Source Code:

```
bw = 30000
schannel_bw = 25

print('Channel Bandwidth..')
dup_ch_bw = 2 * schannel_bw
t_ch = bw / dup_ch_bw
print(dup_ch_bw)
print('Total available channel')
print(t_ch)

cc_bw = 1000
t_cc = cc_bw / dup_ch_bw
print('Total control channel')
print(t_cc)

N = [4, 7, 12]
for i in range(3):
    ch = t_ch / N[i]
    ch_per_cell = int(ch)
    print('Channel per cell')
    print(N[i])
    print(ch_per_cell)
    c = t_cc / N[i]
    cc = round(c)
    v = ch_per_cell - cc
    vc = round(v)
    print('Control channel and voice channel are..')
    print(cc)
    print(vc)
```

Output:

```
Channel Bandwidth  :
50
Total available channel :
600.0
Total control channel  :
20.0
Channel per cell :
4
150
Control channel and voice channel are :
5
145
```

```
Channel per cell :
7
85
Control channel and voice channel are :
3
82
Channel per cell :
12
50
Control channel and voice channel are :
2
48
```

Source Code:

```
import math

R_SI = 15
io = 6
n = [4, 3]

for a in range(2):
    N = 7
    Q = round(math.sqrt(3 * N), 2)
    print('n :', n[a])
    print('Frequency reuse factor:', Q)
    SI = round(10 * math.log10((1 / io) * (Q ** n[a])), 2)
    print('Signal to interference ratio:', SI)
    if R_SI < SI:
        print('N=' + str(N) + ' can be used')
    else:
        print('N=' + str(N) + ' cannot be used')
    if SI < R_SI:
        i = 2
        j = 2
        N = (i ** 2) + (i * j) + (j ** 2)
        Q = round(math.sqrt(3 * N), 2)
        print('n :', n[a])
        print('Frequency reuse factor:', Q)
        SI1 = round(10 * math.log10((1 / io) * (Q ** n[a])), 2)
        print('Signal to interference ratio:', SI1)
        print('N=' + str(N) + ' can be used')
```

Output:

```
n : 4
Frequency reuse factor: 4.58
Signal to interference ratio: 18.65
N=7 can be used
n : 3
Frequency reuse factor: 4.58
Signal to interference ratio: 12.04
N=7 cannot be used
n : 3
Frequency reuse factor: 6.0
Signal to interference ratio: 15.56
N=12 can be used
```

Source Code:

```
import numpy as np

Gos = 0.5 / 100
Au = 0.1
A = np.array([0.005, 1.13, 3.96, 11.1, 80.9])
c = np.array([1, 5, 10, 20, 100])

print('Blocking probability =', Gos)
print('Traffic intensity per user =', Au)
print('Traffic intensity', A)
print('Channel', c)

U = np.ceil(A / Au)
print('Number of users')
print(U)
```

Output:

```
Blocking probability = 0.005
Traffic intensity per user = 0.1
Traffic intensity [5.00e-03 1.13e+00 3.96e+00 1.11e+01 8.09e+01]
Channel [  1   5  10  20 100]
Number of users
[  1.  12.  40. 111. 809.]
```