Exp-1

```
N = input('Enter the number of cells: ');
i0 = N - 1;
n = input('Enter the path loss: ');
SIR = (sqrt(3 * N))^n / i0;
dB = 10 * log10(SIR);

disp(['Signal-to-Interference Ratio (SIR): ', num2str(SIR)]);
disp(['SIR in dB: ', num2str(dB)]);
```

Exp-2

```
radius = input('Enter Radius: ');
area_hexagonal = 2.598 * (radius)^2;
disp(['Area of Hexagonal Cell = ', num2str(area_hexagonal)])
total_channels = input('Enter the number of Channels: ');
disp('// For Small Cluster Size')
num_cells_small = input('Enter Number of cells: ');
C1 = total_channels / num_cells_small;
disp(['Small Channel Capacity = ', num2str(C1)])
total_cells_small = input('Enter Total number of cells in system: ');
sc1 = C1 * total_cells_small;
disp(['System Smallest Capacity = ', num2str(sc1)])
disp('// For Large Cluster Size')
num_cells_large = input('Enter Number of cells: ');
C2 = total_channels / num_cells_large;
disp(['Large Channel Capacity = ', num2str(C2)])
total_cells_large = input('Enter Total number of cells in system: ');
sc2 = total_cells_large * C2;
disp(['System Capacity = ', num2str(sc2)])
if sc1 > sc2
   disp('Capacitance increases with decrease in cluster size')
else
   disp('Capacitance decreases with increase in cluster size')
end
```

Exp-3

```
 % For Three-Sector Case
N = 4;
n = 4;
Q = sqrt(3 * N);
CI = (1 / Q^n) + (1 / (Q + 0.7)^n);
fprintf('For N = %d, the S/I ratio is: %.4f\n', N, CI);

N = 7;
Q = sqrt(3 * N);
CI = (1 / Q^n) + (1 / (Q + 0.7)^n);
fprintf('For N = %d, the S/I ratio is: %.4f\n\n', N, CI);
disp('As N increases, the S/I ratio increases.');

% For Six-Sector Case
N = 4;
Q = sqrt(3 * N);
CI = 1 / (Q + 0.7)^n;
fprintf('\nFor N = %d, the S/I ratio is: %.4f\n', N, CI);

N = 7;
Q = sqrt(3 * N);
CI = 1 / (Q + 0.7)^n;
fprintf('For N = %d, the S/I ratio is: %.4f\n\n', N, CI);
disp('Sectoring increases the S/I ratio.');
```

Exp-4

```
d0 = 100;
p0 = 0.01;
pt = 0.01;
plofd0 = 10.* log10(p0);
for n = 2:0.5:4
   for d = 100:50:2000
      pl = plofd0 + 10. * n.* log10(d./ d0);
      pt_dbm = 10 * log10(pt);
      pr = pt_dbm - pl;
      hold on;
      disp(pr);
      plot(d, pr, 'r*');
   end
end

xlabel('Distance (m)');
ylabel('Received Power (dBm)');
title('Long Distance Path Loss');
grid on;
hold off;

disp('Conclusion: As distance and path loss exponent increase, received power decreases.');
```

Exp-5

```
clc;
close all;
clear all;

code_length = input('Enter Lenght of Code: ');
code = [-1,-1;-1,1];
[r1, c1] = size(code);
disp('Walsh Code Generated');
while r1<code_length
   code = [code,code;code,-1*code];
   [r1, c1] = size(code);
   disp(code);
end
```

Exp-6

```
v = input('Enter velocity (m/s): ');
f = input('Enter frequency (Hz): ');

c = 3e8;
lambda = c / f;
fd = v / lambda;
Bd = 2 * fd;
Tc = 0.493 / fd;

fprintf('lambda: %.2f m\nfd: %.2f Hz\nBd: %.2f Hz\nTc: %.6f s\n', lambda, fd, Bd, Tc);

Ts = input('Enter Ts (microseconds): ') * 1e-6;
Bs = input('Enter Bs (kHz): ') * 1e3;

fprintf('Ts: %.6f s\nBs: %.2f Hz\n', Ts, Bs);

if Ts > Tc && Bs < Bd
   disp('It is fast fading');
else
   disp('It is slow fading');
end
```