

UNIT 1.5 GRADED ASSIGNMENT

GOOD SOFTWARE ENGINEERING PRACTICES

Group members

Ifra Saleem (2303.khi.deg.003)

Muhammad Khan (2303.khi.deg.027)

UNIT 1.5 GRADED ASSIGNMENT

Task:

Refactor following code:

```
from typing import List
import pandas as pd
class User:
    sub: bool

def notify(user: User) -> None:
    Pass

def notify_users(x: List[User]) -> None:
    #Filter users with subscription and notify them.
    for u in x:
        if u.sub:
            # u.notify()
            notify(u)
```

Solution:

```

1  from typing import List
2
3  class User:
4      def __init__(self, subscribed: bool):
5          self.subscribed = subscribed
6
7      def notify(self) -> None:
8          (function) def notify_subscribed_users(users: List[User]) -> None:
9              Notify subscribed users.
10
11  def notify_subscribed_users(users: List[User]) -> None:
12      """Notify subscribed users."""
13      subscribed_users = get_subscribed_users(users)
14      for user in subscribed_users:
15          user.notify()
16
17  def get_subscribed_users(users: List[User]) -> List[User]:
18      """Filter subscribed users."""
19      return [user for user in users if user.subscribed]
20
21

```

from typing import List

class User:

def __init__(self, subscribed: bool):

self.subscribed = subscribed

def notify(self) -> None:

pass

def notify_subscribed_users(users: List[User]) -> None:

"""Notify subscribed users."""

subscribed_users = get_subscribed_users(users)

for user in subscribed_users:

user.notify()

def get_subscribed_users(users: List[User]) -> List[User]:

"""Filter subscribed users."""

return [user for user in users if user.subscribed]

Explanation:

- According to the YAGNI (You aren't gonna need it) I removed pandas library because we are not using it in the code.
- Then according to the naming rules, I changed the variable names like I replaced **sub** with **subscribed**, **u** with **user**. I also changed function names like I replaced **notify_users** with **notify_subscribed_users**. I added another function **get_subscribed_users** to get a list of all the subscribed users because according to the good software engineering practices a function should do only one thing. And in the task code **notify_users** is doing two things, one is to check that the user is subscribed or not and second is to notify the subscribed users. Now in the refactored code **get_subscribed_users** will filter the subscribers and **notify_subscribed_users** will send the notification.
- I also created a constructor in the class user and I moved the **notify()** function inside the User class because the user class should have the method to notify themselves.
- And I placed the caller function above the callee functions. The caller function is **notify_subscribed_users()** and the callee function is **get_subscribed_users()**.