1. Write a code segment that creates a window that is 300 pixels wide and 400 pixels tall and then draws a rectangle with one corner at location (50, 30) and its diagonally opposite at location (150, 200). The user should then be prompted (from the Python shell) to strike any key to close the window. Assume the graphics package has been imported with the directive from graphics import *

```python
win = GraphWin("One",300,400)
rect = Rectangle(Point(50,30),Point(150,200))
rect.draw(win)
input("Strike any key to close the window")
win.close()
```

2. Append to the following segment of code additional lines that:
   - Will prompt the user (in the shell) for the number of mouse clicks to expect from the user. Assume it will be an integer greater than 1.
   - Upon receiving the first click from the user the program will draw a point at the location where the user clicked.
   - Upon receiving additional clicks from the user the program will draw a line from the location where the user clicked previously clicked to the latest location where the user clicked. This continues until the number of mouse clicks specified in the user input have been received. When the drawing is done the user should see a collection of connected line segments in the window.
   - Upon receiving one additional mouse click the window will close.

```python
from graphics import *
from math import *
def main():
    win = GraphWin("CSCI220",400,200)
    n = eval(input("n: "))
    prev = None
    for i in range(n):
        c = win.getMouse()
        if prev == None:
            prev = Point(c.getX(),c.getY())
            prev.draw(win)
        else:
            next = Point(c.getX(),c.getY())
            next.draw(win)
            Line(prev,next).draw(win)
            prev = next
    win.getMouse()
```
3. Write a segment of code that opens the file **Test2-3.in** and determines the average number of words per line in the text.

```python
infile = open("Test2-3.in","r")
sum = 0
cnt = 0
for line in infile:
    words = line.split(" ")
    sum = sum + len(words)
    cnt = cnt + 1
infile.close()
print(sum/cnt)
```

4. Write a segment of code that opens the file **Test2-4.in** and copies every odd numbered line to **Test2-4.out**.

```python
infile = open("Test2-4.in","r")
outfile = open("Test2-4.out","w")
cnt = 0
for line in infile:
    if cnt+1 % 2 == 0:
        print(line,end='',file=outfile)
    cnt = cnt + 1
infile.close()
outfile.close()
```

5. Write a conditional statement that prints the appropriate message based on user input.
Assume the user input is stored in a variable `num` and is an integer.

<table>
<thead>
<tr>
<th>User Input</th>
<th>Appropriate Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 or less</td>
<td>“Error”</td>
</tr>
<tr>
<td>1..10</td>
<td>“Good”</td>
</tr>
<tr>
<td>11..15</td>
<td>“OK”</td>
</tr>
<tr>
<td>16 or more</td>
<td>“Awful”</td>
</tr>
</tbody>
</table>

```python
if num <= 0:
    print("Error")
elif num >= 1 and num <= 10:
    print("Good")
elif num >= 11 and num <= 15:
    print("OK")
elif num >= 16:
    print("Awful")
```
6. Write a function `changeName(name)` that takes a string value `name` as an argument, where name is in the form

   “Last, First Middle”

for example “Anderson, Paul Edward”. Note the use of the comma. The function should return a string that has the name in the form.

   “First Middle Last”

for example “Paul Edward Anderson.”

def changeName(name):
    fields = name.split(“,”)
    print(fields[1][1:],fields[0])

7. Write a function `changeName2(nameList)` that takes a list `nameList` as an argument that contains a person’s name, where name is in the form

   [Last, First, Middle]

for example ['Anderson', 'Paul', 'Edward']. The function should not return a value, but instead should modify the list so the name is in the form

   [First, Middle, Last]

for example ['Paul', 'Edward', 'Anderson'].

def changeName2(nameList):
    last = nameList[0]
    first = nameList[1]
    middle = nameList[2]
    nameList[0] = first
    nameList[1] = middle
    nameList[2] = last
8. Write a function `findValidValue()` that returns a valid value entered by the user. The function should ask the user to enter a number until the user enters a value that is any of the following.
   a. Inclusively within the range of 5 and 17.
   b. Exactly equivalent to 100.
   c. Exclusively within the range of 20 and 25.

```python
def findValidValue():
    while True:
        value = eval(input("Enter a number: "))
        if value >= 5 and value <= 17:
            return value
        if value == 100:
            return value
        if value >= 20 and value <= 25:
            return value
```

8. Evaluate the following Boolean expressions:

<table>
<thead>
<tr>
<th>Expression</th>
<th>Truth Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3 &lt; 7</td>
<td>T</td>
</tr>
<tr>
<td>3 &gt; 20 and 30 &gt;= 20</td>
<td>F</td>
</tr>
<tr>
<td>20 == 20</td>
<td>T</td>
</tr>
<tr>
<td>7 == 8 and not(20 &lt; 18)</td>
<td>F</td>
</tr>
<tr>
<td>3 &gt; 4 or (5 &lt;= -3 and 8 == 7)</td>
<td>F</td>
</tr>
<tr>
<td>(10 &lt; 5 and 40 &gt; 20) or 18 &gt;= 10</td>
<td>T</td>
</tr>
</tbody>
</table>

9. Write a truth table for the following expressions:

   a. A or not(B)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A or not (B)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
</tr>
</tbody>
</table>
b. A and (B and C)

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>A and (B and C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
</tr>
</tbody>
</table>

10. Consider the following code segment.

```python
a, b, c = eval(input("Enter three numbers: "))
if a > b:
    if b > c:
        print("Spam Please!")
    else:
        print("It's a late parrot!")
elif b > c:
    print("Cheese Shoppe")
    if a >= c:
        print("Cheddar")
    elif a < b:
        print("Gouda")
    elif c == b:
        print("Swiss")
else:
    print("Trees")
    if a == b:
        print("Chestnut")
    else:
        print("Larch")

print("Done")
```

Show the output that would result from each of the following possible inputs:

(a) 3,4,5

     Trees
     Larch
     Done
(b) 3,3,3

Trees
Chestnut
Done

(c) 5,4,3

Spam Please!
Done

(d) 3,5,2

Cheese Shoppe
Cheddar
Done

(e) 5,4,7

It’s a late parrot!
Done

(f) 3,3,2

Cheese Shoppe
Cheddar
Done
(g) Rewrite the code so that it does NOT use nested **if-else** statements (switch to **elif** clause).

```python
a, b, c = eval(input("Enter three numbers: "))
if a > b and b > c:
    print("Spam Please!")
elif a > b:
    print("It's a late parrot!")
elif b > c and a >= c:
    print("Cheese Shoppe")
    print("Cheddar")
elif b > c and a < b:
    print("Cheese Shoppe")
    print("Gouda")
elif b > c and c == b:
    print("Cheese Shoppe")
    print("Swiss")
elif a == b:
    print("Trees")
    print("Chestnut")
else:
    print("Larch")

print("Done")
```
11. Write a program that computes the fuel efficiency of a multi-leg journey. The program will first prompt for the starting odometer reading and then get information about a series of legs. For each leg, the user enters the current odometer reading and the amount of gas used (separated by a space). The user signals the end of the trip with a blank line. The program should print out the miles per gallon achieved on each leg and the total MPG for the trip.

```python
last_odometer = eval(input("Odometer reading: "))
sum = 0

    legs = 0
    while True:
        in = input("Enter odometer reading and amount of gas (blank line signals end of trip)"
                   if in.strip() == "":
            break
        fields = in.split(" ")
        odometer = eval(fields[0])
        gas = eval(fields[1])
        print((last_odometer - odometer)/gas)
        sum = sum + (last_odometer - odometer)/gas
        legs = legs + 1
        last_odometer = odometer
    print(sum/legs)
```