**1.** Given a list of filenames, we want to rename all the files with extension hpp to the extension h. To do this, we would like to generate a new list called newfilenames, consisting of the new filenames. Fill in the blanks in the code using any of the methods you’ve learned thus far, like a for loop or a list comprehension.

filenames = ["program.c", "stdio.hpp", "sample.hpp", "a.out", "math.hpp", "hpp.out"]

# Generate newfilenames as a list containing the new filenames

# using as many lines of code as your chosen method requires.

print(newfilenames)

newfilenames = [filename.replace("hpp", "h") if filename[-3:] == "hpp" else filename for filename in filenames]

# Should be ["program.c", "stdio.h", "sample.h", "a.out", "math.h", "hpp.out"]

2.Let's create a function that turns text into pig latin: a simple text transformation that modifies each word moving the first character to the end and appending "ay" to the end. For example, python ends up as ythonpay.

def pig\_latin(text):

  say = ""

  # Separate the text into words

  words = text.split()

  for word in words:

    # Create the pig latin word and add it to the list

    # Turn the list back into a phrase

  return " ".join(pig\_latin\_words)

print(pig\_latin("hello how are you")) # Should be "ellohay owhay reaay ouyay"

    pig\_latin\_words.append(word[1:] + word[0] + "ay")

  pig\_latin\_words = []

print(pig\_latin("programming in python is fun")) # Should be "rogrammingpay niay ythonpay siay unfay"

3.The group\_list function accepts a group name and a list of members, and returns a string with the format: group\_name: member1, member2, … For example, group\_list("g", ["a","b","c"]) returns "g: a, b, c". Fill in the gaps in this function to do that.

def group\_list(group, users):

  members = group + ": " + ", ".join(users)

  return members

print(group\_list("Marketing", ["Mike", "Karen", "Jake", "Tasha"])) # Should be "Marketing: Mike, Karen, Jake, Tasha"

print(group\_list("Engineering", ["Kim", "Jay", "Tom"])) # Should be "Engineering: Kim, Jay, Tom"

print(group\_list("Users", "")) # Should be "Users:"

4. The guest\_list function reads in a list of tuples with the name, age, and profession of each party guest, and prints the sentence "Guest is X years old and works as \_\_." for each one. For example, guest\_list(('Ken', 30, "Chef"), ("Pat", 35, 'Lawyer'), ('Amanda', 25, "Engineer")) should print out: Ken is 30 years old and works as Chef. Pat is 35 years old and works as Lawyer. Amanda is 25 years old and works as Engineer. Fill in the gaps in this function to do that.

def guest\_list(guests):

    for guest in guests:

        name, age, prof = guest

        print("{} is {} years old and works as {}".format(name, age, prof))

guest\_list([('Ken', 30, "Chef"), ("Pat", 35, 'Lawyer'), ('Amanda', 25, "Engineer")])

#Click Run to submit code

"""

Output should match:

Ken is 30 years old and works as Chef

Pat is 35 years old and works as Lawyer

Amanda is 25 years old and works as Engineer

"""