Exercise 1a --- Internet Delicatessen
Sam and Ella's Delicatessen wants a program to take orders from the internet. The program will accept item the user wants, its price, and will ask if the user wants overnight shipping. Regular shipping for items under $10 is $2.00; for items $10 or more shipping is $3.00. For overnight delivery add $5.00.

The item: Tuna Salad
Enter the price: 450
Overnight delivery (0==no, 1==yes): 1
Invoice:
Tuna Salad 4.50
shipping 7.00
total 11.50

Exercise 1b --- Tire Pressure
The front tires of a car should both have the same pressure. Also, the rear tires of a car should both have the same pressure (but not necessarily the same pressure as the front tires.) Write a program that accepts the pressure of the four tires and writes a message that says if the inflation is OK or not.

Right front pressure: 38
Left front pressure: 38
Right rear pressure: 42
Left rear pressure: 42
Inflation is OK
Exercise 2a --- More Tire Pressure
Its not enough that the pressures are the same in the tires, but the pressures must also be within range. Modify the program in exercise 1b so that it also checks that each tire has a pressure between 35 and 45. If there have been any warnings, write out a final error message.

Right front pressure: 32
Warning: pressure is out of range
Left front pressure: 32
Warning: pressure is out of range
Right rear pressure: 42
Left rear pressure: 42
Inflation is BAD

Exercise 2b --- The Pressure is Building
Tires don't have to have exactly the same pressure. Modify the program for exercise 2a so that the front tires can be within 3 psi of each other, and the rear tires can be within 3 psi of each other.

Right front pressure: 35
Left front pressure: 37
Right rear pressure: 41
Left rear pressure: 44
Inflation is OK
**Exercise 3a --- Order Checker**
Bob's Discount Bolts charges the following prices: 5 cents per bolt, 3 cents per nut, 1 cent per washer. Write a program that asks the user for the number of bolts, nuts, and washers in their purchase and then calculates and prints out the total. As an added feature, the program checks the order. It is usually a mistake if there are more bolts than nuts. In this case the program writes out "Check the Order." Otherwise the program writes out "Order is OK." In either case the total price is written out.

Number of bolts: 12  
Number of nuts: 8  
Number of washers: 24  
Check the Order  
Total cost: 108

**Exercise 3b --- Last Chance Gas**
Al's Last Chance Gas station sits on route 190 on the edge of Death Valley. There is no other gas station for 200 miles. You are to write a program to help drivers decide if they need gas. The program asks for:

The capacity of the gas tank, in gallons  
The indication of the gas gauge in percent (full= 100, three quarters full = 75, and so on)  
The miles per gallon of the car.  
The program then writes out "Get Gas" or "Safe to Proceed" depending on if the car can cross the 200 miles with the gas remaining in the tank.  
Tank capacity: 12  
Gage reading: 50  
Miles per gallon: 30  
Get Gas!
Exercise 4a --- Wedge of Stars
Write a program that will write out a wedge of stars. The user will enter the initial number of stars, and the program will write out lines of stars where each line has one few star than the previous line:

Initial number of stars: 7
******
*****
****
***
**
*

Exercise 4b --- Start to End
Write a program that asks the user for a starting value and an ending value and then writes all the integers (inclusive) between those two values.

Start: 5
End: 9
5
6
7
8
9
**Exercise 5 --- Factorial**
Write a program that receives a number and calculates and returns its factorial. Write it first using a while loop. Then write it using a for loop.

**Exercise 6 --- List of divisors**
Write a program that receives a number and calculates and returns a list of its divisors. Write it first using a while loop. Then write it using a for loop.

Hint: Before the loop, create a an empty list to store the divisors e.g., `divisors = []`. Inside the loop, every time you find a divisor, use the method `append` to add the found divisor to the list e.g., `divisors.append(i)`, where `i` is the divisor number being added to the list.

**Exercise 7 --- Fibonacci**
Write a program that receives a number and calculates its `fibonacci`. The `fibonacci` number of `n` is the `fibonacci` number of `n-1` added to the `fibonacci` number of `n-2`. The first two numbers in the `Fibonacci` sequence are 0 and 1. So the function `fib(0)` should return 0; `fib(1)` should return 1; `fib(2)` is 0 + 1, so returns 1; `fib(3)` is 1 + 1, so returns 2. `fib(4)` is 2+1, so returns 3; `fib(5)` is 3+2, so returns 5; etc, etc.

Hint: Before the loop, create a an empty list to store the `fibonacci` sequence e.g., `fibseq = []`. Inside the loop, every time you calculate a new `fibonacci` number, added to the list. To calculate the number, you will need to look in the list and access the past two numbers e.g., `fibseq[n] = fibseq[n-1] + fibseq[n-2].`