**Problem 1: Visual Studio**

Download and install Microsoft Visual Studio. You are entitled to a free copy using DePaul’s MSDNAA license. “The MSDN Academic Alliance (MSDNAA) is a special program to provide academic institutions, faculty, and students with leading-edge tools for students.” You can find more information about the MSDNAA on the CDM website (www.cdm.depaul.edu/Current%20Students/Pages/MSDNAA.aspx).

Download and install the latest version of Python (www.python.org/downloads).

Download and install the latest version of Anaconda (www.anaconda.com).

Watch the first three (of six) tutorials from Microsoft on programming in python with Visual Studio:

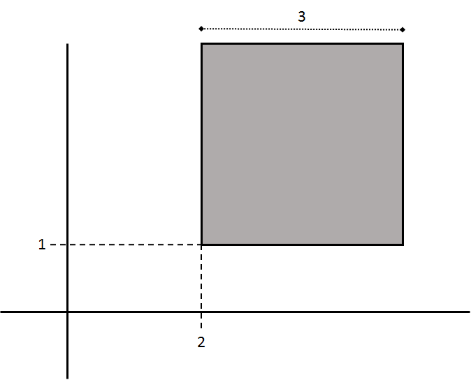
* 1/6 – [www.youtube.com/watch?v=\_okUV47eM5c](http://www.youtube.com/watch?v=_okUV47eM5c)
* 2/6 – [www.youtube.com/watch?v=KHPoVpL7zHg](http://www.youtube.com/watch?v=KHPoVpL7zHg)
* 3/6 – [www.youtube.com/watch?v=uZGZNEyyeKs](http://www.youtube.com/watch?v=uZGZNEyyeKs)

Write a code that:

1. Asks the user for a first name.
2. Asks the user for a last name.
3. Asks the user for an IQ.
4. Responds to the user using all three inputs above to output a message predicting their grade in DSC 430. If their IQ is less than 90, they will likely get a D. If their IQ is between 90 and 110, they will likely earn a C. If their IQ is between 110 and 130, they will likely get a B. If their IQ is greater than 130, call them a liar.

Submission: Submit a selfie of yourself next to your computer. The computer must display Visual Studio demonstrating the above code. Extra credit will be given if the selfie includes pets or t-shirts from my favorite sports teams. Point may be deducted if the selfie include t-shirts from any other sports teams.

Grading: There is no partial credit for this problem.

**Problem 2: Squares**

A square can be defined by 3 ints: x, the horizontal position of the lower left corner; y, the vertical position of the lower left corner; and s, the height and width of the square. These three ints can be stored in a list. For example, [2,1,3] would represent the square in the figure.

1. Write a function that accepts a list and returns true if it contain exactly 3 ints. Otherwise it should return false.
2. Write a function that accepts a list and returns true if it is a valid representation of a square. The x and y coordinate may be positive or negative ints, but the height/width must be a positive int. If the representation is not valid return false.
3. Write a function that accepts a list representing a square and returns the perimeter. If the input is invalid, return -1.
4. Write a function that accepts a list representing a square and returns the area. If the input is invalid, return -1.
5. Write a function that accepts two lists, each representations of two different squares. Compute the area of the overlap between the two squares. If the representations are not valid return -1
6. Test your overlap function with the following code:

sq1 = [1,0,2]

sq2 = [2,1,5]  
sq3 = [4,3,1]  
sq4 = [1,5,3]

sq5 = [6,4,-3]  
sq6 = [6,4,'three']  
  
print("The overlap of " + str(sq1) + " and " + str(sq2) + " is " + str(overlap(sq1,sq2)))

print("The overlap of " + str(sq2) + " and " + str(sq3) + " is " + str(overlap(sq2,sq3)))

print("The overlap of " + str(sq2) + " and " + str(sq4) + " is " + str(overlap(sq2,sq4)))  
print("The overlap of " + str(sq1) + " and " + str(sq4) + " is " + str(overlap(sq1,sq4)))  
print("The overlap of " + str(sq1) + " and " + str(sq5) + " is " + str(overlap(sq1,sq5)))  
print("The overlap of " + str(sq1) + " and " + str(sq6) + " is " + str(overlap(sq1,sq5)))

Submission: Submit a single .py file containing all the code to the D2L. Do not zip or archive the file. Your code must include comments at the top including your name, date and the honor statement, “I have not given or received any unauthorized assistance on this assignment.” Each function must include a docstring and be commented appropriately.

Grading: There is no partial credit for this problem. The code must compile, run without errors and satisfy all requirement to receive full points.

**Problem 3: Happy Primes**

1. A prime number (or a prime) is a natural number greater than 1 that cannot be formed by multiplying two smaller natural numbers. Write a function that returns true if an int is prime. If the int is not prime or if it is an invalid input return false. Write your code efficiently.
2. Happy has many definitions (even in mathematics). For our purposes, a happy number is a number defined by the following process: Starting with any positive integer, replace the number by the sum of the squares of its digits, and repeat the process until the number either equals 1 (where it will stay), or it loops endlessly in a cycle that does not include 1. Those numbers for which this process ends in 1 are happy numbers, while those that do not end in 1 are unhappy numbers (or sad numbers). For example, 19 is happy, as the associated sequence is:  
   * 12 + 92 = 82
   * 82 + 22 = 68
   * 62 + 82 = 100
   * 12 + 02 + 02 = 1

Write a function that returns true if an int is happy.

1. Write a function that returns true if an int is a happy prime, that is to say the int is both happy and a prime.
2. Write a function that computes and prints the first 100 happy primes in a comma separated list.

7, 13, 19, …

1. Write a function that computes and prints the first 100 sad primes in a comma separated list.

2, 3, 5, 11, …

Note: Please consider the efficiency of your code.

Submission: Submit a single .py file containing all the code to the D2L. Do not zip or archive the file. Your code must include comments at the top including your name, date and the honor statement, “I have not given or received any unauthorized assistance on this assignment.” Each function must include a docstring and be commented appropriately.

Grading: There is no partial credit for this problem. The code must compile, run without errors and satisfy all requirement to receive full points.

**Problem 4: Toads and Tacos**

Consider a game that presents the user with a starting word and asks the user to make a single change at each step until it matches a goal word. At each step the user has three options: a) insert a character, b) remove a character, or c) replace a character. For example:

Starting word = toads  
 Goal word = tacos

Step 0: toads #starting word  
Step 1: tads # remove index 1  
Step 2: tacs # replace index 2 with ‘c’  
Step 3: tacos # insert ‘o’ at index 3

Implement this game. Your solution should…

1. Begin by asking the user for a name.
2. Greet the user by name and print instructions.
3. Present the starting and goal words (using something else besides “Toads” and “Tacos”).
4. Offer the user the choice to insert, remove or replace a character.
5. Continue until the goal word is reached.
6. Congratulate the user by name and output the number of steps it took to achieve the goal.

Submission: Submit a single .py file containing all the code to the D2L. **For this problem include in the .py file a sample run of the program as a comment at the end of the file.** Do not zip or archive the file. Your code must include comments at the top including your name, date and the honor statement, “I have not given or received any unauthorized assistance on this assignment.” Each function must include a docstring and be commented appropriately.

Grading: There is no partial credit for this problem. The code must compile, run without errors and satisfy all requirement to receive full points.