מטלת בית
עמית ישי
קבלה 7/3/24 19:50 הגשה 9/3/24 21:00

כנות:

- את הקובץ CSV חילצתי ומשקלו היה 240KB עם 1,995,793 שורות ולא 2.4 מיליון שורות.
- את הקובץ פתחתי ב-pycharm ונעזרתי בספריית csv כדי לבצע את השינויים הדרושים בקובץ. את הקובץ פתחתי rowzero שמאפשר לי לפתוח טבלאות גדולות.
- אני מאמין שניתן לעבוד בצורה יותר יעילה ולהשיג את המטרה שהגעתי אליה, אך עקב זמן קצוב בחרתי ללכת על הדרך שיותר מוכרת לי ותוך כדי למדתי חלק מהדברים המוצגים במטלה.
- נהנתי מאוד לסדר, לנקות ולשאוף לקובץ מושלם בנראות.
- בסוף יש צילומי מסך של כל הפונקציות + השמירה לjson.

חלק 1:

A. ברזיל
B. מספר זיהוי, שם מלא, קידומת מסט, האזור של קידומת המסט, מסט, מדינה, עיר, שכונה, רחוב, פרטי הבית1, פרטי הבית2.
C.

- מספר זיהוי: CPF Number בנוי מ-11 תווים בפורמט הבא: 000.000.000-00.
- שם מלא: אות גדולה בתחילת כל מילה.
- קידומת מסט + האזור: 21 באזור ריו דה ז'נרו, 22 צפון מזרח ברזיל ו- 24 במערב.
- מסט: מספר נייח מורכב מ-8 ספרות ונייד מ-9 ספרות לא כולל קידומת של 2 ספרות.
- מדינה: ברזיל מחולקת ל-26 מדינות.
- עיר: א.מ.ל.
- שכונה: א.מ.ל.
- פרטי בית: נמצאו קיצורים למילים בפורטוגזית והם שונו למילה המקורית.

הערות:

- נעזרתי בויקיפדיה כדי למצוא פרטים על מספר הזיהוי, הקידומת והמסט.
- נעזרתי בגוגל - לחלק מהערכים בטבלאות היו סימני פיסוק בתוך מילים, הסיבה לכך היא שהמחשב שלי לא תומך בפורטוגזית. למשל ă זאת האות ă ולכן היה צריך לייצר פונקציה שמחזיקה מילון עם הערכים והתרגום. הפונקציה `replacePatterns`
- נעזרתי ביוטיוב, stack overflow, chat GPT לחלק מהפונקציות וה-regex.
- נעזרתי בשלושת העמודות האחרונות כדי למצוא תבניות כמו: הפונקציה `replaceShortcut`

lote חלקת אדמה lt/t
quadra בלוק qu/qd/q
travessa רחוב/סמטה trv/trav
Apartamento דירה ap/apt
avenida שדרה av

- יש קצת אי התאמה בין העמודה של state לעמודה אחת אחריה שבחרתי לקרוא לה city כיוון שלא כל הערים בעמודה הזאת שייכות למדינת RJ.

הפונקציות:

- `removeZeros` מקבלת string ומפרקת אותה למערך של מילים בעזרת split. בעזרת join אני מחזיר את כל המילים לstring ללא אפסים מיותרים ובעזרת lstrip מקצץ את האפסים הנותרים.
- `replaceShortcut` מקבלת string ובעזרת מילון מחליפה לי את הקיצורים במילה המקורית. למשל casa בית , הקיצורים שהופיעו בטבלה היו CA or CS. בנוסף קיימת ביטוי רגולרי sub שמפריד לי בין אותיות ומספרים בעזרת רווח בשביל הנראות. לבסוף ה string מסודר באמצעות ביטוי רגולי שמגביל לרווח אחד בין מילים.
- `replaceMarkInStreet` מקבלת string ומקצצת נקודה שנמצאת בתחילת וסוף מילה. בנוסף היא מקצצת סימני פיסוק שנמצאים ב string ולא תורמים לה. במידה והיא מזהה סימן שאלה באמצע מילה היא מחליפה אותו באות(ã) a. ומקצצת סימני שאלה שמתחילים ומסיימים את ה string.
- `toUpperCase` מקבלת string ועוברת דרך ביטוי רגולי שהופך את כל האותיות לאותיות גדולות.
- `replaceToRJ` מקבלת string ומחליפה אותו ב string.
- checkPhoneNum מקבלת מספר וקידומת אזור חיוג. הפונקציה מנקה אותיות ודואגת שיש לו את כמות התווים הנכונה ומכינה את המספר להיות בפורמט בהתאם לתקן בברזיל:

  +55 (xx) xxxx-xxxx
- `insertStatePrefix` מקבלת שורה מטבלה ועוזרת לי להציג עמודה חדשה של מיקום אזור החיוג. למשל אזור חיוג 24 משומש במערב ברזיל.
- `checkId` מקבלת מספר זיהוי בודקת שהוא לא מורכב רק מאפסים ומקצצת אותיות לבסוף דואגת שהוא יהיה 11 תווים ובפורמט: xxx.xxx.xxx-xx
- `setCapitaleFirstLetter` מקבלת string בעזרת split מפצלת אותו למערך מילים ורק לאות הראשונה בכל מילה הופכת אותה לאות גדולה ומחברת הכל חזרה ל string עם join.
- `replacePatterns` מקבלת string ובודקת אם יש בה מילים מוצפנות מהשפה הפורטוגזית ומתרגמת אותם לאות מקבילה בשפה האנגלית.
- `replaceMarkInFullName` מקבלת string ומקצצת סימני פיסוק מיותרים, מחליפה ? באות a רק אם זה במילה ומקצצת נקודות מיותרות בתחילת וסוף מילה. מוחקת מספרים שלא קשורים לשם ומוחקת מה שיש בתוך סוגריים ().

מטלת בית
עמית ישי
קבלה 7/3/24 19:50 הגשה 9/3/24 21:00

```python
def removeZeros(str):
    words = str.split()
    filteredWord = [word for word in words if '0' in word and all(char == '0' for char in word)]
    outputWords = [word for word in words if word not in filteredWord]
    str = ' '.join(outputWords)
    return str.lstrip('0')
def replaceShortcut(str):
    replacePatterns = {
        r'\b(qu|qd|Q|Qu|QD|QU)\b': 'Quadra ',
        r'\b(trav|trv|TV)\b': 'Travessa ',
        r'\b(ap|apt|AP|Ap)\b': 'Apartamento ',
        r'\b(lt|lot|LT|Lt|LOTE|LO)\b': 'Loteamento ',
        r'\b(ave|av|AV|Av)\b': 'Avenida ',
        r'\b(BL|Bl|bl|block|BLOCK)\b': 'Bloco ',
        r'\b(CA|CS)\b': 'Casa ',
        r'(n\?|N\?|n\?:|N\?:)': 'Number '

    }
    str = re.sub( pattern: r'(\d)([a-zA-Z])|([a-zA-Z])(\d)', repl: r'\1\3 \2\4', str)
    for pattern, replacement in replacePatterns.items():
        str = re.sub(pattern,replacement,str)
    return re.sub( pattern: r'\s+', repl: ' ', str)
```

```python
def replaceMarkInStreet(name):
    if name.startswith('.'):
        name = name[1:]
    if name.endswith('.'):
        name = name[:-1]
    characters = ['!','%','*',',','-','/',':',';','#','.']
    ...
    for char in characters:
        name = name.replace(char, '')
    if '?' in name:
        name = name.replace('?','a')
    if name.startswith('? ') or name.endswith(' ?'):
        name = name.replace('?', '')
    ...

    return name

def toUpperCase(str):
    return re.sub( pattern: r'[a-z]', lambda match: match.group().upper(), str)


def replaceToRJ(str):
    str = 'RJ'
    return str
```

מטלת בית
עמית ישי
קבלה 7/3/24 19:50 הגשה 9/3/24 21:00

```python
def checkPhoneNum(num, prefix):
    if num == '0' or num.lstrip('0') == '':
        return ''
    removeLetters = re.sub( pattern: r'[a-zA-Z-]', repl: '',num)
    if len(removeLetters) > 9:
        return ''
    elif len(removeLetters) == 9:
        newNum = '+55' + prefix + removeLetters
        numFormat = f"{newNum[:3]} ({newNum[3:5]}) {newNum[5:10]}-{newNum[10:]}"
    elif len(removeLetters) == 8:
        newNum = '+55' + prefix + removeLetters
        numFormat = f"{newNum[:3]} ({newNum[3:5]}) {newNum[5:9]}-{newNum[9:]}"
    return numFormat

def insertStatePrefix(row):
    if row == '21':
        str = 'Greater Rio de Janeiro and Teresopolis'
    elif row == '22':
        str = 'East and North'
    elif row == '24':
        str = 'West'
    return str
```

```python
def checkId(id):
    if id == '0' or id.lstrip('0') == '':
        return ''
    removeLetters = re.sub( pattern: r'[a-zA-Z]', repl: '',id)
    if len(removeLetters) > 11:
        removeZeros = removeLetters.lstrip('0')
        fixZeros = removeZeros[:11].zfill(11)
    else:
        fixZeros = removeLetters.zfill(11)
    if len(fixZeros) == 11:
        idFormat = f"{fixZeros[:3]}.{fixZeros[3:6]}.{fixZeros[6:9]}-{fixZeros[9:]}"
        return idFormat

    return fixZeros


def setCapitaleFirstLetter(name):
    words = name.split()
    capitalized_words = [word.capitalize() for word in words]
    return ' '.join(capitalized_words)
```

מטלת בית
עמית ישי
קבלה 7/3/24 19:50 הגשה 9/3/24 21:00

```python
def replacePatterns(name):
    replacePatterns = {
        '&#259;' : 'a',
        '&#1091;' : 'y',
        '&#1047;' : 'c',
        '&#30136;' : 'ee',
        '&#227;' : 'a'
    }
    pattern = '|'.join(re.escape(key) for key in replacePatterns.keys())
    return re.sub(pattern, lambda match: replacePatterns[match.group()], name)
```

```python
def replaceMarkInFullName(name):
    characters = ['!','%','*',',','-','/',':',';','#',"'"]
    ...
    for char in characters:
        name = name.replace(char, '')
    if '?' in name:
        name = name.replace('?','a')
    if name.startswith('? ') or name.endswith(' ?'):
        name = name.replace('?', '')
    ...
    if name.startswith('.'):
        name = name[1:]
    if name.endswith('.'):
        name = name[:-1]

    name = ''.join(char for char in name if not char.isdigit())

    while '(' in name and ')' in name:
        start_index = name.find('(')
        end_index = name.find(')')
        if start_index < end_index:
            name = name[:start_index] + name[end_index + 1:]
        else:
            break
    # if '(' in name:
    #     name = name.split('(', 1)[0]

    return name
```

מטלת בית
עמית ישי
קבלה 7/3/24 19:50 הגשה 9/3/24 21:00

שמירה לקובץ json:

```python
data = {"csvToJsonFile": []}

with open("new_table.csv", "r") as f:
    reader = csv.reader(f)
    next(reader)
    for row in reader:
        if len(row) >= 11:
            data["csvToJsonFile"].append({
                "ID": row[0],
                "fullName": row[1],
                "prefix": row[2],
                "statePrefix": row[3],
                "mobileNumber": row[4],
                "state": row[5],
                "city": row[6],
                "neighborhood": row[7],
                "street": row[8],
                "houseDetails1": row[9],
                "houseDetails2": row[10]
            })
        else:
            print(f"Row skipped: {row}")

with open("csvToJsonFile.json", "w") as f:
    json.dump(data, f, indent=4)
```