```
Customer & Product Text Analysis for Marketing Insights
```

Copy

```
#### How can we validate and correct store ZIP Code data when leading zeros are missing, while flagging invalid ZIP Codes that do not meet U.S. formatting rules?

def zip_checker(zipcode):
    if len(zipcode) == 5:
        if zipcode[0:2] =='00':
            return 'Invalid ZIP Code.'
        else:
            return zipcode
    elif zipcode[0] != '0':
        zipcode = '0' + zipcode
        return zipcode
    else:
        return 'Invalid ZIP Code.'
### Test Fuctions
print(zip_checker('02806'))
print(zip_checker('2806'))
print(zip_checker('0280'))
print(zip_checker('00280'))
```

```
02806
02806
Invalid ZIP Code.
Invalid ZIP Code.
```

```
### How can we validate store URLs to ensure they use the correct https: protocol and contain a store ID that is exactly seven characters long?

def url_checker (url):
    url = url.split('/')
    protocol = url[0]
    store_id = url[-1]

    if protocol != 'https:' and len(store_id) != 7:
        print(f'{protocol} is an invalid protocol.',
            f'\n{store_id} is an invalid store ID.')
    elif protocol != 'https:' :
        print (f'{protocol} is an invalid protocol.')
    elif len(store_id) != 7:
        print (f'\n{store_id} is an invalid store ID.')
    else:
        return store_id
## Test Functions
url_checker('http://exampleURL1.com/r626c3')
print()
url_checker('ftps://exampleURL1.com/r626c36')
print()
url_checker('https://exampleURL1.com/r626c3')
print()
url_checker('https://exampleURL1.com/r626c36')
```

```
http: is an invalid protocol.
r626c3 is an invalid store ID.

ftps: is an invalid protocol.


r626c3 is an invalid store ID.
```

```
'r626c36'
```