cp_hw1

September 10, 2020

1 Han Gao 17353019

```
In [2]: %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

1.1 hw 1.1

```
In [3]: x = float(input('x = '))
        y = float(input('y = '))

        r = np.sqrt(x**2 + y**2)
        theta = np.arctan(y/x)
        # theta in 0~360 deg
        if x < 0:
            theta += np.pi
        elif x > 0 and y < 0:
            theta += 2*np.pi

        theta = theta*180/np.pi

        print('(r,theta) = (' + str(r) + ',' + str(theta) + ')')
```

```
x = 3
y = 4
(r,theta) = (5.0,53.13010235415598)
```

1.2 hw 1.2

```
In [4]: # calculate n!
        def fac(n):
            if n < 2:
                return 1
            else:
```

```
            return fac(n-1)*n

        # calculate C_n^k
        def bin(n,k):
            return int(fac(n)/(fac(n-k)*fac(k)))

        def prob(n,k):
            return bin(n,k)/2**n
```

In [5]:
```
for i in range(1,11):
    nums = []
    for j in range(0,i+1):
        nums.append(bin(i,j))
    print(nums)
```

```
[1, 1]
[1, 2, 1]
[1, 3, 3, 1]
[1, 4, 6, 4, 1]
[1, 5, 10, 10, 5, 1]
[1, 6, 15, 20, 15, 6, 1]
[1, 7, 21, 35, 35, 21, 7, 1]
[1, 8, 28, 56, 70, 56, 28, 8, 1]
[1, 9, 36, 84, 126, 126, 84, 36, 9, 1]
[1, 10, 45, 120, 210, 252, 210, 120, 45, 10, 1]
```

In [6]:
```
print('P(k=60) = '+str(prob(100,60)))
print('P(k>=60) = '+str(sum([prob(100,i) for i in range(60,101)])))
```

```
P(k=60) = 0.010843866711637987
P(k>=60) = 0.0284439668204904
```

1.3 hw 1.3

In [7]:
```
n = 10000

def is_prime(x,ref):
    det = 1
    for num in ref:
        if x%num == 0:
            det -= 1
            break
            #break the loop when x can be devided by an element of reference list
    return bool(det)

def quicker_is_prime(x,ref):
```

```
        det = 1
        for num in ref:
            if num > x**2:
                break
            if x%num == 0:
                det -= 1
                break
        return bool(det)

    def prime_list(n):
        primes = [2]
        for i in range(3,n):
            if is_prime(i,primes):
                primes.append(i)
        return primes

    def quicker_prime_list(n):
        primes = [2]
        for i in range(3,n):
            if quicker_is_prime(i,primes):
                primes.append(i)
        return primes


    print(prime_list(10000))
    print(quicker_prime_list(10000))
```

```
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 9
[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 9
```