Unit 15

Functions & Modules

Asg 15.3 (Coding)

In [44]:
```
# set up notebook to display multiple output in one cell

from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
import random
import numpy as np
import math

print('The notebook is set up to display multiple output in one cell.')
```

```
The notebook is set up to display multiple output in one cell.
```

Problem #1: Complete the following task:

Write a function print_triangular_numbers(n) that prints out the first n triangular numbers. A call to print_triangular_numbers(5) would produce the following output: ! [trianglular%20numbers.JPG](attachment:trianglular%20numbers.JPG) (Hint: use a web search to find out what a triangular number is.)

In [5]:
```
def print_triangular_numbers(n):
    return n * (n+1)/2

print(int(print_triangular_numbers(5)))
```

```
15
```

Problem #2: Complete the following task:
Write a function that will return the number of digits in an integer. Test your code on the following integers: 5000, 27, 861, and 12500

In [25]:
```
def countDigits(n):
    global counter
    while n != 0:
        n //= 10
        counter += 1
    return counter


counter = 0
print(countDigits(5000))
```

```
counter = 0
print(countDigits(27))
counter = 0
print(countDigits(861))
counter = 0
print(countDigits(12500))
```

```
4
2
3
5
```

Problem #3: Use the built-in random module and a for loop to generate 40 random numbers between -50 and 50 then display a list of only the negative numbers rounded to two decimal places. The program should also count the number of negative numbers and display the count. Your output should look similar to the desired output that is found below (numbers and counts may vary):

DESIRED OUTPUT ![image.png](attachment:image.png)

In [33]:
```
ranlist = []
for x in range(40):
    num = random.uniform(-50,50)
    num = round(num,2)
    ranlist.append(num)
print(ranlist)
```

```
[-33.29, -21.39, -19.44, 9.62, -7.69, -46.86, 31.49, -33.83, 48.2, -8.34, 48.22, 26.21,
22.0, 24.84, -46.99, -47.56, 25.86, -15.33, 44.06, -3.98, -9.52, -24.88, -3.32, -43.92,
-12.82, 2.94, 12.43, 43.96, -41.88, -31.92, 25.97, -41.46, -23.55, -34.38, 15.94, 21.3
8, 23.58, 12.07, -24.99, 30.73]
```

Problem #4: Define a function called common_divisors that takes two positive integers m and n as its arguments and returns the list of all common divisors (including 1) of the two integers, unless 1 is the only common divisor. If 1 is the only common divisor, the function should print a statement indicating the two numbers are relatively prime. Otherwise, the function prints the number of common divisors and the list of common divisors.

Do Parts (a) and (b) below to test your code.

a. Call the function common_divisors by passing 5 and 38.

b. Call the function common_divisors by passing 72 and 48.

Your output for Parts (a) and (b) should look like this:

DESIRED OUTPUT ![common%20div.JPG](attachment:common%20div.JPG)

In [20]:
```
def common_divisors(m, n):
    divlist=[]
    for i in range(1,m+1):
        if m % i == 0 and n % i ==0:
            divlist.append(i)
    if len(divlist) == 1:
        return f"{m} and {n} are relatively prime"
    else:
        return f"{m} and {n} have {len(divlist)} common divisors, including 1\n{divlist}
print(common_divisors(5,38))
print("\n")
print(common_divisors(72,48))
```

```
5 and 38 are relatively prime


72 and 48 have 8 common divisors, including 1
[1, 2, 3, 4, 6, 8, 12, 24]
```

Problem #5: Complete the following steps: 1. Define a function called polys that takes 4 arguments x, a, b, and c and returns the value of a polynomial of the form $\mathbf{ax^3 + bx^2 + cx}$. 2. Next, prompt the user to enter the coefficients of the polynomial (see testing input below). 3. Initialize x and assign initial values to variables named max_y and min_y by calling the function polysand passing the initial value of x and the user's input. 4. Calculate the maximum and minimum values and their corresponding x-values over the interval **[0, 20]**. To accomplish this, create a while loop or a for loop to iterate over the values x = 0.1, 0.2, ..., 19.9, 20 and evaluate the function at each of these x-values (by calling polys) to determine the maximum and minimum function values.

5. Create two formatted print statements that round the values to 2 decimal places and display the output shown below.

Sample input and output are shown here:

Your input should look like this:

```
Enter the coefficient of x^3: 1
Enter the coefficient of x^2: -17
Enter the coefficient of x: 72
```

Your output should look like this:

```
The maximum value is 2581.23 when x is 19.90.
The minimum value is -2.12 when x is 8.50.
```

In [49]:
```
def polys(x,a,b,c):
    return (a*math.pow(x,3)) + (b*math.pow(x,2)) + c*x
```

```
a = int(input("Enter the coefficient of x^3: "))
b = int(input("Enter the coefficient of x^2: "))
c = int(input("Enter the coefficient of x: "))

min = 0
minIndex = 0
max = 0
maxIndex = 0
for i in np.arange(0.1, 20.1, 0.1):
    num = polys(i,a,b,c)
    if(num < min):
        min = num
        minIndex = i
    if(num > max):
        max = num
        maxIndex = i

print(f"The maximum value is {max:.2f} when x is {maxIndex:.2f}.")
print(f"The minimum value is {min:.2f} when x is {minIndex:.2f}.")
```

```
Enter the coefficient of x^3: 1
Enter the coefficient of x^2: -17
Enter the coefficient of x: 72
The maximum value is 2640.00 when x is 20.00.
The minimum value is -2.12 when x is 8.50.
```

Problem #6: Complete the following steps: a. Write a function that converts degrees Celsius (C) to degrees Fahrenheit (F). The formula to convert between the two temperature scales is F = 1.8C + 32. Convert 100 degrees C to degrees F using your function. b. Write a function that converts Kelvin temperature (K) to degrees Celsius (C). The formula to convert between the two temperature scales is C = K − 273.15. c. Use the functions in Parts (a) and (b) to convert the temperature at Standard Temperature and Pressure (STP) of 273.15 K into degrees F. d. Use the functions in Parts (a) and (b) to convert the temperature at absolute zero, 0 K, into degrees Celsius and degrees Fahrenheit.

Use formatted print statements to produce meaningful and comprehensive output.

In [2]:
```
def CelsiusToFarenheit(c):
    return 1.8*c + 32
def KelvinToCelsius(k):
    return k - 273.15

print(CelsiusToFarenheit(KelvinToCelsius(273.15)))
print(round(CelsiusToFarenheit(KelvinToCelsius(0)),2))
```

```
32.0
-459.67
```

Problem #7: Complete the following steps:
One way to calculate how much an investment will be worth is to use the Future Value formula: ![FV.JPG](attachment:FV.JPG) (a) Write a function called future_value() which accepts an initial investment I_0 and a number of years n and calculates the future value FV. Include r

= 0. 5 as the default yearly rate of return. (b) Use your future_value() function to calculate the future value of an initial investment of 2000 dollars over 30 years with the default yearly rate of return (c) Use your future_value() function to calculate the future value of the same initial investment of 2000 dollars over 30 years, but a rate of return of 8% (0.08). (d) Use your future_value() function to determine when 2000 dollars is invested over 30 years, how much more do you make if the rate of return is 10% (0.10) instead of 5% (0.05).

Use formatted print statements to produce meaningful and comprehensive output.

In [4]:
```
def future_value(init_investment, n, r = 0.05):
    return (init_investment) * ((1 + r)**n)

print(f"Over 30 years with a 5% rate of return, the future value is {future_value(2000,
print(f"Over 30 years with a 8% rate of return, the future value is {future_value(2000,
print(f"Over 30 years with a 10% rate of return, the future value is {future_value(2000
```

```
Over 30 years with a 5% rate of return, the future value is 8643.88
Over 30 years with a 8% rate of return, the future value is 20125.31
Over 30 years with a 10% rate of return, the future value is 34898.80
```

Problem #8: Complete the following steps:
Write a function that reverses its string argument. Test your code on the string 'Python'.

In [5]:
```
def reverseString(string):
    return string[::-1]

reverseString('Python')
```

Out[5]:
```
'nohtyP'
```

Problem #9: Complete the following steps:
Write a function that will find the average of a variable number of test scores and then print out that average (to 2 decimal places) along with the letter grade that corresponds to that average (see the grade scale below). ![grades.JPG](attachment:grades.JPG) Test your code using the following test scores:

scores_term1 = (84, 76, 98, 92, 90)
scores_term2 = (100, 84, 96, 98, 88, 94, 82, 96)

In [8]:
```
def averageTestScore(*scores):
    total = 0
    for score in scores:
        total += score
    return total/len(scores)
def getLetterGrade(percent):
    if percent >= 93 and percent <= 100:
        return 'A'
    elif percent >= 85 and percent < 93:
```

```
        return 'B'
    elif percent >= 77 and percent < 85:
        return 'C'
    elif percent >= 70 and percent < 77:
        return 'D'
    else:
        return 'F'
scores_term1 = (84, 76, 98, 92, 90)
scores_term2 = (100, 84, 96, 98, 88, 94, 82, 96)
print(f"Your average test score in term 1 was {averageTestScore(*scores_term1)}% with a
print(f"Your average test score in term 2 was {averageTestScore(*scores_term2)}% with a
```

```
Your average test score in term 1 was 88.0% with a letter grade of B
Your average test score in term 2 was 92.25% with a letter grade of B
```

Problem #10: Complete the following steps:
Write a function that takes a variable number of keyword arguments and then prints out those keyword arguments along with all of the key value pairs using the format key -> value for each key-value pair.

Test your code using 3 cases:

(Teacher1 = 'Mr. Baker', Department = 'Science', Undergrad = 'UW-Madison')
(Teacher2 = 'Ms. Jones', Department = 'Business', Undergrad = 'Marquette', Grad = 'UWM')
(Teacher3 = 'Mrs. Smith', Department = 'Math', Undergrad = 'MIT', Grad = 'Harvard', PHD = 'Yale', Awards = 'Nobel Peace Prize')

In [25]:
```
def listOfTeachers(**teachers):
    for key, value in teachers.items():
        print(f"{key : <20}    {value}")
    print("--------------------------------")

listOfTeachers(Teacher1 = 'Mr. Baker', Department = 'Science', Undergrad = 'UW-Madison'
listOfTeachers(Teacher2 = 'Ms. Jones', Department = 'Business', Undergrad = 'Marquette'
listOfTeachers(Teacher3 = 'Mrs. Smith', Department = 'Math', Undergrad = 'MIT', Grad =
```

```
Teacher1             Mr. Baker
Department           Science
Undergrad            UW-Madison
--------------------------------
Teacher2             Ms. Jones
Department           Business
Undergrad            Marquette
Grad                 UWM
--------------------------------
Teacher3             Mrs. Smith
Department           Math
Undergrad            MIT
Grad                 Harvard
PHD                  Yale
Awards               Nobel Peace Prize
--------------------------------
```