Name: _____

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
Total	

Circle course section: **CIS-166 MW 1-3** **CIS-166 MW 11-1**
CIS-166 MW 6-8 **CMP-230 TTh 1-3**
CMP-230 TTh 9-11 **CMP-230 MW 9-11**
CMP-230 MW 11-1 **CMP-230 TTh 6-8**

Lehman College, CUNY
CIS-166 and CMP-230 Final Exam, Fall 2013, Version 1

All Reference Tables Are On The Last Sheet Of The Exam

1. What is the exact output of the following?

```
my_friends = "Linus Torvalds,Steve Jobs,Bill Gates,Monty Python"
friends_list = my_friends.split(",")
count = len(friends_list)
favorite = friends_list[0].split(" ")
what = favorite[0].replace("s","x")
l_names = ""

for f in friends_list:
    l_names = l_names + ", " + f.split(" ")[-1]

print("I have", count, "good friends:")
print(l_names.strip(", "))
print("My favorite friend is", favorite[0])
print("who invented", what.upper())
```

Output:

2. Using Python syntax, define a function named `eng2si` that accepts 2 parameters, `feet` and `inches` and returns the total distance (feet + inches combined) in meters and centimeters. In other words, the 2 input parameters to the function indicate the number of feet and some number of inches. The function returns 2 values: the resulting conversion in meters, and the same result in centimeters.

   Hints:

   1 foot = 0.3048 meters
   1 foot = 12 inches
   1 meter = 100 centimeters

3. Complete the following program using Zelle's graphics:

```
def main():
   w = getWindow()          # creates and returns a window
   p1,p2 = getPoints(w)     # get two points by mouse clicks
   c = getColor()           # ask user to enter a color
   displayShape(w,c,p1,p2)  # draw a Rectangle colored with c, using points
                            # p1 and p2 in window w

main()
```

(That is, write the functions getWindow(), getPoints(), getColor() and displayShape().)

4. Write the definition of a function named `getTotalPrice` that

   - takes two parameters:
     - `prices`: a list of prices.
     - `discountApplies`: a boolean indicating whether or not a discount applies.
   - **returns** `total`: the total amount to be paid.

   For each of the prices in the given list:

   - if the price is less than 100, then 3% tax is added to the price.
   - if the price is larger than or equal to 100, then 5% tax is added to the price.

   If discount applies, then the total cost is reduced by 10%. Otherwise the total price remains unchanged.

   For example, `getTotalPrice([50,150,200], True)` returns 377.1, whereas `getTotalPrice([50,150,200], False)` returns 419.0.

5. What is returned when the function `foo()` is invoked on the inputs below?

```
def kuwae( inLst ):
    tot = 1
    for item in inLst:
        tot = tot * item
    return tot

def foo( inLst ):
    if (inLst[-1] > inLst[0]):
        return kuwae( inLst )
    else:
        return -1
```

(a) `foo( [2, 4, 6, 8] )` **Return:** [ ]

(b) `foo( [4002, 328, 457, 1] )` **Return:** [ ]

6. Given the following program and input, what is printed:

```
def main():
    infile = open("infile.txt", "r")
    for line in infile:
        index = line.find(" ")
        if line.find("de") > -1:
            print( line[:index] )
        else:
            print( line[index+1:], end="" )

main()
```

infile.txt

```
Punta Cana
San Juan de la Maguana
San Pedro de Macoris
Puerto Plata
Santo Domingo
```

Output:

7. Write a **program** that reads in a text file, `infile.txt` and writes to an output file, `outfile.txt`. Your program should write all the lines in `infile.txt` that have more than 15 characters to `outfile.txt` in all upper case.

8. Draw what would be displayed in the graphics window when the following program is executed. Remember to indicate the final position and direction of the turtle at the end of program. The turtle always starts by pointing to the right.

```
from turtle import *
def draw(tur, a, xs):
    tur.right(90 * a)
    for x in xs:
        tur.forward(x)
        tur.backward(x)
        tur.right(90)
        tur.forward(10)
        tur.left(90)

t = Turtle()
ls = [64, 16, 32, 16, 64, 16, 32, 16, 64]
draw(t, 0, ls)
```

Graphics Displayed:

9. Write Python code for the following algorithm:

```
function  decodeMessage(numbers, k)
    create an empty message
    for each number in the list of numbers
        code = 97 + ((number + k) modulo 26)
        convert the code to the corresponding Unicode character
        concatenate the character to the beginning of the message
    return the message
```

10. Write Python code for each of the following 4 functions:

    (a) `square(n)` - Returns n^2.

    (b) `iSquareRoot(n)` - Returns the smallest integer greater than or equal to $\sqrt{n}$.
    Examples: `iSquareRoot(24) = 5`, `iSquareRoot(25) = 5`, `iSquareRoot(26) = 6`.

    (c) `perfectSquare(n)` - Returns `True` if `n` is a perfect square, returns `False` otherwise. `n` is a perfect square if, and only if, **square(iSquareRoot(n)) = n**.

    (d) `main()` - prints i "is a perfect square" for every perfect square i from 0 to 100, inclusive.

Useful String Methods: (from p 140 of textbook)

Function	Meaning
`s.capitalize()`	Copy of `s` with only the first character capitalized.
`s.center(width)`	Copy of `s` is centered in a field of given width.
`s.count(sub)`	Count the number of occurrences of `sub` in `s`.
`s.find(sub)`	Find the first position where `sub` occurs in `s`.
`s.join(list)`	Concatenate `list` into a string using `s` as a separator.
`s.ljust(width)`	Like `center`, but `s` is left-justified.
`s.lower()`	Copy of `s` with all characters converted to lowercase.
`s.lstrip()`	Copy of `s` with leading whitespace removed.
`s.replace(oldsub,newsub)`	Replace all occurrences of `oldsub` in `s` with `newsub`.
`s.rfind(sub)`	Like `find`, but returns rightmost position.
`s.rjust(sub)`	Like `center`, but `s` is right-justified.
`s.rstrip()`	Copy of `s` with trailing whitespace removed.
`s.split()`	Split `s` into a list of substrings.
`s.title()`	Copy of `s` with first character of each word capitalized.
`s.upper()`	Copy of `s` with all characters converted to uppercase.

Useful Unicode Ordinal Numbers

letter	Unicode
space	32
!	33
"	34
#	35
$	36
%	37
&	38
'	39
(	40
)	41
*	42
+	43
,	44
-	45
.	46
/	47
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57
:	58
;	59
<	60
=	61
>	62
?	63
@	64

letter	Unicode
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75
L	76
M	77
N	78
O	79
P	80
Q	81
R	82
S	83
T	84
U	85
V	86
W	87
X	88
Y	89
Z	90

letter	Unicode
a	97
b	98
c	99
d	100
e	101
f	102
g	103
h	104
i	105
j	106
k	107
l	108
m	109
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122

Zelle's Graphics Reference: (from p 108-111 of the textbook)

GraphWin Objects
`GraphWin(title, width, height)`
`plot(x,y,color)`
`plotPixel(x,y,color)`
`setBackground(color)`
`close()`
`getMouse()`
`checkMouse()`
`setCoords(xll,yll,xur,yur)`

Graphics Objects
`setFill(color)`
`setOutline(color)`
`setWidth(pixels)`
`draw(aGraphWin)`
`undraw()`
`move(dx,dy)`
`clone()`

Text Methods
`Text(anchorPoint, string)`
`setText(string)`
`getText()`
`getAnchor()`
`setFace(family)`
`setSize(point)`
`setStyle(style)`
`setTextColor(color)`

Point Methods
`Point(x,y)`
`getX()`
`getY()`

Line Methods
`Line(point1, point2)`
`setArrow(string)`
`getCenter()`
`getP1(), getP2()`

Circle Methods
`Circle(centerPoint, radius)`
`getCenter()`
`getRadius()`
`getP1(), getP2()`

Rectangle Methods
`Rectangle(point1,point2)`
`getCenter()`
`getP1(), getP2()`

Oval Methods
`Oval(point1, point2)`
`getCenter()`
`getP1(), getP2()`

Polygon Methods
`Polygon(P1, P2, P3,...)`
`getPoints()`

Turtle Graphics Reference:

Turtle Graphics
`t = Turtle()`
`t.forward(steps)`
`t.backward(steps)`
`t.right(degrees)`
`t.left(degrees)`
`t.up()`
`t.down()`

Name: _____

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
Total	

Circle course section: **CIS-166 MW 1-3** **CIS-166 MW 11-1**
CIS-166 MW 6-8 **CMP-230 TTh 1-3**
CMP-230 TTh 9-11 **CMP-230 MW 9-11**
CMP-230 MW 11-1 **CMP-230 TTh 6-8**

Lehman College, CUNY
CIS-166 and CMP-230 Final Exam, Fall 2013, Version 2

All Reference Tables Are On The Last Sheet Of The Exam

1. What is the exact output of the following?

```
my_friends = "Linus Torvalds,Steve Jobs,Bill Gates,Monthy Python"
count = my_friends.count(" ")
friends_list = my_friends.split(",")
favorite = friends_list[0].split(" ")
what = favorite[0].replace("s","x")
f_names = ""

for f in friends_list:
    f_names = f_names + ", " + f.split(" ")[0]

print("I have", count, "good friends:")
print(f_names.strip(", "))
print("My favorite friend is", favorite[-1])
print("who invented", what.upper())
```

Output:

2. Using Python syntax, define a function named `eng2si` that accepts 2 parameters, `feet` and `inches` and returns the total distance (feet + inches combined) in meters and kilometers. In other words, the 2 input parameters to the function indicate the number of feet and some number of inches. The function returns 2 values: the resulting conversion in meters, and the same result in kilometers.

   Hints:

   1 foot = 0.3048 meters
   1 foot = 12 inches
   1 meter = 0.001 Kilometers

3. Complete the following program using Zelle's graphics:

```
def main():
   w = getWindow()              # creates and returns a window
   p = getPoint(w)              # get a point by mouse click
   c, r = getColorAndRadius()   # ask user for the color and radius
   displayShape(w,c,p,r)        # display Circle colored with c, centered at point p
                                # with radius r, in window w

main()
```

(That is, write the functions getWindow(), getPoint(), getColorAndRadius() and displayShape().)

4. Write the definition of a function named `getTotalPrice` that

   - takes two parameters:
     - `prices`: a list of prices.
     - `discountApplies`: a boolean indicating whether or not a discount applies.
   - **returns** `total`: the total amount to be paid.

   For each of the prices in the given list:

   - if the price is less than 100, then 5% tax is added to the price.
   - if the price is larger than or equal to 100, then 7% tax is added to the price.

   If discount applies, then the total cost is reduced by 20%. Otherwise the total price remains unchanged.

   For example, `getTotalPrice([50,150,200], True)` returns 341.6, whereas `getTotalPrice([50,150,200], False)` returns 427.0.

5. What is returned when the function `foo()` is invoked on the inputs below?

```
def pro( inLst ):
    tot = 1
    for item in inLst:
        tot = tot * item
    return tot

def foo( inLst ):
    if (inLst[-2] > inLst[1]):
        return pro( inLst )
    else:
        return -1
```

(a) `foo( [1, 3, 5, 7] )` **Return:** [ ]

(b) `foo( [4002, 457, 325, 1] )` **Return:** [ ]

6. Given the following program and input, what is printed:

```
def main():
    infile = open("infile.txt", "r")
    for line in infile:
        index = line.rfind(" ")
        if line.find("San") > -1:
            print( line[index+1:], end="" )
        else:
            print( line[:index] )

main()
```

infile.txt

```
Punta Cana
San Juan de la Maguana
San Pedro de Macoris
Puerto Plata
Santo Domingo
```

Output:

7. Write a **program** that reads in a text file, `infile.txt` and writes to an output file, `outfile.txt`. Your program should write all the lines in `infile.txt` that have more than five spaces (so more than six words) to `outfile.txt` in all lowercase.

8. Draw what would be displayed in the graphics window when the following program is executed. Remember to indicate the final position and direction of the turtle at the end of program. The turtle always starts by pointing to the right.

```
from turtle import *
def draw(tur, a, xs):
    tur.right(90 * a)
    for x in xs:
        tur.forward(x)
        tur.backward(x)
        tur.right(90)
        tur.forward(10)
        tur.left(90)

t = Turtle()
ls = [64, 16, 32, 16, 64, 16, 32, 16, 64]
draw(t, 1, ls)
```

Graphics Displayed:

9. Write Python code for the following algorithm:

```
function  decodeMessage(numbers, k)
    create an empty message
    for each number in the list of numbers
        code = 97 + ((number - k) modulo 26)
        convert the code to the corresponding Unicode character
        concatenate the character to the beginning of the message
    return the message
```

10. Write Python code for each of the following 4 functions:

    (a) `square(n)` - Returns n^2.

    (b) `iSquareRoot(n)` - Returns the greatest integer less than or equal to $\sqrt{n}$.
    Examples: `iSquareRoot(24) = 4`, `iSquareRoot(25) = 5`, `iSquareRoot(26) = 5`.

    (c) `perfectSquare(n)` - Returns `True` if `n` is a perfect square, returns `False` otherwise. `n` is a perfect square if, and only if, **square(iSquareRoot(n)) = n**.

    (d) `main()` - prints i "is a perfect square" for every perfect square i from 0 to 100, inclusive.

Useful String Methods: (from p 140 of textbook)

Function	Meaning
`s.capitalize()`	Copy of `s` with only the first character capitalized.
`s.center(width)`	Copy of `s` is centered in a field of given width.
`s.count(sub)`	Count the number of occurrences of `sub` in `s`.
`s.find(sub)`	Find the first position where `sub` occurs in `s`.
`s.join(list)`	Concatenate `list` into a string using `s` as a separator.
`s.ljust(width)`	Like `center`, but `s` is left-justified.
`s.lower()`	Copy of `s` with all characters converted to lowercase.
`s.lstrip()`	Copy of `s` with leading whitespace removed.
`s.replace(oldsub,newsub)`	Replace all occurrences of `oldsub` in `s` with `newsub`.
`s.rfind(sub)`	Like `find`, but returns rightmost position.
`s.rjust(sub)`	Like `center`, but `s` is right-justified.
`s.rstrip()`	Copy of `s` with trailing whitespace removed.
`s.split()`	Split `s` into a list of substrings.
`s.title()`	Copy of `s` with first character of each word capitalized.
`s.upper()`	Copy of `s` with all characters converted to uppercase.

Useful Unicode Ordinal Numbers

letter	Unicode
space	32
!	33
"	34
#	35
$	36
%	37
&	38
'	39
(	40
)	41
*	42
+	43
,	44
-	45
.	46
/	47
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57
:	58
;	59
<	60
=	61
>	62
?	63
@	64

letter	Unicode
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75
L	76
M	77
N	78
O	79
P	80
Q	81
R	82
S	83
T	84
U	85
V	86
W	87
X	88
Y	89
Z	90

letter	Unicode
a	97
b	98
c	99
d	100
e	101
f	102
g	103
h	104
i	105
j	106
k	107
l	108
m	109
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122

Zelle's Graphics Reference: (from p 108-111 of the textbook)

GraphWin Objects
`GraphWin(title, width, height)`
`plot(x,y,color)`
`plotPixel(x,y,color)`
`setBackground(color)`
`close()`
`getMouse()`
`checkMouse()`
`setCoords(xll,yll,xur,yur)`

Graphics Objects
`setFill(color)`
`setOutline(color)`
`setWidth(pixels)`
`draw(aGraphWin)`
`undraw()`
`move(dx,dy)`
`clone()`

Text Methods
`Text(anchorPoint, string)`
`setText(string)`
`getText()`
`getAnchor()`
`setFace(family)`
`setSize(point)`
`setStyle(style)`
`setTextColor(color)`

Point Methods
`Point(x,y)`
`getX()`
`getY()`

Line Methods
`Line(point1, point2)`
`setArrow(string)`
`getCenter()`
`getP1(), getP2()`

Circle Methods
`Circle(centerPoint, radius)`
`getCenter()`
`getRadius()`
`getP1(), getP2()`

Rectangle Methods
`Rectangle(point1,point2)`
`getCenter()`
`getP1(), getP2()`

Oval Methods
`Oval(point1, point2)`
`getCenter()`
`getP1(), getP2()`

Polygon Methods
`Polygon(P1, P2, P3,...)`
`getPoints()`

Turtle Graphics Reference:

Turtle Graphics
`t = Turtle()`
`t.forward(steps)`
`t.backward(steps)`
`t.right(degrees)`
`t.left(degrees)`
`t.up()`
`t.down()`

Name: _____

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
Total	

Circle course section: **CIS-166 MW 1-3** **CIS-166 MW 11-1**
CIS-166 MW 6-8 **CMP-230 TTh 1-3**
CMP-230 TTh 9-11 **CMP-230 MW 9-11**
CMP-230 MW 11-1 **CMP-230 TTh 6-8**

Lehman College, CUNY
CIS-166 and CMP-230 Final Exam, Fall 2013, Version 3

All Reference Tables Are On The Last Sheet Of The Exam

1. What is the exact output of the following?

```
my_friends = "Monty Python,Steve Jobs,Bill Gates,Guido vanRossum"
friends_list = my_friends.split(",")
count = len(friends_list)
favorite = friends_list[3].split(" ")
what = friends_list[0][-6:]
f_names = ""

for f in friends_list:
    f_names = f_names + f.split(" ")[0] + ", "

print("I have", count, "good friends:")
print(f_names.strip(", "))
print("My favorite friend is", favorite[-1])
print("who invented", what.upper())
```

Output:

2. Using Python syntax, define a function named `eng2si` that accepts 2 parameters, `pounds` and `ounces` and returns the total weight (pounds + ounces combined) in grams and milligrams. In other words, the 2 input parameters to the function indicate the number of pounds and some number of ounces. The function returns 2 values: the resulting conversion in grams, and the same result in milligrams.

   Hints:

   1 pound = 453.59 grams
   1 pound = 16 ounces
   1 gram = 1000 milligrams

3. Complete the following program using Zelle's graphics:

```
def main():
   w = getWindow()              # creates and returns a window
   p1,p2,p3 = getPoints(w)      # get three points by mouse clicks
   c = getColor()               # ask user for color
   displayShape(w,c,p1,p2,p3)   # display Polygon colored with c, with vertices
                                # at p1, p2, and p3 in window w

main()
```

(That is, write the functions getWindow(), getPoints(), getColor() and displayShape().)

4. Write the definition of a function named `getTotalPrice` that

   - takes two parameters:
     - `prices`: a list of prices.
     - `discountApplies`: a boolean indicating whether or not a discount applies.
   - **returns** `total`: the total amount to be paid.

   For each of the prices in the given list:

   - if the price is less than 200, then 3% tax is added to the price.
   - if the price is larger than or equal to 200, then 7% tax is added to the price.

   If discount applies, then the total cost is reduced by 10%. Otherwise the total price remains unchanged.

   For example, `getTotalPrice([50,150,200], True)` returns 378.0, whereas `getTotalPrice([50,150,200], False)` returns 420.0.

5. What is returned when the function `foo()` is invoked on the inputs below?

```
def kuwae( inLst ):
    tot = 1
    for i in range( len(inLst) ):
        tot = tot * inLst[i]
    return tot

def foo( inLst ):
    if (inLst[-1] < inLst[0]):
        return kuwae( inLst )
    else:
        return -1
```

(a) `foo( [6, 5, 4, 3] )` **Return:** [ ]

(b) `foo( [1, 457, 325, 4002] )` **Return:** [ ]

6. Given the following program and input, what is printed:

```
def main():
    infile = open("infile.txt", "r")
    for line in infile:
        index = line.rfind(" ")
        if line.find("la") > -1:
            print( line[index+1:], end="" )
        else:
            print( line[:index] )

main()
```

infile.txt

```
Punta Cana
San Juan de la Maguana
San Pedro de Macoris
Puerto Plata
Santo Domingo
```

Output:

7. Write a **program** that reads in a text file, `infile.txt` and writes to an output file, `outfile.txt`. Your program should write all the lines in `infile.txt` that contain the substring 'ss' to `outfile.txt` in title case.

8. Draw what would be displayed in the graphics window when the following program is executed. Remember to indicate the final position and direction of the turtle at the end of program. The turtle always starts by pointing to the right.

```
from turtle import *

def draw(tur, a, xs):
    tur.right(90 * a)
    for x in xs:
        tur.forward(x)
        tur.backward(x)
        tur.right(90)
        tur.forward(10)
        tur.left(90)

t = Turtle()
ls = [64, 16, 32, 16, 64, 16, 32, 16, 64]
draw(t, 2, ls)
```

Graphics Displayed:

9. Write Python code for the following algorithm:

```
function  decodeMessage(numbers, k)
    create an empty message
    for each number in the list of numbers
        code = 97 + ((number * k) modulo 26)
        convert the code to the corresponding Unicode character
        concatenate the character to the end of the message
    return the message
```

10. Write Python code for each of the following 4 functions:

    (a) `square(n)` - Returns n^2.

    (b) `iSquareRoot(n)` - Returns the smallest integer greater than or equal to $\sqrt{n}$.
    Examples: `iSquareRoot(24) = 5`, `iSquareRoot(25) = 5`, `iSquareRoot(26) = 6`.

    (c) `perfectSquare(n)` - Returns `True` if `n` is a perfect square, returns `False` otherwise. `n` is a perfect square if, and only if, **square(iSquareRoot(n)) = n**.

    (d) `main()` - prints i "is not a perfect square" for every non perfect square i from 0 to 100, inclusive.

Useful String Methods: (from p 140 of textbook)

Function	Meaning
`s.capitalize()`	Copy of `s` with only the first character capitalized.
`s.center(width)`	Copy of `s` is centered in a field of given width.
`s.count(sub)`	Count the number of occurrences of `sub` in `s`.
`s.find(sub)`	Find the first position where `sub` occurs in `s`.
`s.join(list)`	Concatenate `list` into a string using `s` as a separator.
`s.ljust(width)`	Like `center`, but `s` is left-justified.
`s.lower()`	Copy of `s` with all characters converted to lowercase.
`s.lstrip()`	Copy of `s` with leading whitespace removed.
`s.replace(oldsub,newsub)`	Replace all occurrences of `oldsub` in `s` with `newsub`.
`s.rfind(sub)`	Like `find`, but returns rightmost position.
`s.rjust(sub)`	Like `center`, but `s` is right-justified.
`s.rstrip()`	Copy of `s` with trailing whitespace removed.
`s.split()`	Split `s` into a list of substrings.
`s.title()`	Copy of `s` with first character of each word capitalized.
`s.upper()`	Copy of `s` with all characters converted to uppercase.

Useful Unicode Ordinal Numbers

letter	Unicode
space	32
!	33
"	34
#	35
$	36
%	37
&	38
’	39
(	40
)	41
*	42
+	43
,	44
-	45
.	46
/	47
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57
:	58
;	59
<	60
=	61
>	62
?	63
@	64

letter	Unicode
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75
L	76
M	77
N	78
O	79
P	80
Q	81
R	82
S	83
T	84
U	85
V	86
W	87
X	88
Y	89
Z	90

letter	Unicode
a	97
b	98
c	99
d	100
e	101
f	102
g	103
h	104
i	105
j	106
k	107
l	108
m	109
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122

Zelle's Graphics Reference: (from p 108-111 of the textbook)

GraphWin Objects
`GraphWin(title, width, height)`
`plot(x,y,color)`
`plotPixel(x,y,color)`
`setBackground(color)`
`close()`
`getMouse()`
`checkMouse()`
`setCoords(xll,yll,xur,yur)`

Graphics Objects
`setFill(color)`
`setOutline(color)`
`setWidth(pixels)`
`draw(aGraphWin)`
`undraw()`
`move(dx,dy)`
`clone()`

Text Methods
`Text(anchorPoint, string)`
`setText(string)`
`getText()`
`getAnchor()`
`setFace(family)`
`setSize(point)`
`setStyle(style)`
`setTextColor(color)`

Point Methods
`Point(x,y)`
`getX()`
`getY()`

Line Methods
`Line(point1, point2)`
`setArrow(string)`
`getCenter()`
`getP1(), getP2()`

Circle Methods
`Circle(centerPoint, radius)`
`getCenter()`
`getRadius()`
`getP1(), getP2()`

Rectangle Methods
`Rectangle(point1,point2)`
`getCenter()`
`getP1(), getP2()`

Oval Methods
`Oval(point1, point2)`
`getCenter()`
`getP1(), getP2()`

Polygon Methods
`Polygon(P1, P2, P3,...)`
`getPoints()`

Turtle Graphics Reference:

Turtle Graphics
`t = Turtle()`
`t.forward(steps)`
`t.backward(steps)`
`t.right(degrees)`
`t.left(degrees)`
`t.up()`
`t.down()`

Name: _____

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
Total	

Circle course section: **CIS-166 MW 1-3** **CIS-166 MW 11-1**
CIS-166 MW 6-8 **CMP-230 TTh 1-3**
CMP-230 TTh 9-11 **CMP-230 MW 9-11**
CMP-230 MW 11-1 **CMP-230 TTh 6-8**

Lehman College, CUNY
CIS-166 and CMP-230 Final Exam, Fall 2013, Version 4

All Reference Tables Are On The Last Sheet Of The Exam

1. What is the exact output of the following?

```
my_friends = "Monty Python,Steve Jobs,Bill Gates,Guido vanRossum"
friends_list = my_friends.split(",")
count = len(friends_list)
favorite = friends_list[1].split(" ")
what = friends_list[2][:4]
f_names = ""

for f in friends_list:
    f_names = f_names + ", " + f.split(" ")[0]

print("I have", count, "good friends:")
print(f_names.strip(", "))
print("My favorite friend is", favorite[-1])
print("who pays the", what.lower())
```

Output:

2. Using Python syntax, define a function named `eng2si` that accepts 2 parameters, `pounds` and `ounces` and returns the total weight (pounds + ounces combined) in grams and kilograms. In other words, the 2 input parameters to the function indicate the number of pounds and some number of ounces. The function returns 2 values: the resulting conversion in grams, and the same result in kilograms.

   Hints:

   1 pound = 453.59 grams
   1 pound = 16 ounces
   1 gram = 0.001 Kilograms

3. Complete the following program using Zelle's graphics:

```
def main():
   w = getWindow()          # creates and returns a window
   p1,p2 = getPoints(w)     # get 2 points by mouse clicks
   c = getColor()           # ask user for color
   displayShape(w,c,p1,p2)  # display Oval colored with c, bounded by
                            # points p1 and p2 in window w

main()
```

(That is, write the functions getWindow(), getPoints(), getColor() and displayShape().)

4. Write the definition of a function named `getTotalPrice` that

   - takes two parameters:
     - `prices`: a list of prices.
     - `discountApplies`: a boolean indicating whether or not a discount applies.
   - **returns** `total`: the total amount to be paid.

   For each of the prices in the given list:

   - if the price is less than 200, then 5% tax is added to the price.
   - if the price is larger than or equal to 200, then 8% tax is added to the price.

   If discount applies, then the total cost is reduced by 20%. Otherwise the total price remains unchanged.

   For example, `getTotalPrice([50,150,200], True)` returns 340.8, whereas `getTotalPrice([50,150,200], False)` returns 426.0.

5. What is returned when the function `foo()` is invoked on the inputs below?

```
def pro( inLst ):
    tot = 1
    for i in range( len(inLst) ):
        tot = tot * inLst[i]
    return tot

def foo( inLst ):
    if (inLst[-2] < inLst[1]):
        return pro( inLst )
    else:
        return -1
```

(a) `foo( [5, 4, 3, 2] )` **Return:** [ ]

(b) `foo( [4002, 325, 457, 1] )` **Return:** [ ]

6. Given the following program and input, what is printed:

```
def main():
    infile = open("infile.txt", "r")
    for line in infile:
        index = line.find(" ")
        if line.find("la") > -1:
            print( line[:index] )
        else:
            print( line[index+1:], end="" )

main()
```

infile.txt

```
Punta Cana
San Juan de la Maguana
San Pedro de Macoris
Puerto Plata
Santo Domingo
```

Output:

7. Write a **program** that reads in a text file, `infile.txt` and writes to an output file, `outfile.txt`. Your program should write the first five characters of each line in `infile.txt` that has more than two instances of `'e'` to `outfile.txt`.

8. Draw what would be displayed in the graphics window when the following program is executed. Remember to indicate the final position and direction of the turtle at the end of program. The turtle always starts by pointing to the right.

```
from turtle import *
def draw(tur, a, xs):
    tur.right(90 * a)
    for x in xs:
        tur.forward(x)
        tur.backward(x)
        tur.right(90)
        tur.forward(10)
        tur.left(90)

t = Turtle()
ls = [64, 16, 32, 16, 64, 16, 32, 16, 64]
draw(t, 3, ls)
```

Graphics Displayed:

9. Write a Python code for the following algorithm:

```
function  decodeMessage(numbers, k)
    create an empty message
    for each number in the list of numbers
        code = 97 + (power(number, k) modulo 26)
        convert the code to the corresponding Unicode character
        concatenate the character to the end of the message
    return the message
```

Note that `power(number, k)` simply means the number raised to the power of k.

10. Write Python code for each of the following 4 functions:

    (a) `square(n)` - Returns n^2.

    (b) `iSquareRoot(n)` - Returns the greatest integer less than or equal to $\sqrt{n}$.
    Examples: `iSquareRoot(24) = 4`, `iSquareRoot(25) = 5`, `iSquareRoot(26) = 5`.

    (c) `perfectSquare(n)` - Returns `True` if `n` is a perfect square, returns `False` otherwise. `n` is a perfect square if, and only if, **square(iSquareRoot(n)) = n**.

    (d) `main()` - prints i "is not a perfect square" for every non-perfect square i from 0 to 100, inclusive.

Useful String Methods: (from p 140 of textbook)

Function	Meaning
`s.capitalize()`	Copy of `s` with only the first character capitalized.
`s.center(width)`	Copy of `s` is centered in a field of given width.
`s.count(sub)`	Count the number of occurrences of `sub` in `s`.
`s.find(sub)`	Find the first position where `sub` occurs in `s`.
`s.join(list)`	Concatenate `list` into a string using `s` as a separator.
`s.ljust(width)`	Like `center`, but `s` is left-justified.
`s.lower()`	Copy of `s` with all characters converted to lowercase.
`s.lstrip()`	Copy of `s` with leading whitespace removed.
`s.replace(oldsub,newsub)`	Replace all occurrences of `oldsub` in `s` with `newsub`.
`s.rfind(sub)`	Like `find`, but returns rightmost position.
`s.rjust(sub)`	Like `center`, but `s` is right-justified.
`s.rstrip()`	Copy of `s` with trailing whitespace removed.
`s.split()`	Split `s` into a list of substrings.
`s.title()`	Copy of `s` with first character of each word capitalized.
`s.upper()`	Copy of `s` with all characters converted to uppercase.

Useful Unicode Ordinal Numbers

letter	Unicode
space	32
!	33
"	34
#	35
$	36
%	37
&	38
'	39
(	40
)	41
*	42
+	43
,	44
-	45
.	46
/	47
0	48
1	49
2	50
3	51
4	52
5	53
6	54
7	55
8	56
9	57
:	58
;	59
<	60
=	61
>	62
?	63
@	64

letter	Unicode
A	65
B	66
C	67
D	68
E	69
F	70
G	71
H	72
I	73
J	74
K	75
L	76
M	77
N	78
O	79
P	80
Q	81
R	82
S	83
T	84
U	85
V	86
W	87
X	88
Y	89
Z	90

letter	Unicode
a	97
b	98
c	99
d	100
e	101
f	102
g	103
h	104
i	105
j	106
k	107
l	108
m	109
n	110
o	111
p	112
q	113
r	114
s	115
t	116
u	117
v	118
w	119
x	120
y	121
z	122

Zelle's Graphics Reference: (from p 108-111 of the textbook)

GraphWin Objects
`GraphWin(title, width, height)`
`plot(x,y,color)`
`plotPixel(x,y,color)`
`setBackground(color)`
`close()`
`getMouse()`
`checkMouse()`
`setCoords(xll,yll,xur,yur)`

Graphics Objects
`setFill(color)`
`setOutline(color)`
`setWidth(pixels)`
`draw(aGraphWin)`
`undraw()`
`move(dx,dy)`
`clone()`

Text Methods
`Text(anchorPoint, string)`
`setText(string)`
`getText()`
`getAnchor()`
`setFace(family)`
`setSize(point)`
`setStyle(style)`
`setTextColor(color)`

Point Methods
`Point(x,y)`
`getX()`
`getY()`

Line Methods
`Line(point1, point2)`
`setArrow(string)`
`getCenter()`
`getP1(), getP2()`

Circle Methods
`Circle(centerPoint, radius)`
`getCenter()`
`getRadius()`
`getP1(), getP2()`

Rectangle Methods
`Rectangle(point1,point2)`
`getCenter()`
`getP1(), getP2()`

Oval Methods
`Oval(point1, point2)`
`getCenter()`
`getP1(), getP2()`

Polygon Methods
`Polygon(P1, P2, P3,...)`
`getPoints()`

Turtle Graphics Reference:

Turtle Graphics
`t = Turtle()`
`t.forward(steps)`
`t.backward(steps)`
`t.right(degrees)`
`t.left(degrees)`
`t.up()`
`t.down()`