```
...
import tsect
T=tsect()
class x:
    def mts(self,p,a,b):
        global T
        l=[k for k in p if k.p(a)]
        for i in l:
            if T.calc(i)>3.456:
                i.pst=1
            elif T.calc(i)<=3.456 and T.calc(i)>1.45:
                i.pst=2
                T.pl=False
...
ep=x()
ep.mts(pnl1,34,zx)
…
```

Observações: As mais fáceis em primeiro lugar:

1. Não está seguindo um padrão aceitável de nomenclaturas:
    1. Clean code G20 – Nomes de função devem dizer o que elas fazem;
    2. Clean code G24 – Seguir convenções padronizadas de codificação;
    3. Clean code N1 – Escolha nomes descritivos;
2. Uso de “números mágicos”:
    1. Clean code G25 – Substitua números “mágicos” por constantes nomeadas
3. Argumento não utilizado:
    1. Talvez Clean code F1 – Muitos argumentos;

Agora as observações mais profundas:

4. O método está alterando propriedades dos argumentos e não há indicação em seu nome de que fará isso:
    1. Clean code F2 – Argumentos de “saída”;
5. O método está alterando propriedades de variáveis de escopo superior:
    1. Isso é Acoplamento Comum e é bem descrito na literatura;
6. Parece que a intenção do método está implementada incorretamente:
    1. A função percorre os elementos de uma lista passada como argumento (“p”) modificando o estado de alguns elementos dependendo de outro argumento passado (“a”). Isso lembra o padrão “Visitor”… Talvez devesse ser um método de uma classe que encapsulasse a lista “p”, com um nome significativo. E a alteração de propriedades do objeto global “T” deveria ser feita no escopo superior. Talvez seja Clean code G21 – Entender o algoritmo;