

שאלה 2 – OOP (34 נקודות)

הממשלה במדינת לה-לנד מורכבות משרדים ומשרדים-בפועל (ممלאי מקום שר). שרים רשאים להוסיף ולבטל חוקים לחוקת המדינה כאוות נפשם. לעומת זאת, שרים-בפועל סמכות פחותה ועל כן הם יכולים להוסיף ולבטל חוקים רק אם מתקיימים תנאים מסוימים שיפורטו בהמשך. בשאלת זו עליהם את למש המשלחת **שר-בפועל** (Minister) ואת המשלחת **שר** (ActingMinister) תוקן שימוש במילוי המשלחת עוזר בשם **Constitution** (חוקה) שננתונה לכם.

שימוש לב:

- המתוודות בשתי המשלחות אוטן עליהם למש אין מחזירות דבר.
- ניקוד מלא ניתן לפתרונות שיכללו תכנון נכון של המשלחות הנדרשות ויישמו נכון בתוכנות מונחה עצמים לצורך צמצום שכפול קוד.

הרשומות מימוש מוכן של המשלחת **Constitution** (חוקה) המייצגת את אוסף חוקי המדינה. שימושם לב שהמימוש המלא של המשלחת אינו חזוף כדי להמעיט בפרטים ולהתמקד בעיקר, אבל אתם יכולים להניח שהוא תקין ועובד.

```
class Constitution:
    def __init__(self):
        # initiates a constitution object
        ...

    def __repr__(self):
        # returns a string representation of the constitution
        object
        ...

    def add_rule(self, rule):
        # adds a rule (string) to the constitution object
        ...

    def remove_rule(self, rule):
        # removes a rule from the constitution object
        ...
```

סעיף א' – 12 נקודות

משו את המשלחת **Minister** אשר תיציג שר. שר יכול להוסיף ולבטל חוקים כאוות נפשו. המשלחת תכלול את המתוודות הבאות:

1. (4 נק') בניין המשלחת (מתודת `__init__`) קיבל את הפרמטרים הבאים וישמר כל אחד מהם לשדה:
 - 1.1. `name` – מחוץ המציג את שם השר.
 - 1.2. `party` – מחוץ המציג את שם המפלגה אליה משתייך השר.
2. (4 נק') המתודה `add_rule(self, rule, constitution)` מקבל מחוץ (`rule`) שמצוינת שם של חוק, ואובייקט מסווג **Constitution** המייצג חוקה. המתודה תוסיף את החוק לחוקה באמצעות המתודה המתאימה של המשלחת **Constitution** ותДЕיס את המחוות הباءה:

"A new rule by \$minister_name was accepted"

כasher \$minister_name הינו שם השר היוזם את החוק. ניתן להניח שהחוק לא נמצא בחוקה.
3. (4 נק') המתוודה \$minister_name מקבל מחרוזת rule(החוק) שמצוינת שם של חוק ואובייקט מסווג Constitution המייצג חוקה. המתוודה תבטל את החוק באובייקט החוקה constitution באמצעות המתואימה של המחלקה Constitution ותדפיס את המחרוזות הבאה:

"Rule was canceled by \$minister_name"

כasher \$minister_name הינו שם השר אשר מבטל את החוק. ניתן להניח שהחוק קיים בחוקה.

דוגמאות ריצה:

```
>>> c = Constitution()
>>> M1 = Minister('Bilbi', 'Kibud')
>>> M2 = Minister('GantzAndRoses', 'Hosen to lala-land')
>>> M1.add_rule('rule1: defense', c)
A new rule by Bilbi was accepted
>>> M1.add_rule('rule2: dEfeNse', c)
A new rule by Bilbi was accepted
>>> M2.add_rule('rule4: Economy', c)
A new rule by GantzAndRoses was accepted
>>> M2.cancel_rule('rule2: dEfeNse', c)
Rule was canceled by GantzAndRoses
>>> print c
Constitution of lala-land contains the following rules:
rule1: defense
rule4: Economy
```

```
class Minister:

    def __init__(self, name, party):
        self.name = name
        self.party = party

    def add_rule(self, rule, constitution):
        constitution.add_rule(rule)
        print('A new rule by '+self.name+' was accepted')

    def cancel_rule(self, rule, constitution):
        constitution.remove_rule(rule)
        print('Rule was canceled by '+self.name)
```

סעיף ב' – 13 נקודות

ממשו את המחלקה **ActingMinister** שתציג שר-בפועל (ممלא מקום שר). שר-בפועל הוא סוג של שר, אבל ביכולתו להוציא ולבטל חוקים רק אם מתקיימים תנאים מסוימים כמפורט להלן: המחלקה תכלול את המתודות הבאות:

1. (3 נק') בנאי המחלקה (מתודת `__init__`) יקבל את אותם פרמטרים שמקבל בנאוי המחלקה **Minister** ויתחל את אותן השדות. בנוסף, יקבל הבנאי (וישמור בשדה מתאים) פרמטר בשם `ministry_name` – מחרוזת המייצגת את שם המשרד אליו משוויך השר-בפועל. יש להמיר את מחרוזות הקלט לאותיות גדולות (upper case) בטרם שמירתה לשדה.
2. (5 נק') המתודה `add_rule(self, rule, constitution, votes)` שמצוינת שם של חוק, אובייקט מסווג **Constitution** המייצג חוקה ורשימה של הצבעות (`votes`, כאשר 0 ייצג "נגד" ויאילו 1 ייצג "בעד") של חברי ועדת מחוקקת. אם הושג רוב, כלומר, יותר מחצי מההצבעות היו בעד, המתודה תבצע בדיקה מה שמצעתה המתודה `add_rule` של המחלקה **Minister**. אחרת, תזפיס למסך את המחרוזות הבאה: "Rule was not accepted". תישאר כפי שהיא). ניתן להניח ש `votes` אינה ריקה ושאייריה הינם 0 או 1 (מטיפוס `int`) בלבד.
3. (5 נק') המתודה `cancel_rule(self, rule, constitution)` שמצוינת שם של חוק ואובייקט מסווג **Constitution** המייצג חוקה. אם החוק רלוונטי למשרד אליו משוויך **Minister**. השר-בפועל, המתודה תעשה בדיקה מה שעשויה **ValueError** (עם הודעת שגיאה לבחירתכם). חוק רלוונטי למשרד אחרית, תעלת חריגה מסווג **ValueError** (עם הודעת שגיאה לבחירתכם). חוק רלוונטי למשרד מסוים רק אם שם המשרד מופיע במיקום כלשהו בתחום שם החוק לא חשיבות לגודל האותיות **Defense** – **Defense insensitive**. למשל בחוק של משרד `Defense` חייבת להופיע המילה `Defense` (כאשר האותיות בגדים כלשנות). ניתן להניח שהחוק קיים בחוקה.

דוגמאות הרצה: (המשךים עם אותה חוקה מהסעיף הקודם)

```
>>> print c # same constitution (and ministers) as before
Constitution of lala-land contains the following rules:
rule1: defense
rule4: Economy
>>> AM1 = ActingMinister('Lafeed', 'Ein future', 'defense')
>>> M1.add_rule('rule5: DEfeNse', c) #no voting (M1 is a minister)
A new rule by Bilbi was accepted
>>> AM1.cancel_rule('rule4: Economy', c)
Traceback (most recent call last):
...
ValueError
>>> AM1.cancel_rule('rule5: DEfeNse', c)
Rule was canceled by Lafeed
```

```
>>> AM1.add_rule('rule6: DEFENSE', c, [0,0,1,1]) # tie
Rule was not accepted
>>> AM1.add_rule('rule7: DEFENSE', c, [0,0,1,1,1]) # majority
A new rule by Lafeed was accepted
>>> print c
Constitution of lala-land contains the following rules:
rule1: defense
rule4: Economy
rule7: DEFENSE
```

```
class ActingMinister(Minister):

    def __init__(self, name, party, ministry_name):
        Minister.__init__(self, name, party)
        self.ministry_name = ministry_name.upper()

    def add_rule(self, content, constitution, votes):
        if sum(votes)*2 > len(votes):
            Minister.add_rule(self, content, constitution)
        else:
            print('Rule was not accepted')

    def cancel_rule(self, rule, constitution):
        if self.ministry_name not in rule.upper():
            raise ValueError
        Minister.cancel_rule(self, rule, constitution)
```

סעיף ג' – 9 נקודות

כדי לעודד את משרדי הממשלה השונים לחוקים, החליט נשיא ללה-לנד לעורוך תחרות חודשית בה יוענק אות כבוד למשרד שהייה הכי פעיל החודש. בסעיף זה (שאינו קשור לסעיפים הקודמים) עליכם לממש פונקציה (לא מותודה) בשם most_productive_ministry(rule_to_ministry_dict) אשר תעוזר לנשיא לגלוות מייהו המשרד שחוקק הכי הרבה חוקים החדש. הפונקציה תקבל מילון (לא ריק) שהמפתחות שלו הם שמות של חוקים (מחזורות) והערך של כל מפתח יהיה שם המשרד (מחזורות) שחוקק את החוק. הפונקציה תחזיר את שם המשרד שחוקק הכי הרבה חוקים (ניתן להניח שיש בדיק אחד כזה).

דוגמאות הרצה:

```
>>> rule_to_ministry_dict = {'rule1': ...': 'Ministry of Defense',
                               'rule2': ...': 'Ministry of Science',
                               'rule3': ...': 'Ministry of Culture',
                               'rule4': ...': 'Ministry of Defense',
                               'rule5': ...': 'Ministry of Science',
                               'rule6': ...': 'Ministry of Science'}
```

```
>>> print 'Most productive ministry is:',
most_productive_ministry(rule_to_ministry_dict)
```

```
Most productive ministry is: Ministry of Science
```

```
def most_productive_ministry(rule_to_ministry_dict):
    ministry_to_count_dict = {}
    for rule in rule_to_ministry_dict :
        ministry = rule_to_ministry_dict [rule]
        ministry_to_count_dict[ministry ] =
ministry_to_count_dict.get(party, 0) + 1
    return max(ministry_to_count_dict,
key=ministry_to_count_dict.get)
```