Add Binary ¶

In [15]:
```
def binary(a,b):
    return bin(int(a,2)+int(b,2))[2:]
```

In [16]:
```
binary("11","1")
```

Out[16]:
```
'100'
```

Sum of Distance

In [74]:
```
def distance(x1,x2,x3,y1,y2,y3):
    res1=((x2 - x1)**2 + (y2 - y1)**2)**0.5
    res2=((x3 - x2)**2 + (y3 - y2)**2)**0.5
    res3=((x1 - x3)**2 + (y1 - y3)**2)**0.5
    return round(res1+res2+res3,1)
```

In [75]:
```
distance(x1=1,y1=1,x2=2,y2=4,x3=3,y3=6)
```

Out[75]:
```
10.8
```

In [76]:
```
#'42' 42
# -42 -42
# 4193 fcbsc  4193
# ' 193' 193
# range from 2**31
```

In [103]:
```
def atoi(s):
    #whitespaces
    s=s.strip()
    if not s:
        return 0
    # print(int(s))
    # print(int(s.replace(" ","")))
    #Leading whitespaces just strip() if inbetween whitespaces we have to .rep

    #signedness
    sign=1
    if s[0]=="-":
        sign=-1
        s=s[1:]
    elif s[0]=="+":
        s=s[1:]
    #conversion:
    result=0
    for i in s:
        if i.isdigit():
            digit=int(i)
            result=result*10 + digit
        else:
            break
    result=result*sign
    maxi=2**31-1
    mini=-2**31
    if result>maxi:
        return maxi
    elif result<mini:
        return mini
    return result
```

In [107]:
```
atoi("                    -99999999999999999999999999999999999")
```

Out[107]:
```
-2147483648
```

In [124]:
```
def atoi1(s):
    s=s.strip()#whitespaces
    if not s:
        return 0
    sign=1
    #sign
    if s[0]=="-":
        sign=-1
        s=s[1:]
    elif s[0]=="+":
        s=s[1:]
    #conv
    result=0
    for i in s:
        if i.isdigit():
            digit=int(i)
            result=result*10+digit
        else:
            break
    mini,maxi=-2**31,2**31-1
    result=result*sign
    if result>maxi:
        return maxi
    elif result<mini:
        return mini
    return result
```

In [130]:
```
atoi1("   -042")
```

Out[130]:
```
-42
```

In [134]:
```
class Solution(object):
    def myAtoi2(self, s):
        """
        :type s: str
        :rtype: int
        """
        s=s.strip()#whitespaces
        if not s:
            return 0
        sign=1
        #sign
        if s[0]=="-":
            sign=-1
            s=s[1:]
        elif s[0]=="+":
            s=s[1:]
        #conv
        result=0
        for i in s:
            if i.isdigit():
                digit=int(i)
                result=result*10+digit
            else:
                break
        mini,maxi=-2**31,2**31-1
        result=result*sign
        if result>maxi:
            return maxi
        elif result<mini:
            return mini
        return result
```

In [135]:
```
s=Solution()
s.myAtoi2("   -042")
```

Out[135]: -42

Roman To Integer

In [176]:
```
def romantoint(s):
    summ=0
    i=0
    n=len(s)
    d={'I':1,'V':5,'X':10,'L':50,'C':100,'D':500,'M':1000}
    while i<n:
        if i<n-1 and d[s[i]] < d[s[i+1]]:
            summ+=d[s[i+1]]-d[s[i]]
            i+=2
        else:
            summ+=d[s[i]]
            i+=1
    return summ
```

In [177]:
```
romantoint("XXVI")
```

Out[177]: 26

In [199]:
```
def shuffle(a):
    import random
    random.shuffle(a)
    return a
```

In [ ]:

In [200]:
```
shuffle([1,2,3,4,5,6,7,8,9,10])
```

Out[200]: [8, 6, 2, 5, 7, 10, 4, 3, 9, 1]

HayStack Needle Or Substring Problem

In [204]:
```
def substring(haystack,needle):
    return haystack.find(needle)
```

In [205]:
```
substring("hellooanjansadbutsad","sad")
```

Out[205]: 11

In [215]:
```
def tax_calc(sal):
    res=0
    if sal<0:
        return 0
    elif sal>=0 and sal<=5000:
        return "No Tax"
    elif sal>=5001 and sal<=10000:
        res=res+(sal-(sal*0.05+10))
        return res
    elif sal>=10001 and sal<=50000:
        res=res+(sal-(sal*0.1+100))
        return res
    elif sal>=50001 and sal<=100000:
        res=res+(sal-(sal*0.15+150))
        return res
    elif sal>=100000 and sal<=500000:
        res=res+(sal-(sal*0.2+150))
        return res
    elif sal>500000:
        return "Salary Limit Exceeded"
```

In [216]:
```
tax_calc(4000)
```

Out[216]: 'No Tax'

In [217]:
```
tax_calc(5600)
```

Out[217]: 5310.0

In [218]:
```
tax_calc(500000000)
```

Out[218]: 'Salary Limit Exceeded'

In [219]:
```
#1+1+1+1
#2+2
#1+2+1
#2+1+1
#1+1+2
#5 ways
```

In [220]:
```
# Input: n = 10
# Output: 4
# Explanation: There are 4 prime numbers less than 10, they are 2, 3, 5, 7.
```

In [245]:
```
memo={1:1,2:2}
def f(n):
    if n in memo:
        return memo[n]
    else:
        memo[n]=f(n-2)+f(n-1)
        return memo[n]
#     return f(n)
```

In [248]:
```
f(6)
```

Out[248]:
```
13
```

In [253]:
```
n=10
for i in range(1,n):
    for j in range(i+i,n,i):
        print(j)
```

```
2
3
4
5
6
7
8
9
4
6
8
6
9
8
```

In [255]:
```
20**0.5+1
```

Out[255]:
```
5.47213595499958
```

In [270]:
```
a=[True]*4
print(a)
print(sum(a))
```

```
[True, True, True, True]
4
```

In [257]:
```
int(True)
```

Out[257]:
```
1
```

In [258]:
```
float(True)
```

Out[258]:
```
1.0
```

In [296]:
```
def count_prime(n):
    if n==0 or n==1:
        print(0)
    ans=[True]*n
    ans[0]=ans[1]=False
#     print(ans)
    for i in range(2,int(n**0.5)+1):
        for j in range(i+i,n,i):
            if ans[j]:
                ans[j]=False
    return sum(ans)
#     return sum(ans)
```

In [297]:
```
count_prime(10)
```

Out[297]:
```
4
```

In [298]:
```
count_prime(20)
```

Out[298]:
```
8
```

In [299]:
```
count_prime(100)
```

Out[299]:
```
25
```

In [306]:
```
class Solution(object):
    def cd(self, nums):
        """
        :type nums: List[int]
        :rtype: bool
        """
        flag=False
        nums.sort()
        for i in range(len(nums)):
            if i+1<len(nums) and nums[i]==nums[i+1]:
                i+=2
                flag=True
        return flag
```

In [307]:
```
s=Solution()
s.cd(nums=[1,2,1,4,1,5,6])
```

Out[307]:
```
True
```

In [ ]: