HW 6-7 Sols

Prog 1 # The Euclidean alg

```
def gcd(y, x):
    while x>0:
        oldx=x
        x=y%x
        y=oldx
        #print(oldx,x,y)
    return oldx
print(gcd(287,92))
```

Prog2 # Adding binary (Note: reads the binary backwards)

```
import time
def add_bin(list1,list2):
    n=len(list1)
    lst=[0]*(n+1)
    c=0
    for i in range(n):
        lst[i]=(list1[i]+list2[i]+c)%2

        c=(list1[i]+list2[i]+c)//2  # c is the carry
    lst[n]=c
    return lst
```

prog 4 Karatsuba

```
import math
global count  # counts recursive calls
count = 0
x=1234567891011121314151617181920212223242526272829303132333435363
y=1357911131517192123252729313335373941434547495153555759616365676
#x=1234
#y=5678

def kara(x, y):
    #print("x y = ", x, y)
    global count  # Counts recursive calls
    count += 1

    n = max(int(math.log10(x+.001))+1,int(math.log10(y+.001))+1)
    # The .001 is a kludge to fix when x=0
    if x<10 or y<10:
        return x*y
    else:
        frontx, backx = int(x//(10**(n//2))), x % int((10**(n//2)))
        fronty, backy = int(y//(10**(n//2))), y % int((10**(n//2)))
        z2 = kara(frontx, fronty)
        z0 = kara(backx, backy)
        z1 = kara(frontx+backx, fronty+backy)-z2-z0

        return 10**(2*(n//2))*z2+10**(n//2)*z1+z0


print(kara(x,y),'\n')

print("recursive calls = ",count,'\n')  # Prints recursive calls

print(x*y)
```

```
16764334818177052661295147374180744306860071176019146696226647848396037553321171925334
14749453310762762797651012520861917800388

recursive calls =  1525
```

Hw 6-7 Sols

2) b) $101 - 100 = 1 = \underline{gcd}$

c) $gcd(1001, 1331) = gcd(330, 1001) =$
$gcd(11, 330) = \boxed{11}$

d) $gcd(1529, 14039) = gcd(278, 1529)$
$= gcd(139, 278) \Rightarrow \boxed{139}$

e) $gcd(1529, 14038) = gcd(277, 1529)$
$\rightarrow (144, 277) \rightarrow (133, 144) \rightarrow (11, 133) \rightarrow (1, 11)$
$\boxed{1}$

f) $gcd(11111, 111111) \rightarrow \boxed{1}$

3) $(21, 34) \xrightarrow[1]{} (13, 21) \xrightarrow[2]{} (8, 13) \xrightarrow[3]{} (5, 8) \xrightarrow[4]{} (3, 5)$

$\xrightarrow{5} (2, 3) \xrightarrow{6} (1, 2)$

$\boxed{6}$

4) $(34, 55) \rightarrow (21, 34) \rightarrow (13, 21) \rightarrow (8, 13) \rightarrow (5, 8)$

$\rightarrow (3, 5) \rightarrow (2, 3) \rightarrow (1, 2)$

$\boxed{7}$

HW 6-7

18) a) $22 \to 10110 \to \boxed{010110}$

b) $31 \to 11111 \to \boxed{011111}$

c) $-7 \to 111 \to 000111 \to \boxed{111000}$

d) $-19 \to 10011 \to 010011 \to \boxed{101100}$

---

19) a) $11001 \to -(1001) \to -(0110) \to \boxed{-6}$

b) $01101 \to \boxed{13}$

c) $10001 \to -(1110) \to \boxed{-14}$

d) $11111 \to -(1111) = \boxed{0}$

---

24) a) $22 \to \boxed{010110}$

b) $31 \to \boxed{011111}$

c) $n=6 \quad 2^5-7 = 25$ so 2's compl. $= 111001$

d) $2^5-19 = 13 \Rightarrow$ 2's compl $= 101101$

Notice how you get from 1's complement to 2's complement

---

25) a) $11001 \to 11000 = -(0111) = \boxed{-7}$

1's

b) $01101 \to \boxed{13}$

c) $10001 \rightarrow -(0000) \rightarrow -(1111) = \boxed{-15}$

1's

d) $11111 \rightarrow -(1110) \rightarrow -(0001) = \boxed{-1}$

1's

39) See back of text

Prob K is interesting:a) Should be 3 calls for each level. ie for 2 -4 digits it would be 1 call for then 3 calls plus 3 more calls for each of 3 calls for 1 + 3 + 9 =13 calls but its more complicated since sometimes the adding of two n digit numbers will have sums with more digits. ie a few extra calls are required.

b) with two 8 digit numbers it should be 1+3+9+3^(3)

c) in general it would be 1 + 3 + 9 + …+3^(log(n)) This is a geometric series

1 + r + r^2 + .. r^n = (1-r^(n+1))/(1-r).

so we get 1 + 3 + 9 + …+3^(log(n)) = (1-3^(log(n)+1)/(1 - 3)