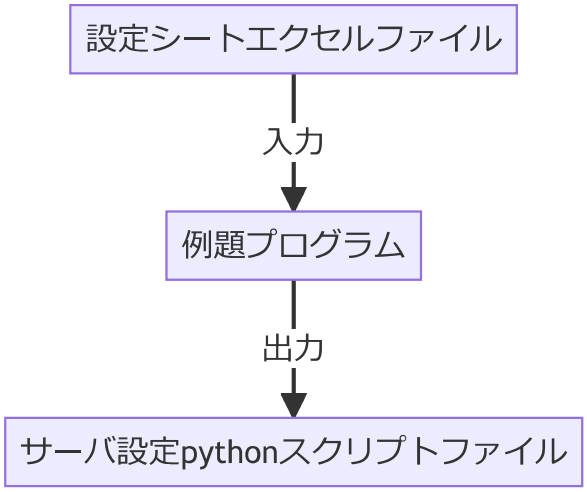
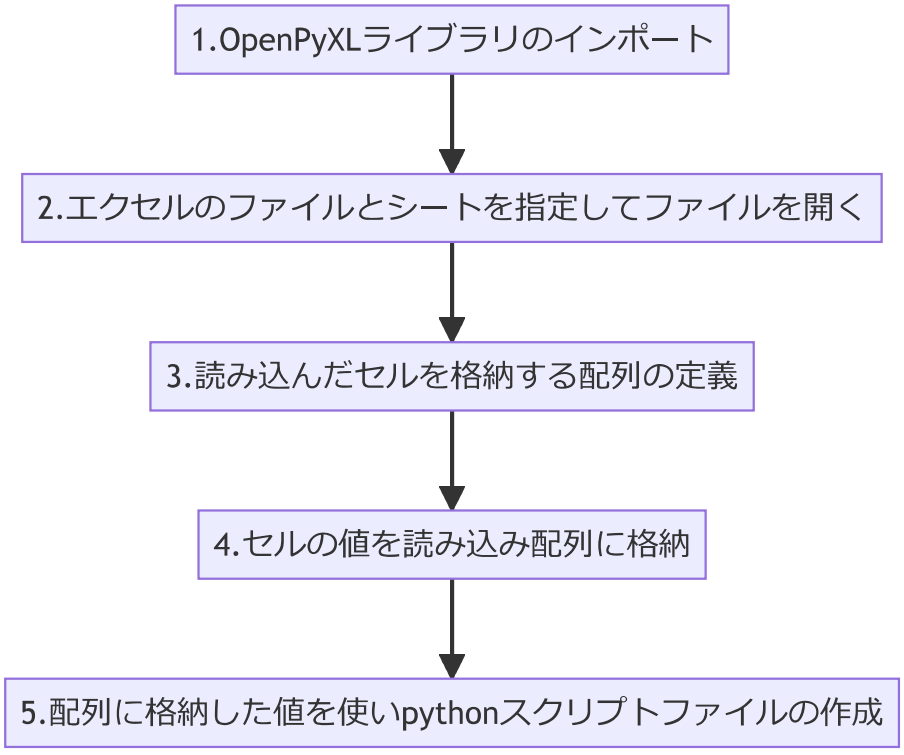
例題プログラムの機能

設定シートから設定情報を読み込み、 Linuxで実行可能なサーバ設定用のpythonスクリプトファイルを作成

例題プログラムの入力と出力



例題プログラム全体の流れ:



サーバ設定pythonスクリプトファイル実行時に実施されるコマンド

```
#IPアドレスの設定
nmcli connection modify ens192 ipv4.address 172.19.6.20/24
#デフォルトゲートウェイの設定
nmcli connection modify ens192 ipv4.gateway 172.19.6.254
#IPアドレス設定の手動化(DHCP OFF)
nmcli connection modify ens192 ipv4.method manual
#インターフェースの自動起動
nmcli connection modify ens192 connection.autoconnect yes

nmcli connection modify ens224 ipv4.address 192.168.10.22/24
nmcli connection modify ens224 ipv4.method manual
nmcli connection modify ens224 connection.autoconnect yes

nmcli connection modify ens256 ipv4.address 192.168.20.22/24
nmcli connection modify ens256 ipv4.method manual
nmcli connection modify ens256 connection.autoconnect yes

#サービスの再起動
systemctl restart NetworkManager

#インターフェースの再起動
nmcli connection down ens192
nmcli connection up ens192
nmcli connection down ens224
nmcli connection up ens224
nmcli connection down ens256
nmcli connection up ens256
```

1.OpenPyXLライブラリのインポート

```
#import 【ライブラリ名】でライブラリをimport
import openpyxl
```

OpenPyXLとは

OpenPyXLとは、Excelファイルを Pythonで操作するためのライブラリ OpenPyXLを使うことで、Excelファイルのシートやセルからデータを取り出したり、セルにデータ書き出したり様々な操作が可能

プログラミングにおけるライブラリとは

事前作成されたコード(モジュール)をまとめたもの、 利用することで効率的にプログラムを作成できる ライブラリごとに特定の問題を対象としている

2.エクセルのファイルとシートを指定してファイルを開く

```
# エクセルファイルのパスとシート名
file_path = "linux_config.xlsx"
sheet_name = "interface"

# エクセルファイルを開く
workbook = openpyxl.load_workbook(file_path)
sheet = workbook[sheet_name]
```

セルの値の取り出し方

```
interface = sheet["B3"].value

print(interface)
ens192
```

空のセルの値

```
interface = sheet["B6"].value

print(interface)
None
```

3.読み込んだセルを格納する配列の定義

```
interfaces = []
ipaddresses = []
dg = []
bootprots = []
autoconnects = []
```

配列への値の格納と取り出し方法

```
interfaces = []
interfaces.append("ens192")
interfaces.append("ens224")
```

```
interfaces.append("ens256")

print(interfaces)
['ens192', 'ens224', 'ens256']

print(interfaces[0])
ens192

print(interfaces[1])
ens224

print(interfaces[2])
ens256
```

4.セルの値を読み込み配列に格納

```
row = 3

while  True:
    interface = sheet["B" + str(row)].value
    if interface is  None:
        break
    interfaces.append(interface)

    ipaddress = sheet["C" + str(row)].value
    ipaddresses.append(ipaddress)

    default_gateway = sheet["D" + str(row)].value
    dg.append(default_gateway)

    bootprot = sheet["E" + str(row)].value
    bootprots.append(bootprot)

    autoconnect = sheet["F" + str(row)].value
    autoconnects.append(autoconnect)

    row += 1

# エクセルファイルを閉じる
workbook.close()
```

whileループ内で2回目の「interfaces.append(interface)」で配列「interfaces」に入る値は？

while文の使用例

```
a = 0
while  True:
    print(a)
    a = a + 1
    if a == 10:
        break
0
1
2
3
4
5
6
7
8
9
```

例題プログラムで配列に格納された値

```
print(interfaces)
['ens192', 'ens224', 'ens256']

print(ipaddresses)
['172.19.6.20/24', '192.168.10.22/24', '192.168.20.22/24']

print(dg)
['172.19.6.254', None, None]

print(bootprots)
['manual', 'manual', 'manual']

print(autoconnects)
['yes', 'yes', 'yes']
```

5.配列に格納した値を使いpythonスクリプトファイルの作成

```
# Pythonスクリプトファイルを作成してコマンドを出力
with  open("configure_interfaces.py", "w") as f:
    f.write("#!/usr/bin/python3\n\n")
    f.write("import subprocess\n\n")
    f.write("# インターフェースとIPアドレスの設定\n")

    for interface, ipaddress, default_gateway, bootprot, autoconnect in
zip(interfaces, ipaddresses, dg, bootprots, autoconnects):
```

```
        f.write(f"subprocess.run(['nmcli', 'connection', 'modify', '{interface}',
'ipv4.address', '{ipaddress}'])\n")

        if default_gateway:
            f.write(f"subprocess.run(['nmcli', 'connection', 'modify',
'{interface}', 'ipv4.gateway', '{default_gateway}'])\n")

        f.write(f"subprocess.run(['nmcli', 'connection', 'modify', '{interface}',
'ipv4.method', '{bootprot}'])\n")

        f.write(f"subprocess.run(['nmcli', 'connection', 'modify', '{interface}',
'connection.autoconnect', '{autoconnect}'])\n")

    f.write("# NetworkManagerのサービス再起動\n")
    f.write("subprocess.run(['systemctl', 'restart', 'NetworkManager'])\n\n")



    f.write("# インターフェースの再起動\n")
    for interface in interfaces:
        f.write(f"subprocess.run(['nmcli', 'connection', 'down',
'{interface}'])\n")
        f.write(f"subprocess.run(['nmcli', 'connection', 'up',
'{interface}'])\n")
f.close()
```

forループ内で初回の「f.write(f"subprocess.run(['nmcli', 'connection', 'modify', '{interface}', 'ipv4.address', '{ipaddress}'])\n")」で{interface}、{ipaddress}の展開後の値は？

例題プログラムで出力させるpythonスクリプトファイル

```
#!/usr/bin/python3

import subprocess

# インターフェースとIPアドレスの設定
subprocess.run(['nmcli', 'connection', 'modify', 'ens192', 'ipv4.address',
'172.19.6.20/24'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens192', 'ipv4.gateway',
'172.19.6.254'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens192', 'ipv4.method',
'manual'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens192',
'connection.autoconnect', 'yes'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens224', 'ipv4.address',
'192.168.10.22/24'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens224', 'ipv4.method',
'manual'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens224',
'connection.autoconnect', 'yes'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens256', 'ipv4.address',
```

```
'192.168.20.22/24'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens256', 'ipv4.method',
'manual'])
subprocess.run(['nmcli', 'connection', 'modify', 'ens256',
'connection.autoconnect', 'yes'])

# NetworkManagerのサービス再起動
subprocess.run(['systemctl', 'restart', 'NetworkManager'])

# インターフェースの再起動
subprocess.run(['nmcli', 'connection', 'down', 'ens192'])
subprocess.run(['nmcli', 'connection', 'up', 'ens192'])
subprocess.run(['nmcli', 'connection', 'down', 'ens224'])
subprocess.run(['nmcli', 'connection', 'up', 'ens224'])
subprocess.run(['nmcli', 'connection', 'down', 'ens256'])
subprocess.run(['nmcli', 'connection', 'up', 'ens256'])
```

サーバが実行するコマンド

```
#IPアドレスの設定
nmcli connection modify ens192 ipv4.address 172.19.6.20/24
#デフォルトゲートウェイの設定
nmcli connection modify ens192 ipv4.gateway 172.19.6.254
#IPアドレス設定の手動化(DHCP OFF)
nmcli connection modify ens192 ipv4.method manual
#インターフェースの自動起動
nmcli connection modify ens192 connection.autoconnect yes

nmcli connection modify ens224 ipv4.address 192.168.10.22/24
nmcli connection modify ens224 ipv4.method manual
nmcli connection modify ens224 connection.autoconnect yes

nmcli connection modify ens256 ipv4.address 192.168.20.22/24
nmcli connection modify ens256 ipv4.method manual
nmcli connection modify ens256 connection.autoconnect yes

#サービスの再起動
systemctl restart NetworkManager

#インターフェースの再起動
nmcli connection down ens192
nmcli connection up ens192
nmcli connection down ens224
nmcli connection up ens224
nmcli connection down ens256
nmcli connection up ens256
```

subprocessとは

利用することでpythonからOSコマンドが実行可能

利用例

1. pythonファイル「nmcli.py」を作成

```
import subprocess

subprocess.run(['nmcli', 'connection', 'modify', 'ens192', 'ipv4.address',
'172.19.6.20/24'])
```

2. サーバに「nmcli.py」を配置し、実行

```
$ python3 nmcli.py
```

3. サーバで以下コマンドが実行される

```
nmcli connection modify ens192 ipv4.address 172.19.6.20/24
```

zipで複数の要素をまとめて取得させる方法

```
interfaces = ['ens192', 'ens224', 'ens256']
ipaddresses = ['172.19.6.20/24', '192.168.10.22/24', '192.168.20.22/24']
dg = ['172.19.6.254', None, None]
bootprots = ['manual', 'manual', 'manual']
autoconnects = ['yes', 'yes', 'yes']

for interface, ipaddress, default_gateway, bootprot, autoconnect in
zip(interfaces, ipaddresses, dg, bootprots, autoconnects):

    print(interface)
    print(ipaddress)
    print(default_gateway)
    print(bootprot)
    print(autoconnect)
    print("\n")

ens192
172.19.6.20/24
172.19.6.254
manual
yes

ens224
192.168.10.22/24
None
manual
yes
```

```
ens256
192.168.20.22/24
None
manual
yes
```

with openによるファイル作成

```
# Pythonスクリプトファイルを作成してコマンドを出力
with  open("configure_interfaces.py", "w") as f:
    f.write("#!/usr/bin/python3\n\n")
    f.write("import subprocess\n\n")
    f.write("# インターフェースとIPアドレスの設定\n")

    interface = "ens192"
    ipaddress = "172.19.6.20/24"
    #f文字列にて{}で囲まれた箇所を変数展開
    f.write(f"subprocess.run(['nmcli', 'connection', 'modify', '{interface}',
'ipv4.address', '{ipaddress}'])\n")

    f.close()
```

作成されたファイルの内容

```
#!/usr/bin/python3

import subprocess

# インターフェースとIPアドレスの設定
subprocess.run(['nmcli', 'connection', 'modify', 'ens192', 'ipv4.address',
'172.19.6.20/24'])
```

pythonスクリプトファイルの出力ではなくテキストを出力する場合

```
for interface, ipaddress, default_gateway, bootprot, autoconnect in
zip(interfaces, ipaddresses, dg, bootprots, autoconnects):
  print(f"nmcli connection modify {interface} ipv4.address {ipaddress}")
  if default_gateway:
    print(f"nmcli connection modify {interface} ipv4.gateway {default_gateway}")
  print(f"nmcli connection modify {interface} ipv4.method {bootprot}")
  print(f"nmcli connection modify {interface} connection.autoconnec
{autoconnect}")
  print("\n")

print("systemctl restart NetworkManagre")
print("\n")
```

```
  for interface in interfaces:
    print(f"nmcli connection down {interface}")
    print(f"nmcli connection up {interface}")
    print("\n")
```

参考情報

コマンドによるファイル編集

`/etc/selinux/config` 変更前

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=enforcing
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

コマンド実行("SELINUX="で始まる文字列を置換)

```
sed -i "s/^SELINUX=.*/SELINUX=disabled/" '/etc/sysconfig/selinux'
```

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#       enforcing - SELinux security policy is enforced.
#       permissive - SELinux prints warnings instead of enforcing.
#       disabled - No SELinux policy is loaded.
SELINUX=disabled
# SELINUXTYPE= can take one of these two values:
#       targeted - Targeted processes are protected,
#       mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

設定項目名を検索して値を取得

以下の例ではB列(row[1])から項目"selinux"を検索し値"disabled"を代入

```
import openpyxl

# エクセルファイルの読み込み
```

```
workbook = openpyxl.load_workbook('linux_config.xlsx')
sheet = workbook['others']

# "selinux"を検索してC列の値を取得
selinux = None
for row in sheet.iter_rows():
    cell_value = row[1].value # B列の値を取得
    if cell_value == "selinux":
        if  len(row) > 2: # 右隣のセルが存在するか確認
            selinux = row[2].value
        break

print(selinux)
disabled
```