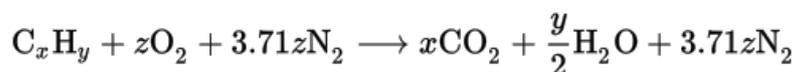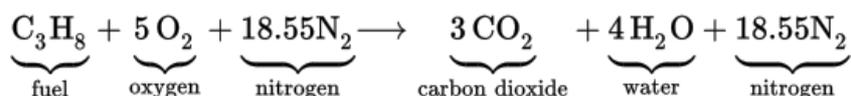**Chemistry Challenge #1**

As a chemist working at Tanzania's largest propane supplier, you have been asked to write some software that verifies the accuracy of the chemical equations used by the company. One of the many equations used by the company involves the combustion of a hydrocarbon in air:

$$C_xH_y + zO_2 + 3.71zN_2 \longrightarrow xCO_2 + \frac{y}{2}H_2O + 3.71zN_2$$

Specifically, your company's most common hydrocarbon is propane:

$$\underbrace{C_3H_8}_{\text{fuel}} + \underbrace{5\,O_2}_{\text{oxygen}} + \underbrace{18.55N_2}_{\text{nitrogen}} \longrightarrow \underbrace{3\,CO_2}_{\text{carbon dioxide}} + \underbrace{4\,H_2O}_{\text{water}} + \underbrace{18.55N_2}_{\text{nitrogen}}$$

One of your fellow employees was originally assigned to work on this project, but was unable to complete it. This is the program so far:

```python
# This function returns a propane-in-air combustion equation with
# the left hand side correct but the right hand side empty.
def get_blank_propane_air_combustion_equation(num_propane_molecules):
    num_oxygen_molecules = 5 * num_propane_molecules
    num_nitrogen_molecules = 3.71 * num_oxygen_molecules
    return (num_propane_molecules, num_oxygen_molecules, \
            num_nitrogen_molecules), (0, 0, 0)

# This function checks the right hand side of a propane-in-air
# combustion equation to ensure it is correct.
def check_propane_air_combustion_equation(lhs, rhs):
    if rhs[0] / lhs[0] != 3:
        return "Incorrect number of CO2 molecules."
    elif (rhs[1] * 2) / 8 != lhs[0]:
        return "Incorrect number of H2O molecules."
    elif rhs[2] != lhs[2]:
        return "Incorrect number of N2 molecules."
    else:
        return "CORRECT"

for n in range(1, 11):
    lhs, rhs = get_blank_propane_air_combustion_equation(n)
    print("LHS: %s\tRHS: %s" % (lhs, rhs))
    print("Empty: %dC3H8 + %dO2 + %.2fN2 --> %dCO2 + %dH2O + %.2fN2" % \
        (lhs[0], lhs[1], lhs[2], rhs[0], rhs[1], rhs[2]))
    # Make changes here to fill the right hand side of the equation
    # with the correct values.
    print("Filled: %dC3H8 + %dO2 + %.2fN2 --> %dCO2 + %dH2O + %.2fN2" % \
        (lhs[0], lhs[1], lhs[2], rhs[0], rhs[1], rhs[2]))
    print "%s\n" % check_propane_air_combustion_equation(lhs, rhs)
```

As you can see, two functions are defined, and then the program generates equations using one to ten propane molecules. Notice that the first function returns two **tuples**. In Python, a **tuple** is a series of values in between "(" and ")", each separated by a comma. For example, this is a tuple of five letters:

$$name = ('B', 'r', 'i', 'a', 'n')$$

Tuples are useful in situations where we want to group multiple related values together. They are very similar to **list**s in Python. Just like with lists, you can use them in a for loop:

```
name = ('B', 'r', 'i', 'a', 'n')
for letter in name:
        print letter
```

and you can get the individual elements inside:

```
name = ('B', 'r', 'i', 'a', 'n')
print name[1]  # This will print the letter "r"
```

but *you cannot change a tuple once it has been created.* From working with lists, you know that we can change an element in a list:

```
names = ['Brian', 'John']
names[1] = 'Magret'
print names   # This will print ['Brian', 'Magret']
```

But if you try to do the same with a tuple:

```
name = ('B', 'r', 'i', 'a', 'n')
name[1] = 'R'
```

Python will give you this error:

```
TypeError: 'tuple' object does not support item assignment
```

For this challenge, your goal is to change the code inside of the for loop in the program on the previous page so that the `rhs` variable contains the correct tuple of numbers representing the output of burning propane in air. Notice that the `check_propane_air_combustion_equation(lhs, rhs)` function takes two tuples, one for the left hand side of the equation and the other for the right, and prints out whether the equation is right or wrong.

Keep in mind: you cannot change the code inside either of the two functions; they must remain the same. All of your changes must be made inside the for loop.

Using your tablet, type the program on the previous page into a new file in QPython. Experiment with the program to understand how it works, then make your changes inside the for loop. Run the program to check your work: when all ten equations display "CORRECT" beneath them, you'll have succeeded. Use your knowledge of combustion, chemical equations, and Python tuples to make the necessary changes. When you're finished, bring your tablet to your ICT Practical instructor so he can review both the program and the output.