COSE215
Project #1 Report
2019320140 Joon Ho Byun
Due Date: 11/20/2023

Goal of Project

- Protection of Data
- Practicing Regex in Python
- Finding valid Korean ID values and replacing them with "*"
- Input file : input.txt
- Output file: output.txt (including "*" updates for all valid ID numbers)

Explanation of Code

Helper Functions

- date_validity (checks if month and date correspond to each other)
- checksum_validity (checks if checksum value is correct)

```
def date_validity(month, day):
    if month == 2:
        if day > 28:
            return False
    elif month == 4 or month == 6 or month == 9 or month == 11:
        if day > 30:
            return False
    else:
        if day > 31:
            return False
    return True
```

The function date_validity takes in a month and day to check if the day value corresponds to the month, and returns True/False if it is valid. Leap years have not been taken into account, as it is outside the scope of this project. This could be easily accomplished by creating a separate function that checks the year and returns if it is a leap year.

```
def checksum_validity(match):

    match = match.replace("-", "")
    values = list(match)
    values.pop()
    multiplier = 2
    total = 0
    for i in range(len(values)):

        total += int(values[i]) * multiplier
        multiplier += 1

        if multiplier == 10:
            multiplier = 2

    total = total % 11
    total = 11 - total
    if total > 9:
        total = total % 10
    return(total)
```

The checksum_validity checks if checksum is correct by first removing the "-" and splitting by character. A local multiplier variable is incremented and changed back to 2 when it arrives to 10. Moreover, if checksum value is greater than 9, then we use modulus to change it back to a single digit: 10 = 0, and 11 = 1. This value is returned to be compared later on.

Main Function

```
def main():
    protectedBody = ""
    fres = open(OUTPUT, 'w', encoding = "UTF-8")
    with open(INPUT, 'r', encoding = "UTF-8") as fp:
        body = ''.join(fp.readlines())
        print ("** UNPROTECTED INPUT **")
        print (body) # Check the input file.

        pattern = re.compile(r'\d\d\d\d\d\d\s*-\s*\d\d\d\d\d\d\d')
        matches = pattern.findall(body)
```

By using this pattern, we store all matches which have a single or no space between the dash.

```
        for match in matches:
            valid = True
            twenty_century = False
            matchCheck = match.replace(" ", "")

            year = int(matchCheck[0:2])
            month = int(matchCheck[2:4])
            day = int(matchCheck[4:6])
            gender = int(matchCheck[7])

           #year, month, day and gender is extracted to run checks

            if year >= 24:
                twenty_century = True
           #a boolean value to check between 1924 - 1999 or 2000+

            if date_validity(month, day) == False:
                print("Invalid Date: " + match)
                valid = False
                continue
           #run date_validity function
            if (((twenty_century == True) and gender not in {1,2}) or
((twenty_century == False) and gender not in {3,4})):
                print("Invalid Gender: " + match)
                valid = False
                continue

            if(checksum_validity(matchCheck) != int(matchCheck[13])):
                print("Invalid Checksum: " + match)
                valid = False
                continue

            print("Valid Registration Number: " + match)
            body = body.replace(match, "******-*******")
            #Replace the valid values with asterisks.

        print ("** PROTECTED OUTPUT **")
        protectedBody = body
        print (protectedBody)
        fres.write(protectedBody)
        fres.close()

""" EXECUTE """
if __name__ == "__main__":
    main()
```

This main function goes through each validity check, and if its not valid, it continues onto the next match. It also prints the reason it isn't valid (the first reason why it isnt valid). At the end, we replace values found to be valid and set protectedBody as this new updated body and write to the output.txt file.

Output