GCD

Code to compute integer d=gcd(m.n) and find x,y so that xm+yn=d

```
# returns unique (d, x, y) such that d is the greatest common divisor of (m,n)
and mx + ny = d
def gcd(m,n):
    a = max(abs(m),abs(n))
    b = min(abs(m),abs(n))
    A = matrix(2,2,[0,1,1,0])
    while b != 0:
        q = int(a / b)
        (a, b) = (b, a % b)
        A *= matrix(2,2,[0,1,1,-q])
    if (abs(m) == min(abs(m),abs(n))):
        x, y = A[0,0], A[1,0]
    else:
        x, y = A[1,0], A[0,0]
    d = sign(m)*m*x + sign(n)*n*y
    return (d, sign(m)*x, sign(n)*y)
```

```
gcd(400,-75)
```

```
    (25, 1, 5)
```

```
gcd(-23,-5)
```

```
    (1, -2, 9)
```

```
def check_gcd():
    for m in srange(1,100):
        for n in srange(1,100):
            answer = gcd(m,n)
            if (m%answer[0]!=0) or (n%answer[0]!=0) or
(answer[0]!=m*answer[1]+n*answer[2]):
                print("error at" + str(m) + "," + str(n))
            answer = gcd(-m,n)
            if (-m%answer[0]!=0) or (n%answer[0]!=0) or (answer[0]!=-
m*answer[1]+n*answer[2]):
                print("error at" + str(-m) + "," + str(n))
            answer = gcd(m,-n)
            if (m%answer[0]!=0) or (-n%answer[0]!=0) or
(answer[0]!=m*answer[1]-n*answer[2]):
                print("error at" + str(m) + "," + str(-n))
            answer = gcd(-m,-n)
            if (-m%answer[0]!=0) or (-n%answer[0]!=0) or (answer[0]!=-
m*answer[1]-n*answer[2]):
                print("error at" + str(-m) + "," + str(-n))
    return("no errors")
```

```
check_gcd()
```

```
    'no errors'
```