CSCI141 -03 -07 PRACTICE EXAM

1. Draw to the right of the code, what the turtle would draw when executing main below. Assume proper imports have been done, and that the turtle starts at the center; pen down, facing east.

```
import turtle as tt
LEN1 = 100
LEN2 = LEN1 / (2 * math.sqrt(3) ) # about 29

def draw_part( ):
      tt.forward( LEN2 )
      tt.pendown( )
      tt.right(90)
      tt.forward(LEN1 / 2)
      tt.right(120)
      tt.forward(LEN1)
      tt.right(120)
      tt.forward(LEN1)
      tt.right(120)
      tt.forward(LEN1 / 2)
      tt. left(90)
      tt.penup( )
      tt.backward( LEN2)

def main( ):
      tt.penup( )
      tt.left( 90)
      draw_part( )
      tt.left(180)
      draw_part( )
```

Is the turtle state changed from the start to the end of **draw_part**? If so, how?

Is the turtle state changed from the start to the end of **main**? If so, how?

2. For each expression, write the value it would evaluate to, the value's type (str, int, float, bool). If the expression is invalid, write ERROR for the resulting value.

| h = 'hello | w = 'world' | z = 0 | x = 15 | y = 37 |
|---|---|---|---|---|
| n = 2 | t = '22' | d = 'd' | c = 'CompSci144EIOU' | |

| **Expression** | **Resulting value or 'ERROR'** | **Type of value or 'ERROR'** |
|---|---|---|
| h + w | | |
| x / 2 | | |
| x // 2 | | |
| x / z | | |
| n * t | | |
| n * x | | |
| w + y | | |
| int( t ) - y | | |
| h[ 1 ] | | |
| h[ 1 : ] | | |
| h[ : 1] | | |
| c[ 0 : 13 : 4 ] | | |
| c[7 : 12] | | |
| x == 15 | | |
| z > x | | |
| d < 'j' | | |
| t > x | | |
| True or False | | |
| True and False | | |
| len( h + ',' + w ) | | |

3. Compute the results of calling the function below

```
def rek( n ):
      if n == 0:
            return 1
      else:
            t = lupe( n )
            return t * rek( n - 1 )

def lupe( n ):
      s = 0
      while n > 0:
            s = s + n
            n = n - 1
      return s
```

| **n** | **result of calling lupe( n )** | **result of calling rek( n )** |
|---|---|---|
| 0 | | |
| 1 | | |
| 2 | | |
| 3 | | |
| 4 | | |

[ TRUE / FALSE ] rek is **tail** recursive

Rewrite lupe using a **range based for loop**

4. Given the tail recursive function reverse and it's helper function **reverse_rec**,

```
def reverse ( s1 ):
      return reverse_rec(s1, “”)

def reverse_rec(s1, acc):
      if s1 = “”
            return acc
      else:
            return reverse_rec(s1[ 1 : ], s1[ 0 ], acc)
```

finish the substitution trace below.

```
reverse( “ABD” ) = reverse_rec( “ABD”, “”)
                 =
```

The above algorithm is tail recursive. Rewrite it as a single function using a **while loop**.

5. Given the following code, show the **graphical output**. The origin is the center of the grid, the turtle is pen down facing east. Each cell is 100 x 100 units in size. Your output should not extend beyond the grid.

```
from turtle import *

def drawPicture(maxSteps):
      loop = 100
      for i in range(4):
            while (loop < maxSteps):
                  forward(loop)
                  left(90)
                  forward(100)
                  left(90)
                  loop = loop + 100
      done( )

drawPicture(600)
```