Source Code:

```
# Import necessary libraries
import numpy as np

# Define the parameters
blocking_p = 2 / 100
lamda = 2
H = 3 / 60
Au = lamda * H

# System A
print('For system A')
channel_a = 19
cell_A = 394
A = 12
print('Number of users in System A')
Ua = A / Au
print(Ua)
print('Total number of subscribers in System A')
subscriber_A = Ua * cell_A
print(subscriber_A)
percentage_market_penetration_for_A = (subscriber_A / 2000000) * 100
print(percentage_market_penetration_for_A)

# System B
print('For system B')
channel_b = 57
cell_B = 98
Ab = 45
print('Number of users in System B')
Ub = Ab / Au
print(Ub)
print('Total number of subscribers in System B')
subscriber_B = Ub * cell_B
print(subscriber_B)
percentage_market_penetration_for_B = (subscriber_B / 2000000) * 100
print(percentage_market_penetration_for_B)
```

```
# System C
print('For system C')
channel_c = 100
cell_C = 49
Ac = 88
print('Number of users in System C')
Uc = Ac / Au
print(Uc)
print('Total number of subscribers in System C')
subscriber_C = Uc * cell_C
print(subscriber_C)
percentage_market_penetration_for_C = (subscriber_C / 2000000) * 100
print(percentage_market_penetration_for_C)

# Total number of subscribers and market penetration for all three
systems
Total_number_of_subscriber = subscriber_A + subscriber_B + subscriber_C
Market_penetration_for_three_system = (Total_number_of_subscriber /
2000000) * 100
print('Total number of subscribers:', Total_number_of_subscriber)
print('Market penetration for the three systems:',
Market_penetration_for_three_system)
```

Output:

```
For system A
Number of users in System A ,UA =120
Total number of subscriber in system A =47280
percentage_market_penetration_for_A =2.3640
For system B
Number of users in System B, UB=450
Total number of subscriber in system B=44100
percentage_market_penetration_for_B =2.2050
For system C
Number of users in System C ,UC = 880
Total number of subscriber in system C=43120
percentage_market_penetration_for_C =2.1560
Total_number_of_subscriber = 134500
Market_penetration_for_three_system = 6.7250
```

Source Code:

```
import numpy as np

# Define the parameters
c = 3 * 10**8  # Speed of light in m/s
f = 900 * 10**6  # Frequency in Hz
D = 1  # Distance in meters

# Calculate wavelength
lamda = c / f

# Calculate Fraunhofer distance
df = 2 * (D**2) / lamda
print("Fraunhofer distance = ")
print(df)

# Calculate Path Loss
PL = -10 * np.log10((lamda**2) / (((4 * np.pi)**2) * (df**2)))
print("Path Loss = ")
print(PL)
```

Output:

```
Fraunhofer distance =
6.0
Path Loss =
47.08964738250805
```

Source Code:

```
# Define the parameters
c = 3 * 10**8  # Speed of light in m/s
f = 900 * 10**6  # Frequency in Hz

# Calculate wavelength
lamda = c / f

# Calculate vehicle speed in m/s
vehicle_speed = 70 * (1000 / (60 * 60))

print('(A)')
print("The Vehicle is moving directly toward the transmitter")
The_received_frequency_of_A_is = int(f + (vehicle_speed / lamda))
print(f"The received frequency of A is:
{The_received_frequency_of_A_is} Hz")

print('(B)')
print("The Vehicle is moving directly away from the transmitter")
The_received_frequency_of_B_is = int(f - (vehicle_speed / lamda))
print(f"The received frequency of B is:
{The_received_frequency_of_B_is} Hz")

print('(C)')
print("The Vehicle is moving perpendicular to the angle of arrival of
the transmitted signal")
print("The received signal frequency is the same as the transmitted
frequency 900MHz")
```

Output:

```
(A)
The Vehicle is moving directly toward the transmitter
The received frequency of A is: 900000058 Hz
(B)
The Vehicle is moving directly away from the transmitter
The received frequency of B is: 899999941 Hz
(C)
The Vehicle is moving perpendicular to the angle of arrival of the
transmitted signal
The received signal frequency is the same as the transmitted
frequency 900MHz
```

Source Code:

```
# Import necessary library
import numpy as np
# (A)
print('(A)')
Tn = 150  # in microseconds
N = 70
delT = Tn / N
print(f'delT = {delT} us')
# (B)
print('(B)')
Tn = 4  # in microseconds
MBW = 2 / delT
print(f'The maximum Bandwidth that the SMRCIM model can accurately
represent = {MBW} MHz')
delT = (Tn / N) * 1000  # convert to nanoseconds
print(f'delT for SMRCIM urban microcell model is {delT} ns')
RFBW = (2 / delT) * 1000  # convert to MHz
print(f'The maximum RF bandwidth that can be represented is {RFBW}
MHz')
# (C)
print('(C)')
Exdel = 500  # in nanoseconds
delT = Exdel / N
print(f'For indoor channel {delT} ns')

RFBW = (2 / delT) * 1000  # convert to MHz
print(f'The maximum RF bandwidth for the indoor channel model is {RFBW}
MHz')
```

Output:

```
(A)
delT = 2.142857142857143 us
(B)
The maximum Bandwidth that the SMRCIM model can accurately represent =
0.9333333333333333 MHz
delT for SMRCIM urban microcell model is 57.14285714285714 ns
The maximum RF bandwidth that can be represented is 35.0 MHz
(C)
For indoor channel 7.142857142857143 ns
The maximum RF bandwidth for the indoor channel model is 279.99999999999994 MHz
```