بسمه تعالی

محمدرضا عسگری شکوه ۹۹۲۴۳۱۰۸
امیر ارشیا ادیبان ۹۹۲۴۳۱۰۵

AI Final Project
Phase 1 & 2 Report

ابتدا توابعی که پیاده سازی شده اند را بررسی میکنیم.

این تابع بر اساس رشته ورودی شکل جدول خالی را میسازد و برمیگرداند.

```
def create_table(row , column, ss):
  table_shape = []
  for i in range(row):
      table_shape.append([])
      for j in range(column):
          if ss[j+i*column]=="0":
              table_shape[i].append(" ")
          else:
              table_shape[i].append("#")

  table = beautifultable.BeautifulTable()
  for i in range(row):
      table.append_row(table_shape[i])
  return table
```

این توابع تعداد خانه های توپر را در ردیف ها و ستون ها را محاسبه میکنند. بعدا برای پر کردن جدول نیاز داریم که بدانیم کدام ردیف ها یا ستون ها دارای خانه پر هستند.

```
def ex_sharp_row(row,column,table):
  sharp= []
  for i in range(row):
    counter = 0
    for j in range(column):
      if table.rows[i][j]=='#':
        counter += 1
    sharp.append(counter)
  return sharp


def ex_sharp_column(row,column,table):
  sharp = []
  for i in range(column):
    counter = 0
    for j in range(row):
      if table.rows[j][i]=='#':
        counter += 1
    sharp.append(counter)
  return sharp
```

```
def filling_row(rows_answers,table,sharp, row,column):
  for i in range(row):
    ss = rows_answers[i][::-1]
    counter = 0
    if ss=='-':
      continue
    else:
      for j in range(column):
        if table.rows[i][j]=='#':
          try:
            tmp = ss[counter]
          except:
            continue
          if ss[counter]!='#':
            continue
          else:
            counter += 1
        elif table.rows[i][j]==' ' and len(ss) + sharp[i] <column:
          sharp[i] += 1
          continue
        else:
          table.rows[i][j] = ss[counter]
          counter += 1
```

تابع پاسخ های ردیف هارا از ورودی میگیرد و جدول را پر میکند.

چون کلمات فارسی هستند ابتدا انها را برعکس میکنیم چرا که جدول از خانه [0][0] که سمت چپ است پر میشود و باید این خانه اخرین حرف کلمه پاسخ باشد.

همچنین اگر توضیح سوال خالی باشد یعنی آن ردیف پاسخ ندارد و آن را رد میکنیم.

در ادامه اگر خانه ای که میخواهیم آن را پر کنیم با # پر شده باشد، و حرف رشته پاسخ در آن لحظه # نباشد، باید این خانه را اسکیپ کنیم.

در حالت بعدی اگر به خانه خالی رسیدیم و رشته پاسخ طولش کمتر از طول ستون های جدول به علاوه خانه های پر باشد، نیز باید اسکیپ کنیم. چرا که فارسی از راست به چپ است و اگر طول کلمه کمتر از طول جدول باشد، باید سمت چپ خالی بماند زیرا از راست کلمات نوشته میشوند.

در غیر این صورت به طور عادی پاسخ ها در جدول نوشته خواهند شد.

```python
def filling_column(columns_answers,table,sharp, row, column):
  for i in range(column):
    ss = columns_answers[i]
    counter = 0
    if ss=='-':
      continue
    else:
      for j in range(row):
        if table.rows[j][i]==' ':
          if table.rows[j][i]=='#':
            try:
              tmp = ss[counter]
            except:
              continue
            if ss[counter]!='#':
              continue
            else:
              counter += 1
          elif table.rows[j][i]==' ' and len(ss) + sharp[i] <row:
            sharp[i] += 1
            continue
          else:
            table.rows[j][i] = ss[counter]
            counter += 1
        else:
          continue
```

مانند تابع قبلی فقط با این تفاوت که ستون هارا پر میکند. اما فقط خانه هایی از جدول را که خالی باشند را پر میکند. تقریبا اکثرا مواقع با پر کردن ردیف ها، جدول کامل میشود.

```python
def search_word(word):
  word_answer = []
  return word_answer
```

تابع پیدا کردن کلمه پاسخ، که در فاز های بعدی قرار است کامل شود.

تابع main:

در ابتدا سایز جدول و رشته شکل جدول را از ورودی میگیریم و تابع ساختن جدول را صدا میزنیم. همچنین توابع محاسبه تعداد خانه های پر در هر ردیف و ستون.

در این قسمت که در این فاز استفاده نمیشود، توضیحات کلمات را از ورودی میگیریم و به ستون ها و ردیف ها تفکیک میکنیم. سپس برای هر ردیف و ستون، تابع یافتن کلمه را صدا میزنیم.

```
print('Enter table scale: ')
scale = input()
row = int(scale.split()[0])
column = int(scale.split()[1])
print('Enter table shape: ')
input_string = input()
table = create_table(row , column, input_string)

sharp_numbers_row = ex_sharp_row(row,column,table)
sharp_numbers_column = ex_sharp_column(row,column,table)

"""
print('Enter descriptions: ')
input_string = input()[1:-1].split('@')

rows_des = input_string[0:row]
columns_des = input_string[row:]
print(rows_des)
print(columns_des)

rows_answers = []
for i in range(row):
  rows_answers.append([])
  if rows_des[i]=='-':
    rows_answers[i] = '-'
  else:
    rows_answers[i] = search_word(rows_des[i])

columns_answers = []
for i in range(column):
  columns_answers.append([])
  if columns_des[i]=='-':
    columns_answers[i] = '-'
  else:
    columns_answers[i] = search_word(columns_des[i])
"""
```

```
#phase 1 and 2: we think that we have the answers, so print the table
rows_answers = ['بشیر#بالن','اسیا#املا','برابری#جم','کپر#اسیمه','اشکار','شریان#انا','وت#باردار','ابرو#سلیم','کالا#ادبا']
columns_answers = ['امرا#همان','بیان#مجلل','دلداری#ما','اسر#اسیاب','انکار','اوباش#بار','لر#یارایی','ابتر#پرسش','کاوش#کباب']

filling_row(rows_answers, table, sharp_numbers_row, row,column)
filling_column(columns_answers, table, sharp_numbers_column, row,column)

print(table)

ss = '&'
for word in rows_answers:
  ss = ss + word + '@'

for word in columns_answers:
  ss = ss + word + '@'

ss = ss + '&'

print(ss)
```

در این فاز فرض میشود پاسخ ها را داریم و توابع پر کردن ردیف ها و ستون ها صدا میزنیم و رشته خروجی که پاسخ ها است را در خروجی نمایش میدهد.

نمونه ورودی ها و خروجی ها:

```
Enter table scale:
9 9
Enter table shape:
000010000000010000000000100000100000110000011000001000001000000000010000000010000
C:\Users\Roham\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.9_qbz5n2kfra8p0\LocalCache\local-packages\Python39\site-packages\beautifultable\utils.py:125: FutureWarning: 'Beau
fulTable.append_row' has been deprecated in 'v1.0.0' and will be removed in 'v1.2.0'. Use 'BTRowCollection.append' instead.
  warnings.warn(message, FutureWarning)
+---+---+---+---+---+---+---+---+---+
| ا | ب | د | ا | # | ا | ل | ا | ک |
+---+---+---+---+---+---+---+---+---+
| م | ی | ل | س | # | و | ر | ب | ا |
+---+---+---+---+---+---+---+---+---+
| ر | ا | د | ر | ا | ب | # | ت | و |
+---+---+---+---+---+---+---+---+---+
| ا | ا | ن | ا | # | ن | ا | ی | ش |
+---+---+---+---+---+---+---+---+---+
| # | # | ر | ا | ک | ش | ا | # | # |
+---+---+---+---+---+---+---+---+---+
| ه | م | ی | س | ا | # | ر | پ | ک |
+---+---+---+---+---+---+---+---+---+
| م | ج | # | ی | ر | ا | ب | ر | ب |
+---+---+---+---+---+---+---+---+---+
| ا | ل | م | ا | # | ا | ی | س | ا |
+---+---+---+---+---+---+---+---+---+
| ن | ل | ا | ب | ا | ر | # | ش | ب |
+---+---+---+---+---+---+---+---+---+
&ن@ام ه#ارم ا@لجم#ن ایی@ام#ی رادلد@پ ایسا#رسا@ر اکن ا@ر اب#شاب ا@یی ا ر ای#رل@شهرپ#رتب ا@پ ابک#ش و اک@ل اب#ری ش@پ الم ا#ایسا@هج#ی رب ا رب@هم ی سا#ریک@ر اکش ا@ان ا#ن ای رش@ر ا د ر اب#ث و@م ی ل س#و ر پ ا@ابد ا#ال اک&
```