Question 1

Script

```matlab
clear; clc;
EMAILS = char('#ks641@company.com','dn266@aol.com','u#m895@company.com',...
    'qu946@nyu.edu','ko737@company.com','of749@nyu.edu',...
    'ek594@hotmail.com','on466@nyu.edu','#ze384@company.com',...
    'db038@nyu.edu','sp188@nyu.edu','#zc497@company.com',...
    'iu690@company.com','ds834@gmail.com','io448@nyu.edu');
nyuDomainName = '@nyu.edu';
companyDomainName = '@company.com';
internChar = '#';

nContacts = size(EMAILS);
NYUEs = [];
CompanyInterns = [];

% Initialize Variables to use in the for loop
nyuIdx = 1;
compIdx = 1;
othIdx = 1;

for i = 1:nContacts
    theEmail = EMAILS(i,:);
    checkNYU = IsFromPlace_(theEmail,nyuDomainName);
    checkCompany = IsFromPlace_(theEmail,companyDomainName);
    if checkNYU
        NYUEs(nyuIdx,:) = theEmail;
        nyuIdx = nyuIdx + 1;
    elseif checkCompany
        if any(theEmail==internChar)
            CompanyInterns(compIdx,:) = theEmail;
            compIdx = compIdx + 1;
        end
    else
        OtherPeops(othIdx,:) = theEmail;
        othIdx = othIdx +1 ;
    end
end
% Convert each vector to a variable vector (after the comparisons
% in the loop above, they are ASCII values)
```
NYUEs = char(NYUEs);
CompanyInterns = char(CompanyInterns);
OtherPeops = char(OtherPeops);

disp('The following email addresses correspond to individuals from NYU:');
disp(NYUEs); disp(newline);
disp('The following email addresses correspond to interns from Company:');
disp(CompanyInterns); disp(newline);
disp('The following email addresses correspond to other people on the list:');
disp(OtherPeops); disp(newline);

function [Log] = IsFromPlace_(EmailToCheck,DomainName)
    DomainL = length(DomainName);
    % We note that all of the email domain names are preceded by the "@" symbol
    % We can use that to take separate the domain name and compare it to DomainName
    domainNameLoc = find(EmailToCheck=='@');
    Log = strcmp(DomainName,EmailToCheck(domainNameLoc:domainNameLoc+DomainL-1));
end

Script

clear; clc;
% The string is a user input
String = input('Please input a string to check: ', 's');
% All that should happen in the script function is calling Function 1 an
display statements
[numOfLetts,numOfNums,numOfOthers,LocsOfLetts,LocsOfNums] = NumOfEach_(String);
disp(['This string contains ' num2str(numOfLetts) ' letters, '...
     num2str(numOfNums) ' digits, and ' num2str(numOfOthers) ' other '
     'characters.']);
disp(['The letters are in the following positions: ' num2str(LocsOfLetts)]);
disp(['The digits are in the following positions: ' num2str(LocsOfNums)]);

function [nLetts,mNums,pOthers,LetLocs,NumLocs] = NumOfEach_(String)
% The first and last values in each range of letters/numbers
firstLower = 'a'; lastLower = 'z';
firstUpper = 'A'; lastUpper = 'Z';
firstNum = '0'; lastNum = '9';
% Call Function 2 to find the location of characters in the ranges of capital
% and lowercase letters, and the number of capital and lowercase letters

Function 2

```matlab
function [LocOfThing, nThisThing] = IsItThis_(String, startVal, endVal)
    IsThisThing = (String >= startVal) & (String <= endVal);
    LocOfThing = find(IsThisThing);
    nThisThing = nnz(IsThisThing); % number of nonzero elements in IsThisThing
    % nThisThing = sum(LocOfThing); % would also work
end
```

**Script**

```matlab
clear; clc;
% Any way to create the string is fine
Vec = input('Enter a string containing uppercase letters and digits: ');
kVal = 4;
[notEnough, startingAt] = ConsecutiveK_(Vec, kVal);
if notEnough
    disp(['Less than ' num2str(kVal) ' consecutive letters or digits!']);
    disp('The value of the second output has no meaning.');</format>
else
disp(['Has at least ' num2str(kVal) ' consecutive letters or digits.']);
disp(['Starting at location: ' num2str(startingAt)]);
end
```

**Function**

```matlab
function [notEnough, atLocation] = ConsecutiveK_(Vec, minConsecutive)
```

---

Function 2

```matlab
function [LocOfThing, nThisThing] = IsItThis_(String, startVal, endVal)
    IsThisThing = (String >= startVal) & (String <= endVal);
    LocOfThing = find(IsThisThing);
    nThisThing = nnz(IsThisThing); % number of nonzero elements in IsThisThing
    % nThisThing = sum(LocOfThing); % would also work
end
```

**Question 3**

```matlab
clear; clc;
% Any way to create the string is fine
Vec = input('Enter a string containing uppercase letters and digits: ');
kVal = 4;
[notEnough, startingAt] = ConsecutiveK_(Vec, kVal);
if notEnough
    disp(['Less than ' num2str(kVal) ' consecutive letters or digits!']);
    disp('The value of the second output has no meaning.');</format>
else
disp(['Has at least ' num2str(kVal) ' consecutive letters or digits.']);
disp(['Starting at location: ' num2str(startingAt)]);
end
```

**Function**

```matlab
function [notEnough, atLocation] = ConsecutiveK_(Vec, minConsecutive)
```
lVec = length(Vec);
notEnough should be true when the number of consecutive letters is less than
the k value, and false otherwise. This condition will be set in the while
loop for now, initialize the value to true so the loop will run
notEnough = true;
% Count of consecutive values in the password
consecutiveCounter = 1;
% Index for the values in the string Vec
idx = 1;
theElement = Vec(idx);

while notEnough && idx < lVec
    idx = idx + 1;
    nextElement = Vec(idx);
    % The condition in the if statement subtracts the ASCII value of
    % theElement from that of nextElement. If the difference is one, they are
    % consecutive
    if nextElement - theElement == 1
        consecutiveCounter = consecutiveCounter + 1;
        notEnough = consecutiveCounter < minConsecutive;
    else
        consecutiveCounter = 1;
    end
    theElement = nextElement;
end

atLocation = idx - minConsecutive + 1;
end

Extra Practice

Script
% Any way to create these two vectors is fine
Needles = [0 2 4 6]
Haystack = [0 4 8 7 4 9 4 3 0]
The script file only contains the vectors and calls the functions
NumberOfTimes = NeedlesHaystack_(Needles,Haystack)

Function

function NumberOfTimes = NeedlesHaystack_(Needles,Haystack)
    nNeedles = length(Needles);
    NumberOfTimes = zeros(1,nNeedles);
    for i = 1:nNeedles
        nTimes = 0; % initialize the number of times a value from Needles
% appears in Haystack to zero
for j = 1:length(Haystack)
    if Haystack(j) == Needles(i)
        nTimes = nTimes + 1;
    end
end
NumberOfTimes(i) = nTimes;
end