Syracuse University

IST-664 NLP Homework2
Contact Finder: Using Regular Expressions to find email addresses and phone numbers

ThulasiRam RuppaKrishnan
IST 664
Professor Michael Larche

Contents

Introduction

This homework is based on a similar homework called SpamLord! from the Stanford NLP course, in particular, both the programs and the data has been adapted.
On the web, many people who want to give their email addresses or phone numbers for people to contact them will try to make it hard for spammers to automatically search text for email addresses and phone numbers by giving non-standard forms of them, sometimes called obfuscation.
Examples:
jurafsky(at)cs.stanford.edu
jurafsky at csli dot stanford dot edu

For this homework, we are going to use regular expressions to try to find these email addresses anyway. Although our main objective is to learn more about using regular expressions, we can also observe what types of obfuscations work best to foil the spammers.

For the obscured email addresses like those above, your job is to return an email address in the standard form:

jurafsky@cs.stanford.edu and jurafsky@csli.stanford.edu

Similarly, for phone numbers, given examples like:

TEL +1-650-723-0293
Phone: (650) 723-0293

About the data and Programme

there is a directory called ContactFinder that has the following structure:

```
ContactFinder/
        ContactFinder.py
        data/
                dev/
                        aiken
                        ashishg
                        . . .
                devGold
```

In the data/dev directory are text files that have html text with obscured emails and phone numbers from faculty and staff at Stanford. (This data is both out-of-date and has been modified to require more interesting regular expressions.) In the data/devGold file are the corresponding standard form emails and phone numbers; these are the correct answers.

The program ContactFinder.py is set up so that you can write a set of regular expressions with parentheses to pick out the parts of the email address or phone numbers that match it in their obscured form. Then there is a function “process_filename” that goes through lines of the file, matches all the regular expressions and builds up a result list called “res” that has the name of the file, an ‘e’ or ‘p’ for email or phone, and the standard format email address or phone number.

The rest of the program processes all the files in the dev directory and combines the standard form results, it compares them with the devGold file and reports how many your program got right.

Analysis and Coding using Regular Expression

1. Regular expressions that correct false positives or fit false negatives

Regular Expression for email address:

True Positives				
Rule No	**Pattern**	**Number of emails captured**	**Email Address**	**Gold**
1	epatterns.append('([A-Za-z.]+)@([A-Za-z.]+)\.edu')	19	('balaji', 'e', 'balaji@stanford.edu'), ('cheriton', 'e', 'cheriton@cs.stanford.edu'), ('engler', 'e', 'engler@lcs.mit.edu'), ('eroberts', 'e', 'eroberts@cs.stanford.edu'), ('fedkiw', 'e', 'fedkiw@cs.stanford.edu'), ('hanrahan', 'e', 'hanrahan@cs.stanford.edu'), ('kosecka', 'e', 'kosecka@cs.gmu.edu'), ('kunle', 'e', 'darlene@csl.stanford.edu'), ('kunle', 'e', 'kunle@ogun.stanford.edu'), ('nass', 'e', 'nass@stanford.edu'), ('nick', 'e', 'nick.parlante@cs.stanford.edu'), ('psyoung', 'e', 'patrick.young@stanford.edu'), ('rinard', 'e', 'rinard@lcs.mit.edu'), ('shoham', 'e', 'shoham@stanford.edu'), ('thm', 'e', 'pkrokel@stanford.edu'), ('widom', 'e', 'siroker@cs.stanford.edu'), ('widom', 'e', 'widom@cs.stanford.edu'), ('zelenski', 'e', 'zelenski@cs.stanford.edu'), ('zm', 'e', 'manna@cs.stanford.edu')	('balaji', 'e', 'balaji@stanford.edu'), ('cheriton', 'e', 'cheriton@cs.stanford.edu'), ('engler', 'e', 'engler@lcs.mit.edu'), ('eroberts', 'e', 'eroberts@cs.stanford.edu'), ('fedkiw', 'e', 'fedkiw@cs.stanford.edu'), ('hanrahan', 'e', 'hanrahan@cs.stanford.edu'), ('kosecka', 'e', 'kosecka@cs.gmu.edu'), ('kunle', 'e', 'darlene@csl.stanford.edu'), ('kunle', 'e', 'kunle@ogun.stanford.edu'), ('nass', 'e', 'nass@stanford.edu'), ('nick', 'e', 'nick.parlante@cs.stanford.edu'), ('psyoung', 'e', 'patrick.young@stanford.edu'), ('rinard', 'e', 'rinard@lcs.mit.edu'), ('shoham', 'e', 'shoham@stanford.edu'), ('thm', 'e', 'pkrokel@stanford.edu'), ('widom', 'e', 'siroker@cs.stanford.edu'), ('widom', 'e', 'widom@cs.stanford.edu'), ('zelenski', 'e', 'zelenski@cs.stanford.edu'), ('zm', 'e', 'manna@cs.stanford.edu')
2	epatterns.append('([A-Za-z.]+)\s+@\s+([A-Za-z.]+)\.edu')	5	('ashishg', 'e', 'ashishg@stanford.edu'), ('ashishg', 'e', 'rozm@stanford.edu'), ('dabo', 'e', 'dabo@cs.stanford.edu'), ('ullman', 'e', 'ullman@cs.stanford.edu'), ('zelenski', 'e', 'zelenski@cs.stanford.edu')	('ashishg', 'e', 'ashishg@stanford.edu'), ('ashishg', 'e', 'rozm@stanford.edu'), ('dabo', 'e', 'dabo@cs.stanford.edu'), ('ullman', 'e', 'ullman@cs.stanford.edu'), ('zelenski', 'e', 'zelenski@cs.stanford.edu')
3	epatterns.append('([A-Za-z.]+)@([A-Za-z.]+)\.EDU')	1	('cheriton', 'e', 'uma@cs.stanford.edu')	('cheriton', 'e', 'uma@cs.stanford.edu')
4	epatterns.append('([A-Za-z]+)\s+at\s+([A-Za-z.]+).edu')	2	('cheriton', 'e', 'cheriton@cs.stanford.edu'), ('lam', 'e', 'lam@cs.stanford.edu')	('cheriton', 'e', 'cheriton@cs.stanford.edu'), ('lam', 'e', 'lam@cs.stanford.edu')

Table 1.1 Regular expressions to match True positives for Email address

False Positives				
Rule No	Pattern	Number of emails captured	Email Address	Gold
1	epatterns.append('([A-Za-z.]+<del>)@([A-Za-z.]+)\.edu')	3	('latombe', 'e', 'latombe<del>@cs.stanford.edu') ('latombe', 'e', 'asandra<del>@cs.stanford.edu') ('latombe', 'e', 'liliana<del>@cs.stanford.edu')	gold: ('latombe', 'e', 'asandra@cs.stanford.edu') gold: ('latombe', 'e', 'latombe@cs.stanford.edu') gold: ('latombe', 'e', 'liliana@cs.stanford.edu')
2	epatterns.append('([A-Za-z]+)\s+at\s+([A-Za-z.]+).edu')	2	('jure', 'e', 'server@cs.stanford.edu') ('plotkin', 'e', 'server@infolab.stanford.edu')	None
3	epatterns.append('([A-Za-z]+)\s+at\s+([A-Za-z]+)\sdt\scom')	1	('ullman', 'e', 'support@gradiance.edu')	gold: ('ullman', 'e', 'support@gradiance.com')
4	epatterns.append('([A-Za-z]+)\s+at\s+[a-zA-Z-!\s<>]+([A-Za-z]+)\s[a-zA-Z-!\s<>]+\sdot\s[a-zA-Z-!\s<>]+edu')	1	('vladlen', 'e', 'vladlen@m.edu')	gold: ('vladlen', 'e', 'vladlen@stanford.edu')
5	epatterns.append('([A-Za-z]+)\s+at\s+([A-Za-z]+\s+dot\s[A-Za-z]+\s+dot\s)edu')	3	('serafim', 'e', 'serafim@cs dot stanford dot .edu') ('hager', 'e', 'hager@cs dot jhu dot .edu') ('subh', 'e', 'uma@cs dot stanford dot .edu')	gold: ('serafim', 'e', 'serafim@cs.stanford.edu') gold: ('hager', 'e', 'hager@cs.jhu.edu') gold: ('subh', 'e', 'uma@cs.stanford.edu')
6	epatterns.append('([A-Za-z]+)\s+AT\s+([A-Za-z]+\s+DOT\s)edu')	1	('subh', 'e', 'subh@stanford dot .edu')	gold: ('subh', 'e', 'subh@stanford.edu')
7	epatterns.append('([A-Za-z]+)\s+at\s+([A-Za-z]+;[A-Za-z]+;)edu')	1	('jks', 'e', 'jks@robotics;stanford;.edu')	gold: ('jks', 'e', 'jks@robotics.stanford.edu')
8	epatterns.append('([A-Za-z]+)\s+at\s+([A-Za-z]+\s[A-Za-z]+\s)edu')	1	('pal', 'e', 'pal@cs stanford .edu')	gold: ('pal', 'e', 'pal@cs.stanford.edu')
9	epatterns.append('([A-Za-z.-]+)@([A-Za-z.-]+)\.-e-d-u')	1	('dlwh', 'e', 'd-l-w-h-@-s-t-a-n-f-o-r-d-.edu')	gold: ('dlwh', 'e', 'dlwh@stanford.edu')
10	epatterns.append('(\(\'[A-Za-z.]+edu\'),(\'[A-Za-z.]+\'\))')	1	('jurafsky', 'e', "('stanford.edu'@'jurafsky').edu")	gold: ('jurafsky', 'e', 'jurafsky@stanford.edu')
11	epatterns.append('([A-Za-z]+)&#x40;([A-Za-z]+.[A-Za-z.]+)edu')	2	('levoy', 'e', 'melissa@graphics.stanford..edu') ('levoy', 'e', 'ada@graphics.stanford..edu')	gold: ('levoy', 'e', 'ada@graphics.stanford.edu') gold: ('levoy', 'e', 'melissa@graphics.stanford.edu')
12	epatterns.append('([A-Za-z]+)\s<at symbol>\s([A-Za-z.]+)edu')	2	('manning', 'e', 'manning@cs.stanford..edu') ('manning', 'e', 'dbarros@cs.stanford..edu')	gold: ('manning', 'e', 'dbarros@cs.stanford.edu') gold: ('manning', 'e', 'manning@cs.stanford.edu')
13	epatterns.append('([A-Za-z]+)\sWHERE\s([A-Za-z\s]+)DOM\sedu')	1	('engler', 'e', 'engler@stanford .edu')	gold: ('engler', 'e', 'engler@stanford.edu')
14	epatterns.append('([A-Za-z.]+)\s[a-z\s&";\(]+@([A-Za-z.]+)edu[a-z\s&;"\)]+')	2	('ouster', 'e', 'teresa.lynn@stanford..edu') ('ouster', 'e', 'ouster@cs.stanford..edu')	gold: ('ouster', 'e', 'teresa.lynn@stanford.edu') gold: ('ouster', 'e', 'ouster@cs.stanford.edu')

Table 1.2 Regular expressions to match False positives for Email address

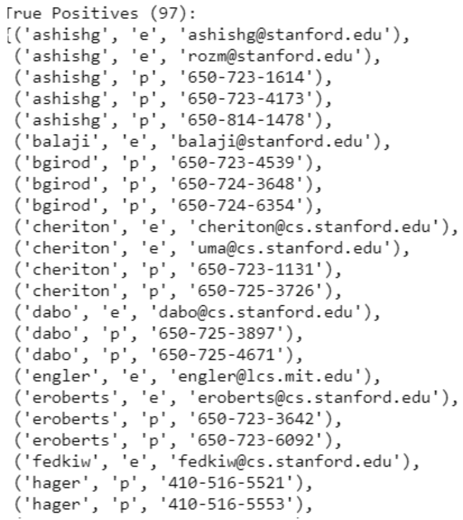
Regular Expression for phone numbers:

True Positives				
Rule No	Pattern	Number of phone captured	Phone Number	Gold
			('cheriton', 'p', '650-723-1131'),	('cheriton', 'p', '650-723-1131'),
			('cheriton', 'p', '650-725-3726'),	('cheriton', 'p', '650-725-3726'),
			('eroberts', 'p', '650-723-3642'),	('eroberts', 'p', '650-723-3642'),
			('eroberts', 'p', '650-723-6092'),	('eroberts', 'p', '650-723-6092'),
			('hager', 'p', '410-516-8000'),	('hager', 'p', '410-516-8000'),
			('rajeev', 'p', '650-723-4377'),	('rajeev', 'p', '650-723-4377'),
			('rajeev', 'p', '650-723-6045'),	('rajeev', 'p', '650-723-6045'),
			('rajeev', 'p', '650-725-4671'),	('rajeev', 'p', '650-725-4671'),
			('subh', 'p', '650-724-1915'),	('subh', 'p', '650-724-1915'),
			('subh', 'p', '650-725-3726'),	('subh', 'p', '650-725-3726'),
			('subh', 'p', '650-725-6949'),	('subh', 'p', '650-725-6949'),
			('ullman', 'p', '650-494-8016'),	('ullman', 'p', '650-494-8016'),
			('ullman', 'p', '650-725-2588'),	('ullman', 'p', '650-725-2588'),
			('ullman', 'p', '650-725-4802'),	('ullman', 'p', '650-725-4802'),
			('widom', 'p', '650-723-0872'),	('widom', 'p', '650-723-0872'),
			('widom', 'p', '650-723-7690'),	('widom', 'p', '650-723-7690'),
			('widom', 'p', '650-725-2588'),	('widom', 'p', '650-725-2588'),
			('zelenski', 'p', '650-723-6092'),	('zelenski', 'p', '650-723-6092'),
1	ppatterns.append(**'(\d{3})-(\d{3})-(\d{4})'**)	19	('zelenski', 'p', '650-725-8596')	('zelenski', 'p', '650-725-8596')
			('ashishg', 'p', '650-723-1614'),	('ashishg', 'p', '650-723-1614'),
			('ashishg', 'p', '650-723-4173'),	('ashishg', 'p', '650-723-4173'),
			('ashishg', 'p', '650-814-1478'),	('ashishg', 'p', '650-814-1478'),
			('bgirod', 'p', '650-723-4539'),	('bgirod', 'p', '650-723-4539'),
			('bgirod', 'p', '650-724-3648'),	('bgirod', 'p', '650-724-3648'),
			('bgirod', 'p', '650-724-6354'),	('bgirod', 'p', '650-724-6354'),
			('dabo', 'p', '650-725-3897'),	('dabo', 'p', '650-725-3897'),
			('dabo', 'p', '650-725-4671'),	('dabo', 'p', '650-725-4671'),
			('hager', 'p', '410-516-5521'),	('hager', 'p', '410-516-5521'),
			('hager', 'p', '410-516-5553'),	('hager', 'p', '410-516-5553'),
			('hanrahan', 'p', '650-723-0033'),	('hanrahan', 'p', '650-723-0033'),
			('hanrahan', 'p', '650-723-8530'),	('hanrahan', 'p', '650-723-8530'),
			('horowitz', 'p', '650-725-3707'),	('horowitz', 'p', '650-725-3707'),
			('horowitz', 'p', '650-725-6949'),	('horowitz', 'p', '650-725-6949'),
			('kosecka', 'p', '703-993-1710'),	('kosecka', 'p', '703-993-1710'),
	ppatterns.append(**'\((\d{3})\)\s?(\d{3})-**		('kosecka', 'p', '703-993-1876'),	('kosecka', 'p', '703-993-1876'),
2	**(\d{4})'**)	47	('kunle', 'p', '650-723-1430'),	('kunle', 'p', '650-723-1430'),

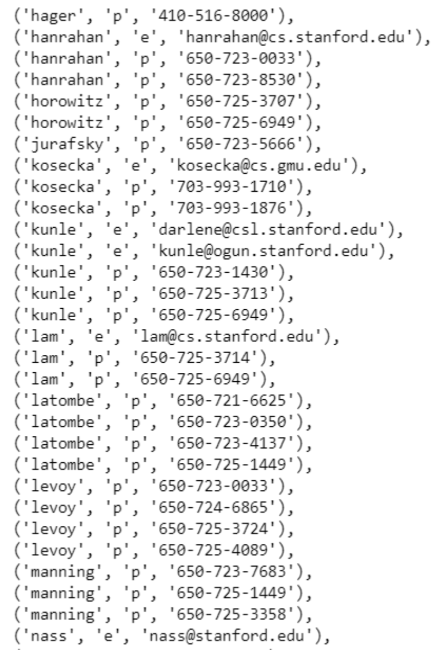
			('kunle', 'p', '650-725-3713'), ('kunle', 'p', '650-725-6949'), ('lam', 'p', '650-725-3714'), ('lam', 'p', '650-725-6949'), ('latombe', 'p', '650-721-6625'), ('latombe', 'p', '650-723-0350'), ('latombe', 'p', '650-723-4137'), ('latombe', 'p', '650-725-1449'), ('levoy', 'p', '650-723-0033'), ('levoy', 'p', '650-724-6865'), ('levoy', 'p', '650-725-3724'), ('levoy', 'p', '650-725-4089'), ('manning', 'p', '650-723-7683'), ('manning', 'p', '650-725-1449'), ('manning', 'p', '650-725-3358'), ('nick', 'p', '650-725-4727'), ('ok', 'p', '650-723-9753'), ('ok', 'p', '650-725-1449'), ('rinard', 'p', '617-253-1221'), ('rinard', 'p', '617-258-6922'), ('serafim', 'p', '650-723-3334'), ('serafim', 'p', '650-725-1449'), ('thm', 'p', '650-725-3383'), ('thm', 'p', '650-725-3636'), ('thm', 'p', '650-725-3938'), ('tim', 'p', '650-724-9147'), ('tim', 'p', '650-725-2340'), ('tim', 'p', '650-725-4671'), ('zm', 'p', '650-723-4364'), ('zm', 'p', '650-725-4671')	('kunle', 'p', '650-725-3713'), ('kunle', 'p', '650-725-6949'), ('lam', 'p', '650-725-3714'), ('lam', 'p', '650-725-6949'), ('latombe', 'p', '650-721-6625'), ('latombe', 'p', '650-723-0350'), ('latombe', 'p', '650-723-4137'), ('latombe', 'p', '650-725-1449'), ('levoy', 'p', '650-723-0033'), ('levoy', 'p', '650-724-6865'), ('levoy', 'p', '650-725-3724'), ('levoy', 'p', '650-725-4089'), ('manning', 'p', '650-723-7683'), ('manning', 'p', '650-725-1449'), ('manning', 'p', '650-725-3358'), ('nick', 'p', '650-725-4727'), ('ok', 'p', '650-723-9753'), ('ok', 'p', '650-725-1449'), ('rinard', 'p', '617-253-1221'), ('rinard', 'p', '617-258-6922'), ('serafim', 'p', '650-723-3334'), ('serafim', 'p', '650-725-1449'), ('thm', 'p', '650-725-3383'), ('thm', 'p', '650-725-3636'), ('thm', 'p', '650-725-3938'), ('tim', 'p', '650-724-9147'), ('tim', 'p', '650-725-2340'), ('tim', 'p', '650-725-4671'), ('zm', 'p', '650-723-4364'), ('zm', 'p', '650-725-4671')
3	ppatterns.append(**'(\d{3})\s(\d{3})\s(\d{4})'**)	2	('pal', 'p', '650-725-9046'), ('jurafsky', 'p', '650-723-5666')	('pal', 'p', '650-725-9046'), ('jurafsky', 'p', '650-723-5666')
4	ppatterns.append(**'\[(\d{3})\]\s?(\d{3})-(\d{4})'**)	2	('nass', 'p', '650-723-5499'), ('nass', 'p', '650-725-2472')	('nass', 'p', '650-723-5499'), ('nass', 'p', '650-725-2472')
5	ppatterns.append(**'(\d{3})\s(\d{3})-(\d{4})'**)	2	('shoham', 'p', '650-725-1449'), ('shoham', 'p', '650-723-3432')	('shoham', 'p', '650-725-1449'), ('shoham', 'p', '650-723-3432')

Table 1.3 Regular expressions to match True positives for Phone Numbers

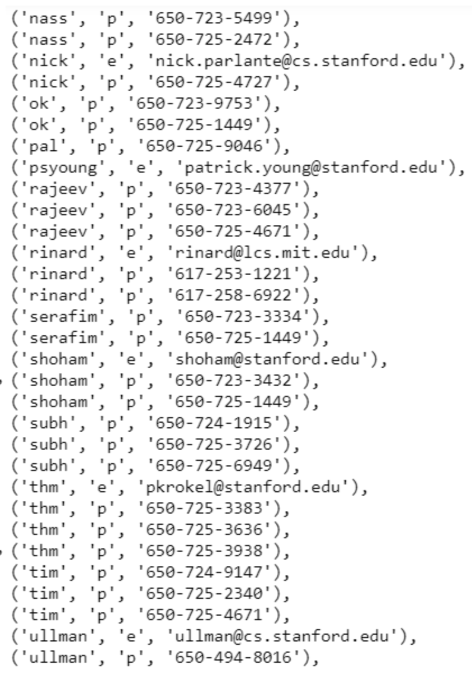
Regex Results (Final Output of the program):



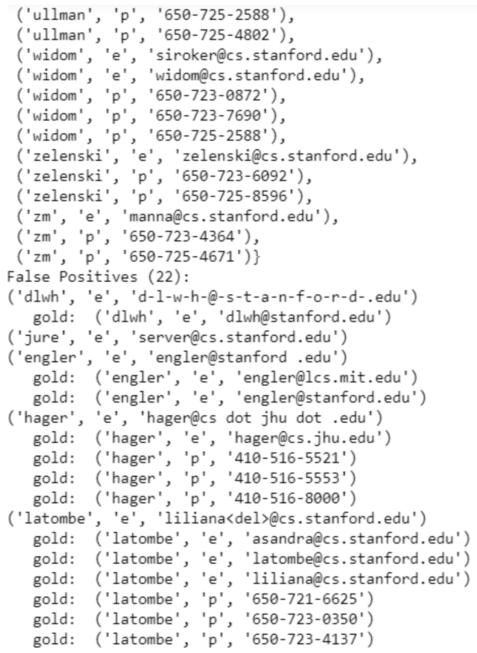
```
True Positives (97):
{('ashishg', 'e', 'ashishg@stanford.edu'),
 ('ashishg', 'e', 'rozm@stanford.edu'),
 ('ashishg', 'p', '650-723-1614'),
 ('ashishg', 'p', '650-723-4173'),
 ('ashishg', 'p', '650-814-1478'),
 ('balaji', 'e', 'balaji@stanford.edu'),
 ('bgirod', 'p', '650-723-4539'),
 ('bgirod', 'p', '650-724-3648'),
 ('bgirod', 'p', '650-724-6354'),
 ('cheriton', 'e', 'cheriton@cs.stanford.edu'),
 ('cheriton', 'e', 'uma@cs.stanford.edu'),
 ('cheriton', 'p', '650-723-1131'),
 ('cheriton', 'p', '650-725-3726'),
 ('dabo', 'e', 'dabo@cs.stanford.edu'),
 ('dabo', 'p', '650-725-3897'),
 ('dabo', 'p', '650-725-4671'),
 ('engler', 'e', 'engler@lcs.mit.edu'),
 ('eroberts', 'e', 'eroberts@cs.stanford.edu'),
 ('eroberts', 'p', '650-723-3642'),
 ('eroberts', 'p', '650-723-6092'),
 ('fedkiw', 'e', 'fedkiw@cs.stanford.edu'),
 ('hager', 'p', '410-516-5521'),
 ('hager', 'p', '410-516-5553'),
```



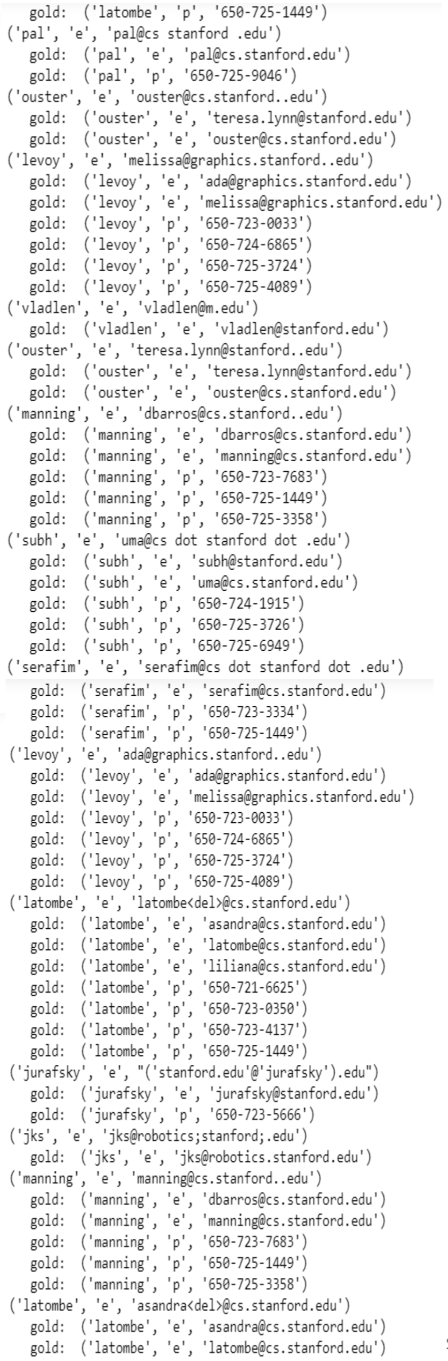
```
 ('hager', 'p', '410-516-8000'),
 ('hanrahan', 'e', 'hanrahan@cs.stanford.edu'),
 ('hanrahan', 'p', '650-723-0033'),
 ('hanrahan', 'p', '650-723-8530'),
 ('horowitz', 'p', '650-725-3707'),
 ('horowitz', 'p', '650-725-6949'),
 ('jurafsky', 'p', '650-723-5666'),
 ('kosecka', 'e', 'kosecka@cs.gmu.edu'),
 ('kosecka', 'p', '703-993-1710'),
 ('kosecka', 'p', '703-993-1876'),
 ('kunle', 'e', 'darlene@csl.stanford.edu'),
 ('kunle', 'e', 'kunle@ogun.stanford.edu'),
 ('kunle', 'p', '650-723-1430'),
 ('kunle', 'p', '650-725-3713'),
 ('kunle', 'p', '650-725-6949'),
 ('lam', 'e', 'lam@cs.stanford.edu'),
 ('lam', 'p', '650-725-3714'),
 ('lam', 'p', '650-725-6949'),
 ('latombe', 'p', '650-721-6625'),
 ('latombe', 'p', '650-723-0350'),
 ('latombe', 'p', '650-723-4137'),
 ('latombe', 'p', '650-725-1449'),
 ('levoy', 'p', '650-723-0033'),
 ('levoy', 'p', '650-724-6865'),
 ('levoy', 'p', '650-725-3724'),
 ('levoy', 'p', '650-725-4089'),
 ('manning', 'p', '650-723-7683'),
 ('manning', 'p', '650-725-1449'),
 ('manning', 'p', '650-725-3358'),
 ('nass', 'e', 'nass@stanford.edu'),
```



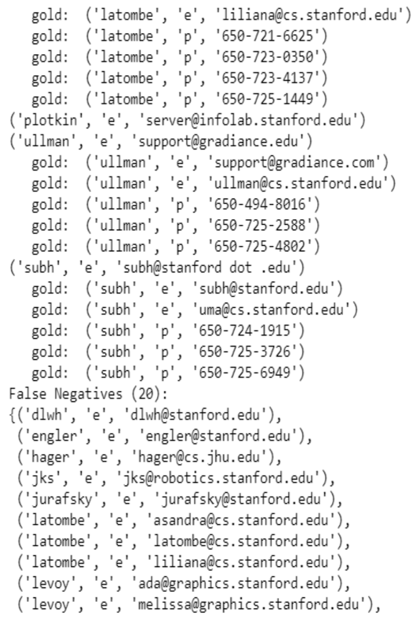
```
 ('nass', 'p', '650-723-5499'),
 ('nass', 'p', '650-725-2472'),
 ('nick', 'e', 'nick.parlante@cs.stanford.edu'),
 ('nick', 'p', '650-725-4727'),
 ('ok', 'p', '650-723-9753'),
 ('ok', 'p', '650-725-1449'),
 ('pal', 'p', '650-725-9046'),
 ('psyoung', 'e', 'patrick.young@stanford.edu'),
 ('rajeev', 'p', '650-723-4377'),
 ('rajeev', 'p', '650-723-6045'),
 ('rajeev', 'p', '650-725-4671'),
 ('rinard', 'e', 'rinard@lcs.mit.edu'),
 ('rinard', 'p', '617-253-1221'),
 ('rinard', 'p', '617-258-6922'),
 ('serafim', 'p', '650-723-3334'),
 ('serafim', 'p', '650-725-1449'),
 ('shoham', 'e', 'shoham@stanford.edu'),
 ('shoham', 'p', '650-723-3432'),
 ('shoham', 'p', '650-725-1449'),
 ('subh', 'p', '650-724-1915'),
 ('subh', 'p', '650-725-3726'),
 ('subh', 'p', '650-725-6949'),
 ('thm', 'e', 'pkrokel@stanford.edu'),
 ('thm', 'p', '650-725-3383'),
 ('thm', 'p', '650-725-3636'),
 ('thm', 'p', '650-725-3938'),
 ('tim', 'p', '650-724-9147'),
 ('tim', 'p', '650-725-2340'),
 ('tim', 'p', '650-725-4671'),
 ('ullman', 'e', 'ullman@cs.stanford.edu'),
 ('ullman', 'p', '650-494-8016'),
```



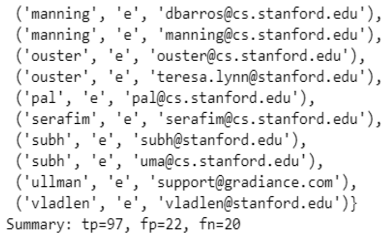
```
 ('ullman', 'p', '650-725-2588'),
 ('ullman', 'p', '650-725-4802'),
 ('widom', 'e', 'siroker@cs.stanford.edu'),
 ('widom', 'e', 'widom@cs.stanford.edu'),
 ('widom', 'p', '650-723-0872'),
 ('widom', 'p', '650-723-7690'),
 ('widom', 'p', '650-725-2588'),
 ('zelenski', 'e', 'zelenski@cs.stanford.edu'),
 ('zelenski', 'p', '650-723-6092'),
 ('zelenski', 'p', '650-725-8596'),
 ('zm', 'e', 'manna@cs.stanford.edu'),
 ('zm', 'p', '650-723-4364'),
 ('zm', 'p', '650-725-4671')}
False Positives (22):
('dlwh', 'e', 'd-l-w-h-@-s-t-a-n-f-o-r-d-.edu')
   gold:  ('dlwh', 'e', 'dlwh@stanford.edu')
('jure', 'e', 'server@cs.stanford.edu')
('engler', 'e', 'engler@stanford .edu')
   gold:  ('engler', 'e', 'engler@lcs.mit.edu')
   gold:  ('engler', 'e', 'engler@stanford.edu')
('hager', 'e', 'hager@cs dot jhu dot .edu')
   gold:  ('hager', 'e', 'hager@cs.jhu.edu')
   gold:  ('hager', 'p', '410-516-5521')
   gold:  ('hager', 'p', '410-516-5553')
   gold:  ('hager', 'p', '410-516-8000')
('latombe', 'e', 'liliana<del>@cs.stanford.edu')
   gold:  ('latombe', 'e', 'asandra@cs.stanford.edu')
   gold:  ('latombe', 'e', 'latombe@cs.stanford.edu')
   gold:  ('latombe', 'e', 'liliana@cs.stanford.edu')
   gold:  ('latombe', 'p', '650-721-6625')
   gold:  ('latombe', 'p', '650-723-0350')
   gold:  ('latombe', 'p', '650-723-4137')
```



```
   gold:  ('latombe', 'p', '650-725-1449')
('pal', 'e', 'pal@cs stanford .edu')
   gold:  ('pal', 'e', 'pal@cs.stanford.edu')
   gold:  ('pal', 'p', '650-725-9046')
('ouster', 'e', 'ouster@cs.stanford..edu')
   gold:  ('ouster', 'e', 'teresa.lynn@stanford.edu')
   gold:  ('ouster', 'e', 'ouster@cs.stanford.edu')
('levoy', 'e', 'melissa@graphics.stanford..edu')
   gold:  ('levoy', 'e', 'ada@graphics.stanford.edu')
   gold:  ('levoy', 'e', 'melissa@graphics.stanford.edu')
   gold:  ('levoy', 'p', '650-723-0033')
   gold:  ('levoy', 'p', '650-724-6865')
   gold:  ('levoy', 'p', '650-725-3724')
   gold:  ('levoy', 'p', '650-725-4089')
('vladlen', 'e', 'vladlen@m.edu')
   gold:  ('vladlen', 'e', 'vladlen@stanford.edu')
('ouster', 'e', 'teresa.lynn@stanford..edu')
   gold:  ('ouster', 'e', 'teresa.lynn@stanford.edu')
   gold:  ('ouster', 'e', 'ouster@cs.stanford.edu')
('manning', 'e', 'dbarros@cs.stanford..edu')
   gold:  ('manning', 'e', 'dbarros@cs.stanford.edu')
   gold:  ('manning', 'e', 'manning@cs.stanford.edu')
   gold:  ('manning', 'p', '650-723-7683')
   gold:  ('manning', 'p', '650-725-1449')
   gold:  ('manning', 'p', '650-725-3358')
('subh', 'e', 'uma@cs dot stanford dot .edu')
   gold:  ('subh', 'e', 'subh@stanford.edu')
   gold:  ('subh', 'e', 'uma@cs.stanford.edu')
   gold:  ('subh', 'p', '650-724-1915')
   gold:  ('subh', 'p', '650-725-3726')
   gold:  ('subh', 'p', '650-725-6949')
('serafim', 'e', 'serafim@cs dot stanford dot .edu')
   gold:  ('serafim', 'e', 'serafim@cs.stanford.edu')
   gold:  ('serafim', 'p', '650-723-3334')
   gold:  ('serafim', 'p', '650-725-1449')
('levoy', 'e', 'ada@graphics.stanford..edu')
   gold:  ('levoy', 'e', 'ada@graphics.stanford.edu')
   gold:  ('levoy', 'e', 'melissa@graphics.stanford.edu')
   gold:  ('levoy', 'p', '650-723-0033')
   gold:  ('levoy', 'p', '650-724-6865')
   gold:  ('levoy', 'p', '650-725-3724')
   gold:  ('levoy', 'p', '650-725-4089')
('latombe', 'e', 'latombe<del>@cs.stanford.edu')
   gold:  ('latombe', 'e', 'asandra@cs.stanford.edu')
   gold:  ('latombe', 'e', 'latombe@cs.stanford.edu')
   gold:  ('latombe', 'e', 'liliana@cs.stanford.edu')
   gold:  ('latombe', 'p', '650-721-6625')
   gold:  ('latombe', 'p', '650-723-0350')
   gold:  ('latombe', 'p', '650-723-4137')
   gold:  ('latombe', 'p', '650-725-1449')
('jurafsky', 'e', "('stanford.edu'@'jurafsky').edu")
   gold:  ('jurafsky', 'e', 'jurafsky@stanford.edu')
   gold:  ('jurafsky', 'p', '650-723-5666')
('jks', 'e', 'jks@robotics;stanford;.edu')
   gold:  ('jks', 'e', 'jks@robotics.stanford.edu')
('manning', 'e', 'manning@cs.stanford..edu')
   gold:  ('manning', 'e', 'dbarros@cs.stanford.edu')
   gold:  ('manning', 'e', 'manning@cs.stanford.edu')
   gold:  ('manning', 'p', '650-723-7683')
   gold:  ('manning', 'p', '650-725-1449')
   gold:  ('manning', 'p', '650-725-3358')
('latombe', 'e', 'asandra<del>@cs.stanford.edu')
   gold:  ('latombe', 'e', 'asandra@cs.stanford.edu')
   gold:  ('latombe', 'e', 'latombe@cs.stanford.edu')
```



```
   gold:  ('latombe', 'e', 'liliana@cs.stanford.edu')
   gold:  ('latombe', 'p', '650-721-6625')
   gold:  ('latombe', 'p', '650-723-0350')
   gold:  ('latombe', 'p', '650-723-4137')
   gold:  ('latombe', 'p', '650-725-1449')
('plotkin', 'e', 'server@infolab.stanford.edu')
('ullman', 'e', 'support@gradiance.edu')
   gold:  ('ullman', 'e', 'support@gradiance.com')
   gold:  ('ullman', 'e', 'ullman@cs.stanford.edu')
   gold:  ('ullman', 'p', '650-494-8016')
   gold:  ('ullman', 'p', '650-725-2588')
   gold:  ('ullman', 'p', '650-725-4802')
('subh', 'e', 'subh@stanford dot .edu')
   gold:  ('subh', 'e', 'subh@stanford.edu')
   gold:  ('subh', 'e', 'uma@cs.stanford.edu')
   gold:  ('subh', 'p', '650-724-1915')
   gold:  ('subh', 'p', '650-725-3726')
   gold:  ('subh', 'p', '650-725-6949')
False Negatives (20):
{('dlwh', 'e', 'dlwh@stanford.edu'),
 ('engler', 'e', 'engler@stanford.edu'),
 ('hager', 'e', 'hager@cs.jhu.edu'),
 ('jks', 'e', 'jks@robotics.stanford.edu'),
 ('jurafsky', 'e', 'jurafsky@stanford.edu'),
 ('latombe', 'e', 'asandra@cs.stanford.edu'),
 ('latombe', 'e', 'latombe@cs.stanford.edu'),
 ('latombe', 'e', 'liliana@cs.stanford.edu'),
 ('levoy', 'e', 'ada@graphics.stanford.edu'),
 ('levoy', 'e', 'melissa@graphics.stanford.edu'),
```



```
 ('manning', 'e', 'dbarros@cs.stanford.edu'),
 ('manning', 'e', 'manning@cs.stanford.edu'),
 ('ouster', 'e', 'ouster@cs.stanford.edu'),
 ('ouster', 'e', 'teresa.lynn@stanford.edu'),
 ('pal', 'e', 'pal@cs.stanford.edu'),
 ('serafim', 'e', 'serafim@cs.stanford.edu'),
 ('subh', 'e', 'subh@stanford.edu'),
 ('subh', 'e', 'uma@cs.stanford.edu'),
 ('ullman', 'e', 'support@gradiance.com'),
 ('vladlen', 'e', 'vladlen@stanford.edu')}
Summary: tp=97, fp=22, fn=20
```

Option 2.

A. List the examples that could not match with the current regular expressions with two extracted parts, ending in .edu. For each example, or set of examples that fit the same pattern, explain briefly why it won't work. If you can make expressions in regexpal that match, but don't work in the program to extract the email or phone numbers, list those here.

True Positives

Email patterns
There are 4 regular expressions shown in **Table 1.1** with two extracted parts ending in .edu which Rule 1,2 and 4 that covers most of the True Positives. Rule 1 covers 19 of them, Rule 2 covers 5 of them, Rule 4 covers 2 of them and Rule 3 covers just one.

1. The expression '([A-Za-z.]+)@([A-Za-z.]+)\.edu' from Rule 1 looks for two parts before and after the @ character and ending with .edu which satisfies 19 cases in True Positive Category
2. The expression '([A-Za-z.]+)\s+@\s+([A-Za-z.]+)\.edu' from Rule 2 looks for two parts before and after the @ character along with the space before and after the @ and ending with .edu which satisfies 5 more cases in True Positive Category
3. The expression '([A-Za-z.]+)@([A-Za-z.]+)\.EDU' from Rule 3 is same as Rule 1 except for .EDU which is capitalized and satisfies one more case from the True Positive Category
4. The expression '([A-Za-z]+)\s+at\s+([A-Za-z.]+).edu' from Rule 4 replaces the @ char with "at" and also looks the same pattern as Rule 2.

The above 4 rules can be merged into single expression by making \s optional and with another part (?i) in the beginning and also be introducing or clause for @ and "at" character and the expression will become complicated. In order to make it simple and easy to understand, the 4 rules discussed above are kept separated as individual rule to cover a total of 25 True Positive cases. If you add the satisfied cases from individual rules it will sum up to 27 as there are 2 duplicate cases covered by more than 1 rule.

Phone number Patterns
There are 5 regular expression patterns shown in **Table 1.3** to match all the phone numbers in all the file

1. The expression ('(\d{3})-(\d{3})-(\d{4})') matches three parts 3 digit , 3 digit and a 4 digit number separated by "-" character which meets 19 True Positive cases
2. The expression ('\((\d{3})\)\s?(\d{3})-(\d{4})') matches three parts 3 digit , 3 digit and a 4 digit number where the 1st part is enclosed in brackets and separated from 2nd part using space whereas the 2nd and 3rd part are separated by "-" character. This meets 47 True Positive cases from the file which covers majority of the phone numbers

3. The expression ('(\d{3})\s(\d{3})\s(\d{4})') matches three parts 3 digit , 3 digit and a 4 digit number separated by space which matches 2 True Positive Cases
4. The expression ('\[(\d{3})\]\s?(\d{3})-(\d{4})') matches three parts 3 digit , 3 digit and a 4 digit number where the 1st part is enclosed in square brackets and separated from 2nd part using space whereas the 2nd and 3rd part are separated by "-" character. This meets 2 True Positive cases from the file.
5. The expression ('(\d{3})\s(\d{3})-(\d{4})') matches three parts 3 digit , 3 digit and a 4 digit where the 1st part is separated by space with 2nd part and the 2nd part is separated by "-" from 3rd part. This meets 2 True Positive cases from the file.

   Note: There are no false positive cases associated with phone numbers in these files

False Positives
These 4 rules will not be able to match any of the patterns given in False Positives as shown in **Table 1.2.** Most of these cases require specific expression that meets/satisfies the criteria to identify the emails. It required 14 individual rules to cater the needs of all email patterns given in the Dev Folder.

Rule No : 1 ('([A-Za-z.]+<del>)@([A-Za-z.]+)\.edu')

This expression is like Rule 1 from True Positives except with additional characters "<del>" before @ and we need to match that pattern by specifying the characters "<del>" in the original expression. This identifies the below False Positive cases

- ('latombe', 'e', 'latombe<del>@cs.stanford.edu')
- ('latombe', 'e', 'asandra<del>@cs.stanford.edu')
- ('latombe', 'e', 'liliana<del>@cs.stanford.edu')

Rule No : 2 ('([A-Za-z]+)\s+at\s+([A-Za-z.]+).edu')

This expression is same as Rule 4 from True Positives where it was able to identify a true positive case but also resulted in identifying 2 false positive cases given below

- ('jure', 'e', 'server@cs.stanford.edu')
- ('plotkin', 'e', 'server@infolab.stanford.edu')

Rule No : 3 ('([A-Za-z]+)\s+at\s+([A-Za-z]+)\sdt\scom')

This expression is same as Rule 4 from True Positives but modified to match the pattern \sdt\s instead "." character and "com" instead "edu" to satisfy the below email address

- support at gradiance dt com

Rule No : 4 ('([A-Za-z]+)\s+at\s+[a-zA-Z-!\s<>]+([A-Za-z]+)\s[a-zA-Z-!\s<>]+\sdot\s[a-zA-Z-!\s<>]+edu')

This expression is unique to match which matches the below email address from the file vladlen

- vladlen at <!-- die!--> stanford <!-- spam pigs!--> dot <!-- die!--> edu</div>

Rule No : 5 ('([A-Za-z]+)\s+at\s+([A-Za-z]+\s+dot\s[A-Za-z]+\s+dot\s)edu')

This expression matches characters “dot” instead dot and “at” instead @, along with the spaces before and after these characters. This helps to pickup the below emails from the files seraphim, hager and subh

- ('serafim', 'e', 'serafim@cs dot stanford dot .edu')
- ('hager', 'e', 'hager@cs dot jhu dot .edu')
- ('subh', 'e', 'uma@cs dot stanford dot .edu')

Rule No : 6 ('([A-Za-z]+)\s+AT\s+([A-Za-z]+\s+DOT\s)edu')

This expression matches “AT” instead @ and “DOT” instead “.” and with spaces before and after these characters. This satisfies the criteria defined in the file subh

- subh AT stanford DOT edu

Rule No : 7 ('([A-Za-z]+)\s+at\s+([A-Za-z]+;[A-Za-z]+;)edu')

This expression matches “at” for @ and “;” for “.” and spaces before and after “at” which satisfies the matching criteria for the email defined in jks

- ('jks', 'e', 'jks@robotics;stanford;.edu')

Rule No : 8 ('([A-Za-z]+)\s+at\s+([A-Za-z]+\s[A-Za-z]+\s)edu')

This expression matches “at” for @ and “\s” for “.” to satisfy the criteria for picking up the below email mentioned in file pal

- ('pal', 'e', 'pal@cs stanford .edu')

Rule No: 9 ('([A-Za-z.-]+)@([A-Za-z.-]+)\.-e-d-u')

This expression is unique where every character in the email address is separated by “-” character defined here in the file dlwh

- ('dlwh', 'e', 'd-l-w-h-@-s-t-a-n-f-o-r-d-.edu')

Rule No: 10 ('(\(\'[A-Za-z.]+edu\'),(\'[A-Za-z.]+\'\))')

This expression is also unique where the username and domain name are separated by comma and wrapped around quote character and the full address is enclosed with brackets and one more catch where username follows domain name instead of usual expression where domain name follows. This will exactly satisfy the below email address requirement from jurafsky

- ('stanford.edu','jurafsky')

Rule No: 11 ('([A-Za-z]+)&#x40;([A-Za-z]+.[A-Za-z.]+)edu')

This expression is using a combination of few characters “&#x40;” instead of @ and seems to match the html code for @ and will be able to meet the requirement for the following email address in levoy

- ada&#x40;graphics.stanford.edu
- melissa&#x40;graphics.stanford.edu

Rule No: 12 ('([A-Za-z]+)\s<at symbol>\s([A-Za-z.]+)edu')

This expression uses “\s<at symbol>\s” instead @ and helps to pick the email from manning

- manning <at symbol> cs.stanford.edu
- dbarros <at symbol> cs.stanford.edu

Rule No : 13 ('([A-Za-z]+)\sWHERE\s([A-Za-z\s]+)DOM\sedu')

This expression uses “\sWHERE\s” for @ and “DOM” for “.” to pick the email address from engler

- engler WHERE stanford DOM edu

Rule No : 14 ('([A-Za-z.]+)\s[a-z\s&";\(]+@([A-Za-z.]+)edu[a-z\s&;"\)]+')

This expression is unique to match the address ouster (followed by &ldquo;@cs.stanford.edu&rdquo;) and teresa.lynn (followed by "@stanford.edu") from ouster

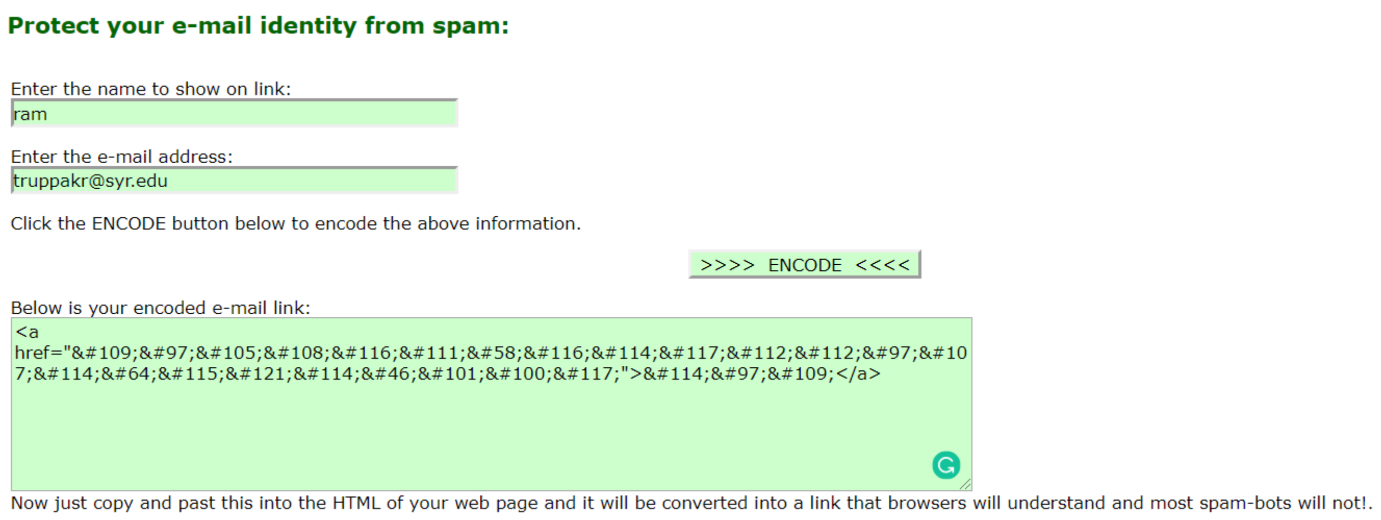
B. Search the web and find a couple of additional examples of obscured email addresses or phone numbers and report on them or try to design a way to obscure an email address that would be extremely difficult for a spamlord to match with a regular expression. For the latter, try to have something more specific than things like “To send me email, try the simplest address that makes sense.”

Online obfuscation tool

This javascript program converts normal e-mail addresses into code that most, if not all spam-bots (robots that run programs that harvest e-mail addresses) were not decoding in 2005. The javascript program to obfuscate an e-mail address is simple to use, just fill out two fields on the form, click the ENCODE button and copy the results into the HTML of a page on your site where you want your e-mail address to be used.

You enter the name you want to show as the link (for example: BobSmith). Then enter his e-mail address (for example: bobsmith@microsoft.com). Then click on the encode button to receive the encoded HTML that you cut and paste into your web site.

https://www.hcidata.info/obfuscate-email-address.htm



Protect your e-mail identity from spam:

Enter the name to show on link:
ram

Enter the e-mail address:
truppakr@syr.edu

Click the ENCODE button below to encode the above information.

Below is your encoded e-mail link:

```
<a
href="&#109;&#97;&#105;&#108;&#116;&#111;&#58;&#116;&#114;&#117;&#112;&#112;&#97;&#10
7;&#114;&#64;&#115;&#121;&#114;&#46;&#101;&#100;&#117;">&#114;&#97;&#109;</a>
```

Now just copy and past this into the HTML of your web page and it will be converted into a link that browsers will understand and most spam-bots will not!.

Other manual obfuscation method

The email address which are difficult for the spam lord to match with a regular expression are as follows

1. This email address will be difficult for spam lord to identify where the 1st part is "truppakr" and the second part is "syr" where both separated by @ and the second part ends with a "." and the 1st three letters from "education".
2. please find the email address as t as tom, r as romeo, u as umbrella, p as parrot twice, a as alpha, k as king, r as romeo @ s as sam, y as yoke and r as romeo dot edu
3. t^ r\\ u! p[p] a(k) r- _@_ s: y; r' ./ e? d# u@
4. Reverse the string for email address ude.rys@rkappurt
5. <truppakr(ignore[a-zA-Z0-9\w\s])@(ignore[a-zA-Z0-9\w\s])syr(ignore[a-zA-Z0-9\w\s]).edu>
6. (t r u p p a k r @ s y r . e d u)
7. For phone number , please type the following in telephone key pad <(mj+)paj-w+gm>