Answer Key

Lesson 2

Task 1:

main.py

```
1  temperature_variability = "significant"
2  print(temperature_variability)
```

```
significant
```

Task 2:

main.py

```
1  average_increase = 2
2  increase_before_1981 = 0.07
3  increase_after_1981 = 0.18
4  print(average_increase)
5  print(increase_before_1981)
6  print(increase_after_1981)
7  print(increase_before_1981 + increase_after_1981)
```

```
2
0.07
0.18
0.25
```

Task 3:

main.py

```
1  temperature_variability = "significant"
2  average_increase = 2
3  increase_before_1981 = 0.07
4  increase_after_1981 = 0.18
5  climate_warming = True
6  climate_cooling = False
7  print(type(temperature_variability))
8  print(type(average_increase))
9  print(type(increase_before_1981))
10 print(type(increase_after_1981))
11 print(type(climate_warming))
12 print(type(climate_cooling))
```

```
<class 'str'>
<class 'int'>
<class 'float'>
<class 'float'>
<class 'bool'>
<class 'bool'>
```

Lesson 3:

Task 1:

main.py

```
1  sea_level_rising_rate = (1.7, 3.2, 6.4)
2  print(sea_level_rising_rate[1])
```

```
3.2
```

Task 2:

main.py

```
1  sea_level_rising_rate = [1.7, 3.2, 6.4, 12.0]
2  print(sea_level_rising_rate[-3])
```

```
3.2
```

Task 3:

main.py

```
1  sea_level_rising_rate = [1.7, 3.2, 6.4, 12.0]
2  print(sea_level_rising_rate[1:3])
```

```
[3.2, 6.4]
```

Lesson 4

Task 1:

main.py

```
1  my_dict = {'anthropogenic GHG':'caused by humans', 'coal':'fossil fuel',
   'CO2 in 2019': 409.8,}
2  print(my_dict)
```

```
{'anthropogenic GHG': 'caused by humans', 'coal': 'fossil fuel', 'CO2 in 2019': 409.8}
```

**Notes: no quotation marks are needed for integers or floats*

Task 2:

main.py

```
1  my_dict = {'anthropogenic GHG':'caused by humans', 'coal':'fossil fuel',
   'CO2 in 2019': 409.8,}
2  print(my_dict['CO2 in 2019'])
3  print(type(my_dict['CO2 in 2019']))
```

```
409.8
<class 'float'>
```

Lesson 5

Task 1:

main.py

```
time = 1.2
if time <2:
  print("not frozen long enough")
elif time ==2:
  print("frozen just long enough")
else:
  print("permafrost")
```

```
not frozen long enough
```

Task 2:

main.py

```
time = 2
if time <2:
  print("not frozen long enough")
elif time ==2:
  print("frozen just long enough")
else:
  print("permafrost")
```

```
frozen just long enough
```

main.py

```
time = 1000
if time <2:
  print("not frozen long enough")
elif time ==2:
  print("frozen just long enough")
else:
  print("permafrost")
```

```
permafrost
```

Task 3:

main.py

```
thawing_1 = 17
thawing_2 = 92
thawing_3 = 1500

if thawing_2 % 2==0:
  print("permafrost")
elif thawing_2 % 5==0:
  print("is thawing")
else:
  print(thawing_2)
```

```
permafrost
```

main.py

```
thawing_1 = 17
thawing_2 = 92
thawing_3 = 1500

if thawing_3 % 2==0:
  print("permafrost")
elif thawing_3 % 5==0:
  print("is thawing")
else:
  print(thawing_3)
```

```
permafrost
```

**The integer 1500 is both divisible by 2 and 5. However, since the condition is met by the if statement, then it will only print "permafrost" and not "is thawing".*

Task 4:

main.py

```
thawing_1 = 17
thawing_2 = 92
thawing_3 = 1500
if (thawing_2 % 2==0) and (thawing_2 % 5==0):
  print("permafrost is thawing")
elif thawing_2 % 2==0:
  print("permafrost")
elif thawing_2 % 5==0:
  print("is thawing")
else:
  print(thawing_2)
```

```
permafrost
```

Task 4 continued:

main.py

```
1  thawing_1 = 17
2  thawing_2 = 92
3  thawing_3 = 1500
4  if (thawing_3 % 2==0) and (thawing_3 % 5==0):
5    print("permafrost is thawing")
6  elif thawing_3 % 2==0:
7    print("permafrost")
8  elif thawing_3 % 5==0:
9    print("is thawing")
10 else:
11   print(thawing_3)
```

```
permafrost is thawing
```

As this example demonstrates, the combination of the two conditions using and, as well as parentheses ( ), is used to create the if statement. If a variable is not divisible by both 2 and 5, then it can be evaluated by the elif statements, and finally by the else statement.

Task 5:

main.py

```
1  permafrost = 2
2  while permafrost <= 1000:
3    print(permafrost)
4    permafrost = permafrost + 1
5
6
```

```
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

In order to print numbers incrementally, another statement that states that the variable is equal to the variable + 1 is added. As this example illustrates, the while loop is executed until the condition (permafrost is less than or equal to 1000) is false.

Task 6:

main.py

```
fTemp = [265, 255, 245, 235, 225, 215]

cTemp = []
for x in fTemp:
  cTemp.append((x - 32)*5/9)

print(cTemp)
```

Console Shell

```
[129.44444444444446, 123.88888888888889, 118.33333333333333, 112.7777
777, 107.22222222222223, 101.66666666666667]
```

Lesson 6

Task 1:

main.py

```
def permafrost_thawing(years_thawing):
  if (years_thawing % 2==0) and (years_thawing % 5==0):
    return "permafrost is thawing"
  elif years_thawing % 2:
    return "permafrost"
  elif years_thawing % 5:
    return "is thawing"
  else:
    return years_thawing

thawing_1 = permafrost_thawing(17)
thawing_2 = permafrost_thawing(92)
thawing_3 = permafrost_thawing(1500)
print(thawing_1, thawing_2)
```

```
permafrost is thawing
```

Task 2:

main.py ×

```
def permafrost_thawing(years_thawing):
  if (years_thawing % 2==0) and (years_thawing % 5==0):
    return "permafrost is thawing"
  elif years_thawing % 2:
    return "permafrost"
  elif years_thawing % 5:
    return "is thawing"
  else:
    return years_thawing

thawing_1 = permafrost_thawing(17)
thawing_2 = permafrost_thawing(92)
thawing_3 = permafrost_thawing(1500)
print(thawing_1 == thawing_2)
```

```
False
```

Task 3:

main.py

```
def fibonacci(number):
  if number == 0:
    return(0)
  elif number == 1:
    return(1)
  else:
    return fibonacci(number -1) + fibonacci(number -2)

for first_8 in range(0,8):
  print(fibonacci(first_8))
```

```
0
1
1
2
3
5
8
13
```

Lesson 7

Task 1:

main.py

```
class Turtle:
  def __init__ (self, species, location, conservation_status):
    self.species = species
    self.location = location
    self.conservation_status = conservation_status


pacific_green_sea_turtle = Turtle("Pacific Green Sea", "Raine Island", "Endangered")

nubian_flapshell_turtle = Turtle("Nubian flapshell", "West Africa", "Critically endangered")

print(nubian_flapshell_turtle.location)
```

Console | Shell

```
West Africa
```

Task 2:

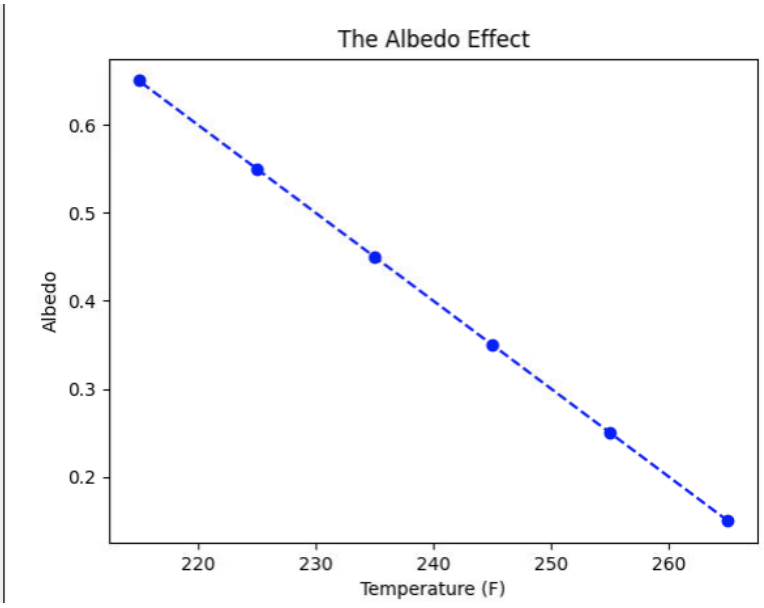
```python
class Greenhouse_gases:
  def __init__ (self, chemical_formula, source):
    self.chemical_formula = chemical_formula
    self.source = source

carbon_dioxide = Greenhouse_gases("CO2", "fossil fuels")
methane = Greenhouse_gases("CH4", "livestock")
nitrogen_dioxide = Greenhouse_gases("NO2", "industrial processes")

print(carbon_dioxide.source)
print(methane.source)
print(nitrogen_dioxide.source)
```

```
fossil fuels
livestock
industrial processes
```

Lesson 8

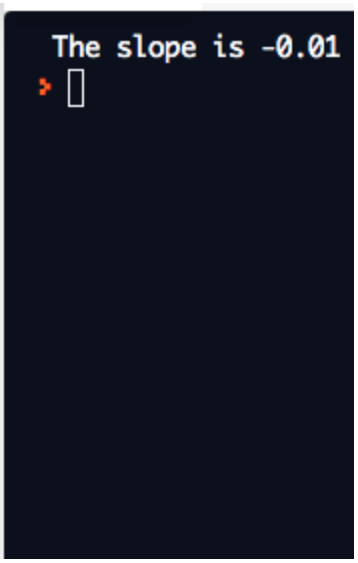
Task 1:

```python
import numpy as np
import matplotlib.pyplot as plt

temp = np.array([265, 255, 245, 235,
 225, 215])
albedo = np.array([0.15, 0.25, 0.35,
 0.45, 0.55, 0.65])

plt.plot(temp, albedo, marker='o',
linestyle = 'dashed', color =
"blue")

plt.xlabel("Temperature (F)")
plt.ylabel("Albedo")
plt.title("The Albedo Effect")
plt.show()
```



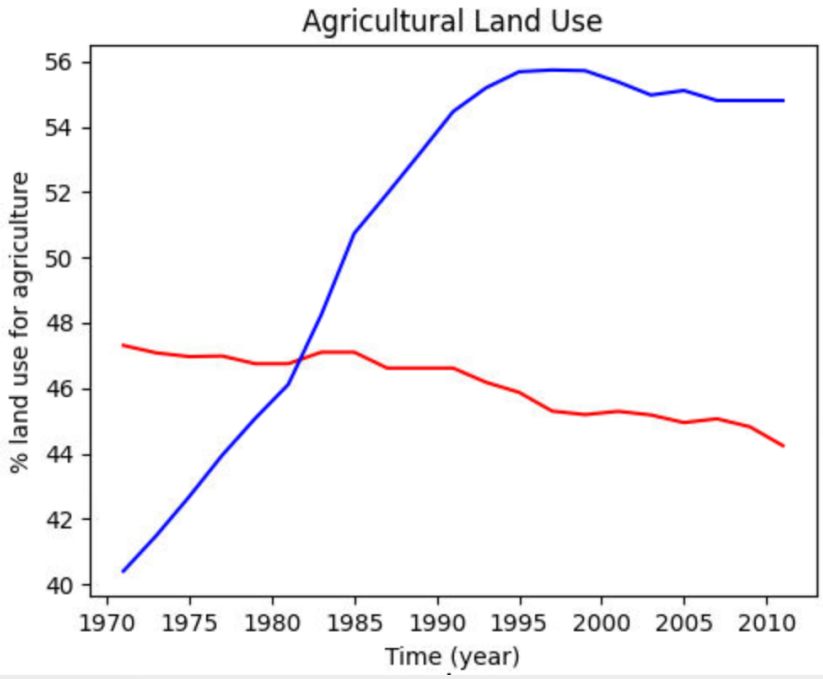
Task 2:

```
import numpy as np
temp = np.array([265, 255, 245, 235,
 225, 215])
albedo = np.array([0.15, 0.25, 0.35,
 0.45, 0.55, 0.65])

def slope(x1, x2, y1, y2):
  m = (y2-y1)/(x2-x1)
  return (f" The slope is {m}")

print(slope(temp[0], temp[5], albedo
[0],albedo[5]))
```



Python - Culminating Task (Answers)

```
import numpy as np
import matplotlib.pyplot as plt

years = np.array([1971, 1973, 1975, 1977, 1979, 1981, 1983, 1985, 1987, 1989, 1991, 1993, 1995, 1997,
1999, 2001, 2003, 2005, 2007, 2009, 2011])

land_area_list_US = np.array([47.3088648, 47.0795811, 46.9658127, 46.9847013, 46.7479932, 46.7479932,
47.1013084, 47.1013084, 46.6153362, 46.6153362, 46.6153362, 46.1786054, 45.8719112, 45.2982653,
45.189301, 45.2900702, 45.178849, 44.9451643, 45.0623443, 44.8170837, 44.2386282])

land_area_list_China = np.array([40.39624, 41.4858999, 42.682076, 43.9485527, 45.0765585, 46.1117887,
48.244135, 50.7386361, 51.9549437, 53.2032061, 54.4701089, 55.1991053, 55.6902039, 55.7435214,
55.7222181, 55.3795407, 54.9767062, 55.1151813, 54.8100187, 54.8084173, 54.8084173])

"""The Graph"""

plt.plot(years, land_area_list_US, color = "red")
plt.plot(years, land_area_list_China, color = "blue")
plt.xlabel("Time (year)")
plt.ylabel("% land use for agriculture")
plt.title("Agricultural Land Use")


"""Function to calculate avg for each country"""

def calculateAVG(num):
  for x in num:
    return sum(num) // len(num)


 """User Input & If statements"""

 x = input("Which country uses more % land for agriculture - US or China?:   ")
 y = calculateAVG(land_area_list_China)
 z = calculateAVG(land_area_list_US)

 if x in ["China", "china"]:
   print(f" You are correct! The % land used for agriculture in China is {y}
   percent compared to {z} percent in the US.")
   i = input("Would you like to see the data from the past 40 years?:  ")
   if i in ["Yes", "yes", "y", "Sure", "sure", "ok", "OK"]:
     plt.show()
   else:
     print("Thank you for your time!")


 elif x in ["US", "us", "United States", "united states", "USA", "usa"]:
   print(f" Actually, China uses more % land for agritculture: {y} compared to
   {z} percent in the United States.")
   i = input("Would you like to see the data from the past 40 years?:  ")
   if i in ["Yes", "yes", "y", "Sure", "sure", "ok", "OK"]:
     plt.show()
   else:
     print("Thank you for your time!")

 else:
   print("You didn't seem to select one of the two countries. Please try
   again.")
   x = input("Which country uses more % land for agriculture - US or China?")
```



Console Shell

```
Matplotlib created a temporary config/cache directory at /t
plotlib-z6yko0pz because the default path (/config/matplotlib) is
 not a writable directory; it is highly recommended to set the MP
LCONFIGDIR environment variable to a writable directory, in parti
cular to speed up the import of Matplotlib and to better support
multiprocessing.
Which country uses more % land for agriculture - US or China?:
China
 You are correct! The % land used for agriculture in China is 50.
0 percent compared to 46.0 percent in the US.
Would you like to see the data from the past 40 years?:  yes
```