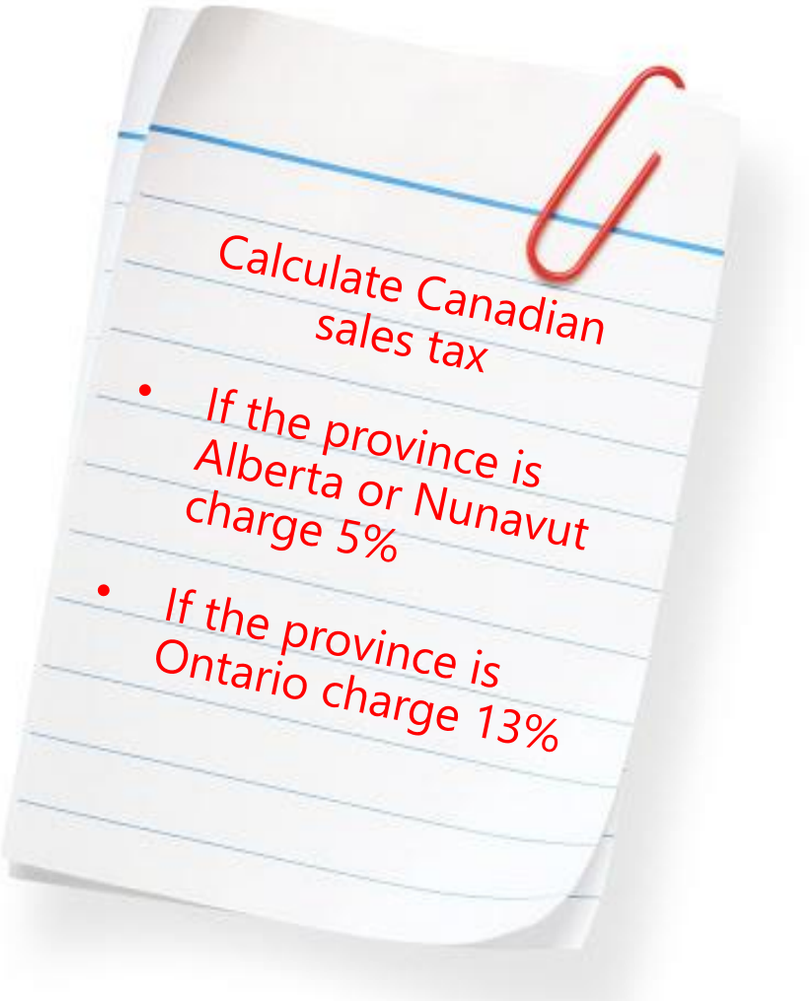
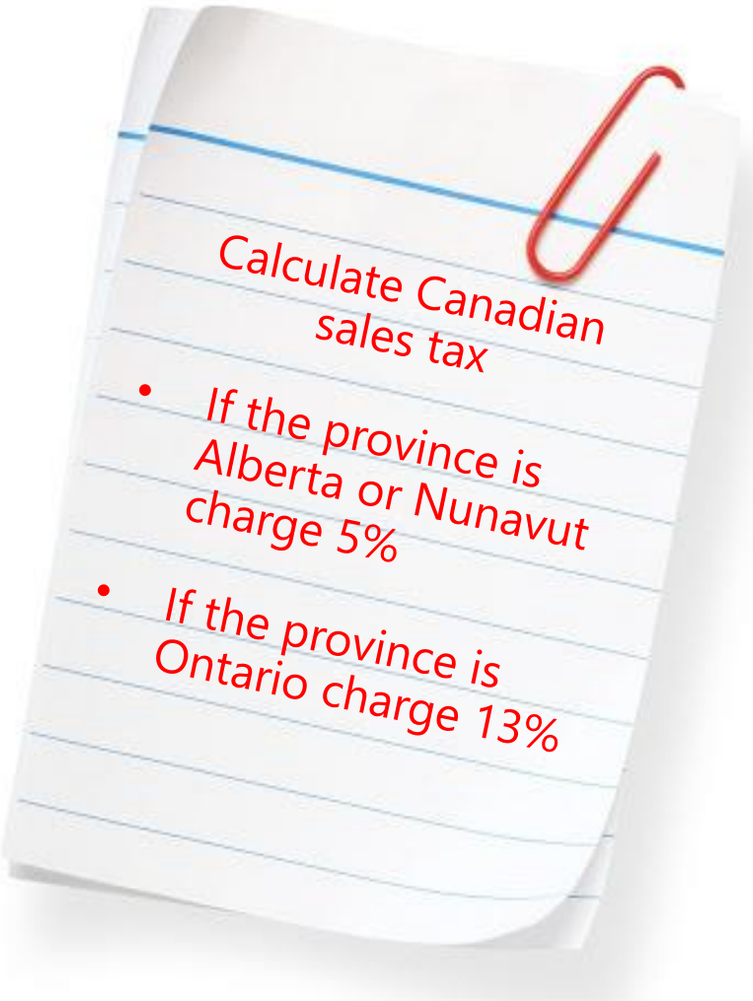
You may need to check multiple conditions to determine the correct action

```
if province == 'Alberta':
    tax = 0.05
if province == 'Nunavut':
    tax = 0.05
if province == 'Ontario':
    tax = 0.13
```



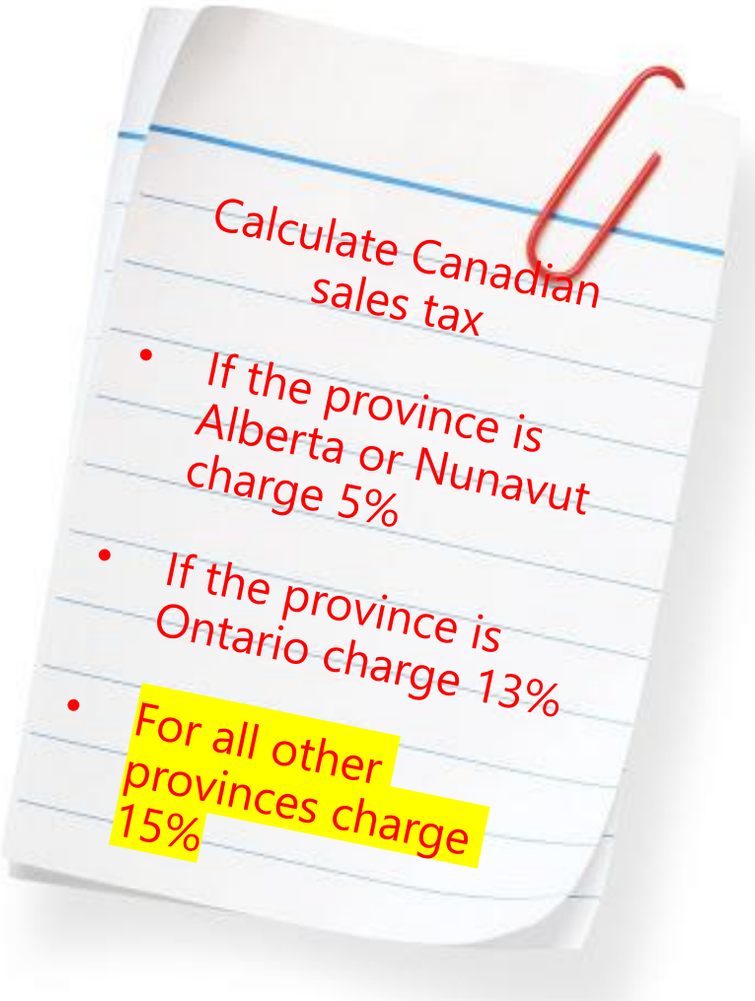
If only one of the conditions will ever occur you can use a single if statement with elif

```
if province == 'Alberta':
    tax = 0.05
elif province == 'Nunavut':
    tax = 0.05
elif province == 'Ontario':
    tax = 0.13
```



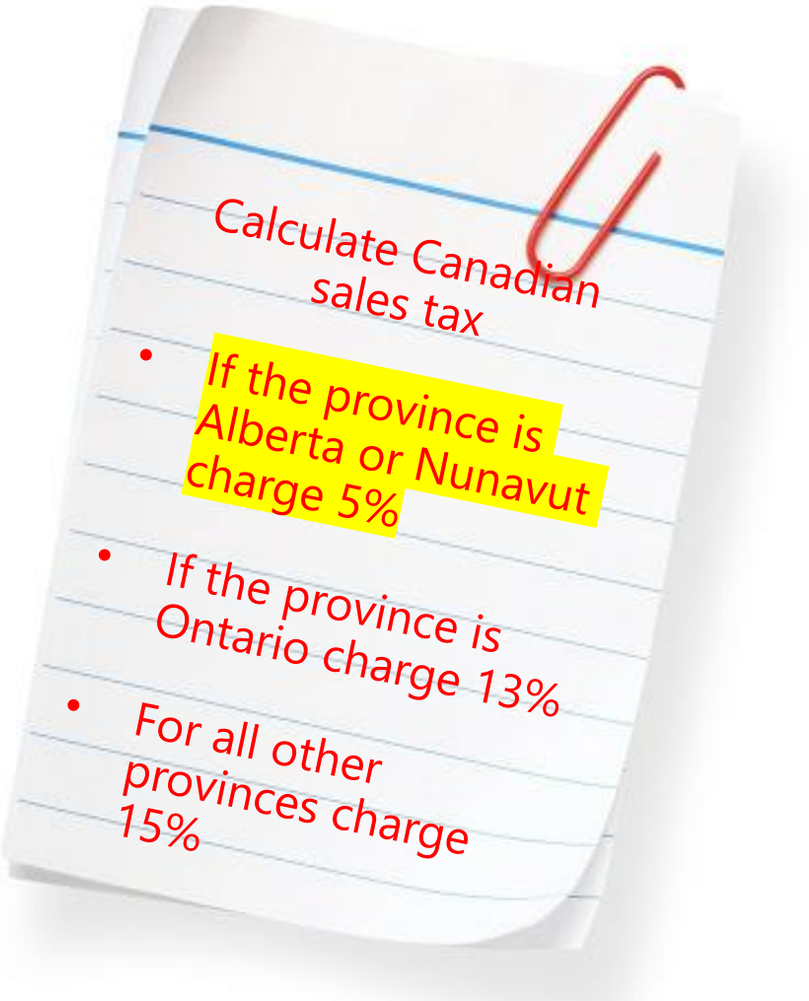
When you use elif instead of multiple if statements you can add a default action

```
if province == 'Alberta':
    tax = 0.05
elif province == 'Nunavut':
    tax = 0.05
elif province == 'Ontario':
    tax = 0.13
else:
    tax = 0.15
```



If multiple conditions cause the same action they can be combined into a single condition

```
if province == 'Alberta' \
    or province == 'Nunavut':
      tax = 0.05
elif province == 'Ontario':
      tax = 0.13
else:
      tax = 0.15
```

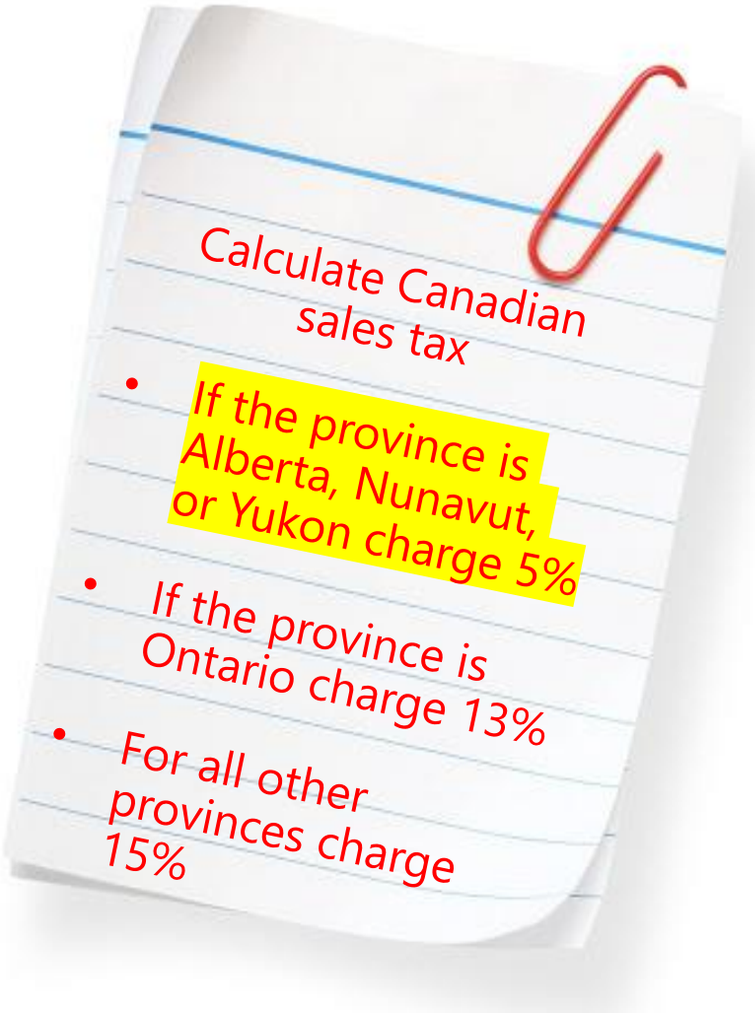


How OR statements are processed

First Condition	Second Condition	Condition evaluates as
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

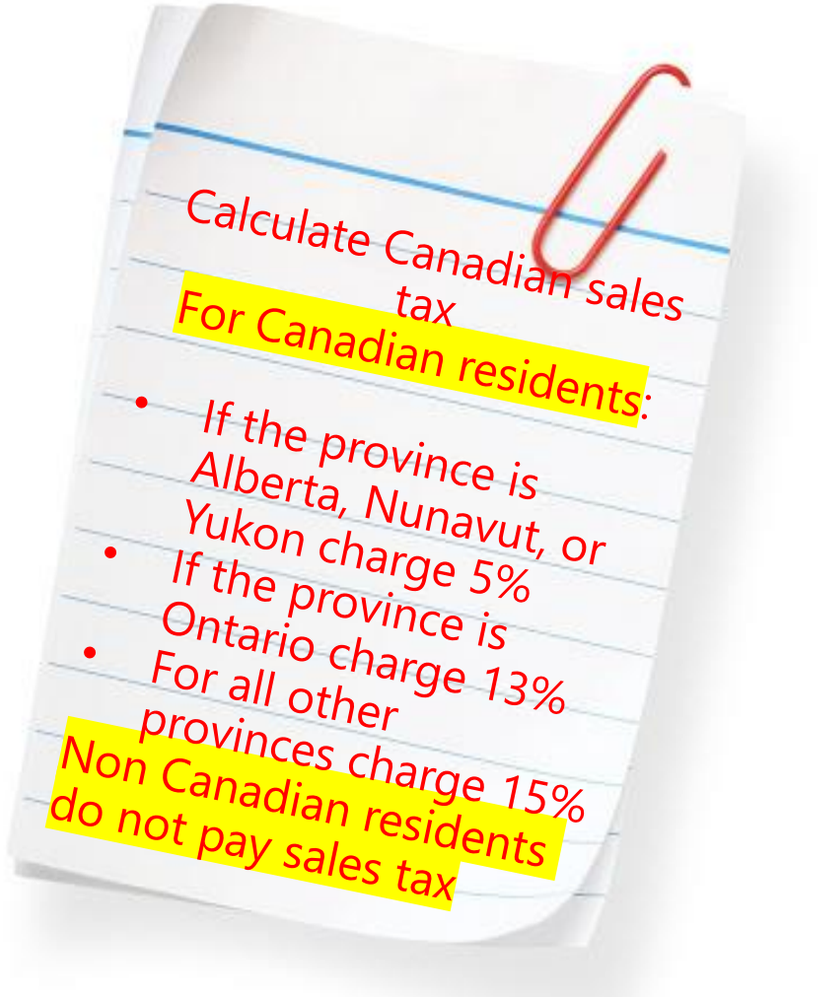
If you have a list of possible values to check , you can use the IN operator

```
if province in('Alberta', \
                'Nunavut','Yukon'):
    tax = 0.05
elif province == 'Ontario':
    tax = 0.13
else:
    tax = 0.15
```



If an action depends on a combination of conditions you can nest if statements

```
if country == 'Canada':
    if province in('Alberta',\
                   'Nunavut','Yukon'):
        tax = 0.05
    elif province == 'Ontario':
        tax = 0.13
    else:
        tax = 0.15
else:
    tax = 0.0
```



Handling real world business rules may require complicated conditional logic, but the code can still be written efficiently

Apply appropriate state or federal taxes based on location
Calculate salary based on job level