB[1])

print(

f"\nThe set of all the combined unique attributes in the profiles of members {memberA[0]} and {memberB[0]} is: \n")

print([categoryCodes[i] *for* i *in* union])

*elif* operationsList[entry] == "Intersection":

print("\nSelect two team members to form union...")

memberA = selectTeamMember()

memberB = selectTeamMember()

intersection = operations["Intersection"](memberA[1], memberB[1])

print(

f"\nThe set of common skills between team members {memberA[0]} and {memberB[0]} is:")

print([categoryCodes[i] *for* i *in* intersection])

*elif* operationsList[entry] == "Single Attribute Subset":

print("\nSelect a skill that you'd like to create a set for...")

skill = selectSkill()

singleAttributeSubset = operations["Single Attribute Subset"](

skill[0], team)

print(

f"\nThe team ( subset ) of all the members that are proficient in the '{skill[1]}' attribute is: ")

print([member *for* member *in* singleAttributeSubset])

*elif* operationsList[entry] == "Set Complement":

universalSet = set([i *for* i *in* categoryCodes.keys()])

print("\nForming subset of management skills...")

mgmtSkillsSubset = set(["M", "O", "A", "V"])

complement = operations["Set Complement"](

universalSet, mgmtSkillsSubset)

complementList = [categoryCodes[i] *for* i *in* complement]

print(

f"\nThe Set Complement (symmetric difference) of mgmt skills and the rest of the skills is:")

print(complementList)

*pass*

*elif* operationsList[entry] == "Difference":

print("\n First, form two random subsets of team members")

lenOptions = [str(i + 1) *for* i *in* range(len(team))]

select\_len1 = TerminalMenu(

lenOptions, *title*="\nHow large will the first team subset be?").show()

subset1 = set(sample(list(team.keys()),

int(lenOptions[select\_len1])))

select\_len2 = TerminalMenu(

lenOptions, *title*="\nHow large will the second team subset be?").show()

subset2 = set(sample(list(team.keys()),

int(lenOptions[select\_len2])))

print("\nFirst random team subset is: ")

print(subset1)

print("\nsecond random team subset is: ")

print(subset2)

difference = operations["Difference"](

subset1, subset2)

print("\nThe difference between the first and second subset is: ")

print(difference)

*else*:

print("Goodbye!")

exit()

*if* \_\_name\_\_ == "\_\_main\_\_":

main()

**operations.py**

def union(*setA*: set, *setB*: set):

*return* *setA*.union(*setB*)

def intersection(*setA*: set, *setB*: set):

*return* *setA*.intersection(*setB*)

def difference(*setA*: set, *setB*: set):

*return* *setA*.difference(*setB*)

def symmetricDifference(*setA*: set, *setB*: set):

*return* *setA*.symmetric\_difference(*setB*)

def singleAttributeSubset(*attribute*: str, *team*: dict):

attributeSet = set()

*for* teamMember *in* *team*.items():

*if* *attribute* in teamMember[1]:

attributeSet.add(teamMember[0])

*return* attributeSet

**OUTPUT:  
  
Please see attached video**