Each of the following assignments is to be submitted to the autograder at gradescope.com. The assignment has a specific filename you must create or the grader will fail your work. You are allowed to submit as many times as you like to get full credit.

Lesson 2

1. Hello World

Description
Write a program that prints Hello World

File Name
hello.py

Score
There is a single test worth 2 points

Lesson 3

1. Coin Counter

Description
Write a program that asks the user to enter a number of quarters, dimes, nickels and pennies
and then outputs the monetary value of the coins in the format of dollars and remaining cents.
Your program should interact with the user exactly as it shows in the following example:

Please enter the number of coins:
# of quarters: 20
# of dimes: 4
# of nickels: 13
# of pennies: 17
The total is 6 dollars and 22 cents

File Name
counter.py

Score

There are five tests each worth 2 points

2. Coin Converter

Description
Write a program that asks the user to enter an amount of money in the format of dollars and remaining cents. The program should calculate and print the minimum number of coins (quarters, dimes, nickels and pennies) that are equivalent to the given amount.

Hint: In order to find the minimum number of coins, first find the maximum number of quarters that fit in the given amount of money, then find the maximum number of dimes that fit in the remaining amount, and so on.

File Name
coins.py

Score
There are five tests each worth 2 points

For example, an execution should look like this:
Please enter the amount of money to convert:

# of dollars: 2
# of cents: 37
The coins are 9 quarters, 1 dimes, 0 nickels and 2 pennies

3. BMI Metric

Description

Body mass index (BMI) is a number calculated from a person's weight and height. According to the Centers for Disease Control and Prevention, the BMI is a fairly reliable indicator of body fatness for most people. BMI does not measure body fat directly, but research has shown that BMI correlates to direct measures of body fat, such as underwater weighing and dual-energy X-ray absorptiometry. The formula for BMI is

$$\text{Weight}/\text{Height}^2$$

Where weight is in kilograms and height is in meters.

Write a program that prompts for metric weight and height and outputs the BMI.

For example, an execution could look like this:

```
Please enter weight in kilograms: 50
Please enter height in meters: 1.58
BMI is: 20.0288415318
```

File Name

bmimetric.py

Score

There are five tests each worth 2 points

4. BMI Imperial

Description

Body mass index (BMI) is a number calculated from a person's weight and height. According to the Centers for Disease Control and Prevention, the BMI is a fairly reliable indicator of body fatness for most people. BMI does not measure body fat directly, but research has shown that BMI correlates to direct measures of body fat, such as underwater weighing and dual-energy X-ray absorptiometry. The formula for BMI is

$$\text{Weight}/\text{Height}^2$$

Where weight is in kilograms and height is in meters.

Write a program that prompts for weight in pounds and height in inches, converts the values to metric, and then calculates the BMI.
Note: 1 pound is 0.453592 kilograms and 1 inch is 0.0254 meters.

For example, an execution could look like this:

```
Please enter weight in pounds: 135
Please enter height in inches: 71
BMI is: 18.8284697141
```

File Name

bmiimperial.py

Score

There are five tests each worth 2 points

Lesson 4

1. BMI Metric With Status Category

Description
Modify your earlier BMI Metric program to also display the CDC standard weight status categories:

The CDC standard weight status categories are:

BMI	Weight Status
Below 18.5	Underweight
18.5-24.9	Normal
25.0-29.9	Overweight
30.0 and above	Obese

For example, an execution could look like this:
Please enter weight in kilograms: 50
Please enter height in meters: 1.58
BMI is: 20.0288415318, Status is Normal

File Name
bmimetric2.py

Score
There are five tests each worth 2 points

2. Cash Register

Description

Write a program that computes how much a customer has to pay after purchasing two items.

The price is calculated according to the following rules:
• Buy one get one half off promotion: the lower price item is half price.
• If the customer is club card member, additional 10% off.
• Tax is added.

Inputs to the program include:
• Two items' prices
• Have club card or not (User enters 'Y' or 'y' for "yes"; 'N' or 'n' for "no")
• Tax rate (User enters the percentage as a number; for example, they enter 8.25 if the tax rate is 8.25%)

Program displays:
• Base price - the price before the discounts and taxes
• Price after discounts - the price after the buy one get one half off promotion and the member's discount, if applicable
• Total price – the amount of money the customer has to pay (after tax) printed with the precision of 2 decimal digits.

Hint: In order to print a number in a specific precision, you can use the round function passing 2 arguments to it. Use help(round) to get a brief explanation of this function, and try playing with it, to better understand what it does.
To format to two decimal places you can use the string format function with the format of 2.2f.

For example, an execution could look like this:

```
Enter price of the first item: 10
Enter price of the second item: 20
Does customer have a club card? (Y/N): y
Enter tax rate, e.g. 5.5 for 5.5% tax: 8.25
Base price = 30.00
Price after discounts = 22.50
Total price = 24.36
```

File Name

cashregister.py

Score

There are five tests each worth 2 points

3. Call Cost Calculator

Description

Write a program that computes the cost of a long-distance call. The cost of the call is determined according to the following rate schedule:
• Any call started between 8:00 A.M. and 6:00 P.M., Monday through Friday, is billed at a rate of $0.40 per minute.
• Any call starting before 8:00 A.M. or after 6:00 P.M., Monday through Friday, is charged at a rate of $0.25 per minute.
• Any call started on a Saturday or Sunday is charged at a rate of $0.15 per minute.
The input will consist of the day of the week, the time the call started, and the length of the call in minutes.
The output will be the cost of the call.

Notes:
1. The time is to be input as 4 digit number, representing the time in 24-hour notation, so the time 1:30 P.M. is input as 1330
2. The day of the week will be read as one of the following three character string: ‘Mon’, ‘Tue’, ‘Wed’, ‘Thr’, ‘Fri’, ‘Sat’ or ‘Sun’
3. The number of minutes will be input as a positive integer.

For example, an execution could look like this:

Enter the day the call started at: Fri
Enter the time the call started at (hhmm): 2350
Enter the duration of the call (in minutes): 22
This call will cost $5.50

File Name
callcoster.py

Score
There are five tests each worth 2 points

Lesson 5

1. Even Numbers

Description

Write a program that reads a positive integer n, and prints the first n even numbers.

For example, one execution would look like this:
Please enter a positive integer: 3
2
4
6

File Name
evennumbers.py

Score
There are three tests each worth 2 points

2. Fibonacci

Description
Fibonacci number is a series of numbers in which each number is the sum of the two preceding numbers. The first two numbers in the series are the number 1. Write a program to print first n Fibonacci Numbers
For example, one execution would look like this:
Please enter a positive integer greater than 1: 4
1
1
2
3

File Name
fibonacci.py

Score
There are five tests each worth 2 points

Lesson 6

1. String Splitter

Description

Read from the user a string containing
odd number of characters. Your program should:
a) Print the middle character.
b) Print the string up to but not including the middle character (i.e., the first half of the string).
c) Print the string from the middle character to the end (not including the middle character).

Sample output:
Enter an odd length string: Fortune favors the bold
Middle character: o
First half: Fortune fav
Second half: rs the bold

File Name
stringsplitter.py

Score
There are five tests each worth 2 points

2. Character Type

Description

Write a program that reads a character (string of length 1) from the user, and classifies it to one of the following: lower case letter, upper case letter, digit, or non-alphanumeric character

Sample output (4 different executions):
Enter a character: j
j is a lower case letter.
Enter a character: 7
7 is a digit.
Enter a character: ^
^ is a non-alphanumeric character.
Enter a character: C
C is an uppercase letter.

File Name
chartype.py

Score
There are five tests each worth 2 points

Lesson 7

1. Leap Year Function

Description

Write a method for determining if a year is a leap year in the Gregorian calendar. The system is to check if it is divisible by 4 but not by 100 unless it is also divisible by 400.

For example, 1896, 1904, and 2000 were leap years but 1900 was not.
Write a function that takes in a year as input and returns True if the year is a leap year, return False otherwise.

Note: background on leap year https://en.wikipedia.org/wiki/Leap_year

The name of the method should be isleapyear and the method should take one parameter which is the year to test. Here is an example call to the function

```
mybirthyear = 2000
If isleapyear(mybirthyear):
   print("Year {0} was a leap year".format(mybirthyear))
```

File Name
isleapyear.py

Score
There are five tests each worth 2 points

Note: You do not need any other code in the main method or any print statements. ONLY the *isleapyear* method is required. Otherwise, the autograder will fail and be unable to grade your code. (I.e., do not include the above example in your code.) The above example should be used to test your code but deleted or commented out upon submission.

2. First Word Function

Description
Write a function that is given a phrase and returns the first word in that phrase.
For example, given ‘the quick brown fox’, your function should return ‘the’.

Here is an example call to the function
print(firstword(“the quick brown fox”)):

File Name
firstword.py

Score
There are three tests each worth 2 points

Note: You do not need any other code in the main method or any print statements. ONLY the *firstword* method is required. Otherwise, the autograder will fail and be unable to grade your code. (I.e., do not include the above example in your code.) The above example should be used to test your code but deleted or commented out upon submission.

3. Remaining Words Function

Description
Write a function that is given a phrase and returns the phrase we get if we take out the first word from the input phrase.
For example, given ‘the quick brown fox’, your function should return ‘quick brown fox’

Here is an example call to the function
print(remainingwords(“the quick brown fox”)):

File Name
remaining.py

Score
There are three tests each worth 2 points

Note: You do not need any other code in the main method or any print statements. ONLY the *remainingwords* method is required. Otherwise, the autograder will fail and be unable to grade your code. (I.e., do not include the above example in your code.) The above example should be used to test your code but deleted or commented out upon submission.

Lesson 8 - Lists

1. Max In List Function

Description

Implement function max_val(lst), which returns the maximum value of the elements in the list.
For example, given a list lst: [-19, -3, 20, -1, 5, -25], the function
should return 20.

The name of the method should be max_val and the method should take one parameter which is the list of values to test. Here is an example call to the function

print(max_val([-19, -3, 20, -1, 5, -25]))

File Name
maxinlst.py

Score
There are three tests each worth 2 points

Note: You do not need any other code in the main method or any print statements. ONLY the *max_val* method is required. Otherwise, the autograder will fail and be unable to grade your code. (I.e., do not include the above example in your code.) The above example should be used to test your code but deleted or commented out upon submission.

2. Max Absolute In List Function

Description

Implement function max_abs_val(lst), which returns the maximum absolute value of the elements in the list.
For example, given a list lst: [-19, -3, 20, -1, 0, -25], the function should return 25.

The name of the method should be max_abs_val and the method should take one parameter which is the list of values to test. Here is an example call to the function

print(max_abs_val([-19, -3, 20, -1, 0, -25]))

File Name
maxabsinlst.py

Score
There are three tests each worth 2 points

Note: You do not need any other code in the main method or any print statements. ONLY the *max_abs_val* method is required. Otherwise, the autograder will fail and be unable to grade your code. (I.e., do not include the above example in your code.) The above example should be used to test your code but deleted or commented out upon submission.

3. Average of List Function

Description

Implement function avg_val(lst), which returns the average value of the elements in the list.
For example, given a list lst: [19, 2, 20, 1, 0, 18], the function should return 10.

The name of the method should be avg_val and the method should take one parameter which is the list of values to test. Here is an example call to the function

print(avg_val([19, 2, 20, 1, 0, 18]))

File Name
avgoflst.py

Score
There are three tests each worth 2 points

Note: You do not need any other code in the main method or any print statements. ONLY the *avg_val* method is required. Otherwise, the autograder will fail and be unable to grade your code. (I.e., do not include the above example in your code.) The above example should be used to test your code but deleted or commented out upon submission.