**Lab 4 Submission**

**Data Wrangler Script and Screenshot: CMSC**

from wrangler import dw

import sys

if(len(sys.argv) < 3):

sys.exit('Error: Please include an input and output file. Example python script.py input.csv output.csv')

w = dw.DataWrangler()

# Split data repeatedly on newline into rows

w.add(dw.Split(column=["data"],

table=0,

status="active",

drop=True,

result="row",

update=False,

insert\_position="right",

row=None,

on="\n",

before=None,

after=None,

ignore\_between=None,

which=1,

max=0,

positions=None,

quote\_character=None))

# Wrap rows where data starts with 'CMSC'

w.add(dw.Wrap(column=[],

table=0,

status="active",

drop=False,

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.StartsWith(column=[],

table=0,

status="active",

drop=False,

lcol="data",

value="CMSC",

op\_str="starts with")])))

# Extract from wrap3 between 'Waitlist: ' and ')'

w.add(dw.Extract(column=["wrap3"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before="\\)",

after="Waitlist: ",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from wrap3 between 'Open: ' and ','

w.add(dw.Extract(column=["wrap3"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=",",

after="Open: ",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from wrap3 between ': ' and ','

w.add(dw.Extract(column=["wrap3"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=",",

after=": ",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Drop wrap3

w.add(dw.Drop(column=["wrap3"],

table=0,

status="active",

drop=True))

# Extract from wrap4 after ' '

w.add(dw.Extract(column=["wrap4"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=None,

after=" ",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from wrap4 on ' any word '

w.add(dw.Extract(column=["wrap4"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on="[a-zA-Z]+",

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None))

# Drop wrap4

w.add(dw.Drop(column=["wrap4"],

table=0,

status="active",

drop=True))

# Extract from wrap8 between 'Waitlist: ' and ')'

w.add(dw.Extract(column=["wrap8"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before="\\)",

after="Waitlist: ",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from wrap8 between 'Open: ' and ','

w.add(dw.Extract(column=["wrap8"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=",",

after="Open: ",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from wrap8 on ' any number '

w.add(dw.Extract(column=["wrap8"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on="\\d+",

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None))

# Drop wrap8

w.add(dw.Drop(column=["wrap8"],

table=0,

status="active",

drop=True))

# Extract from wrap9 after ' '

w.add(dw.Extract(column=["wrap9"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=None,

after=" ",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from wrap9 before ' '

w.add(dw.Extract(column=["wrap9"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=" ",

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None))

# Drop wrap9

w.add(dw.Drop(column=["wrap9"],

table=0,

status="active",

drop=True))

# Extract from wrap5 after ' '

w.add(dw.Extract(column=["wrap5"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=None,

after=" ",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from wrap5 on ' any word '

w.add(dw.Extract(column=["wrap5"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on="[a-zA-Z]+",

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None))

# Drop wrap5

w.add(dw.Drop(column=["wrap5"],

table=0,

status="active",

drop=True))

w.apply\_to\_file(sys.argv[1]).print\_csv(sys.argv[2])

**Data Wrangler Script and Screenshot: World Cup 1**

from wrangler import dw

import sys

if(len(sys.argv) < 3):

sys.exit('Error: Please include an input and output file. Example python script.py input.csv output.csv')

w = dw.DataWrangler()

# Split data repeatedly on newline into rows

w.add(dw.Split(column=["data"],

table=0,

status="active",

drop=True,

result="row",

update=False,

insert\_position="right",

row=None,

on="\n",

before=None,

after=None,

ignore\_between=None,

which=1,

max=0,

positions=None,

quote\_character=None))

# Cut on '"'

w.add(dw.Cut(column=[],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=None,

on="\"",

before=None,

after=None,

ignore\_between=None,

which=1,

max=0,

positions=None))

# Extract from data between 'fff any number f|' and ' '

w.add(dw.Extract(column=["data"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=" ",

after="fff\\d+f\\|",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from data between '|' and ' ('

w.add(dw.Extract(column=["data"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=" \\(",

after="\\|",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Merge extract1, extract with glue

w.add(dw.Merge(column=["extract1","extract"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

glue=""))

# Drop extract1

w.add(dw.Drop(column=["extract1"],

table=0,

status="active",

drop=True))

# Drop extract

w.add(dw.Drop(column=["extract"],

table=0,

status="active",

drop=True))

# Extract from data between '[\[' and ' '

w.add(dw.Extract(column=["data"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=" ",

after="\\[\\[",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from data between '{ any lowercase word |' and '}'

w.add(dw.Extract(column=["data"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before="}",

after="{[a-z]+\\|",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Fill extract2 with values from above

w.add(dw.Fill(column=["extract2"],

table=0,

status="active",

drop=False,

direction="down",

method="copy",

row=None))

# Delete rows where data is null

w.add(dw.Filter(column=[],

table=0,

status="active",

drop=False,

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.IsNull(column=[],

table=0,

status="active",

drop=False,

lcol="data",

value=None,

op\_str="is null")])))

# Delete rows where extract is null

w.add(dw.Filter(column=[],

table=0,

status="active",

drop=False,

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.IsNull(column=[],

table=0,

status="active",

drop=False,

lcol="extract",

value=None,

op\_str="is null")])))

# Drop data

w.add(dw.Drop(column=["data"],

table=0,

status="active",

drop=True))

# Edit merge row 2 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[1])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 1 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[0])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 12 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[11])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 8 to ' 3 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[7])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="3",

update\_method=None))

# Edit merge row 10 to ' 4 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[9])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="4",

update\_method=None))

# Edit merge row 16 to ' 2 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[15])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="2",

update\_method=None))

# Edit merge row 20 to ' 3 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[19])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="3",

update\_method=None))

# Edit merge row 24 to ' 4 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[23])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="4",

update\_method=None))

# Edit merge row 25 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[24])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 26 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[25])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 31 to ' 3 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[30])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="3",

update\_method=None))

# Edit merge row 32 to ' 4 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[31])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="4",

update\_method=None))

# Edit merge row 33 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[32])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 35 to ' 2 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[34])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="2",

update\_method=None))

# Edit merge row 38 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[37])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 43 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[42])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Fill merge with values from above

w.add(dw.Fill(column=["merge"],

table=0,

status="active",

drop=False,

direction="down",

method="copy",

row=None))

w.apply\_to\_file(sys.argv[1]).print\_csv(sys.argv[2])

**Data Wrangler Script and Screenshot: World Cup 2**

from wrangler import dw

import sys

if(len(sys.argv) < 3):

sys.exit('Error: Please include an input and output file. Example python script.py input.csv output.csv')

w = dw.DataWrangler()

# Split data repeatedly on newline into rows

w.add(dw.Split(column=["data"],

table=0,

status="active",

drop=True,

result="row",

update=False,

insert\_position="right",

row=None,

on="\n",

before=None,

after=None,

ignore\_between=None,

which=1,

max=0,

positions=None,

quote\_character=None))

# Cut on '"'

w.add(dw.Cut(column=[],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=None,

on="\"",

before=None,

after=None,

ignore\_between=None,

which=1,

max=0,

positions=None))

# Extract from data between 'fff any number f|' and ' '

w.add(dw.Extract(column=["data"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=" ",

after="fff\\d+f\\|",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from data between '|' and ' ('

w.add(dw.Extract(column=["data"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=" \\(",

after="\\|",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Merge extract1, extract with glue

w.add(dw.Merge(column=["extract1","extract"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

glue=""))

# Drop extract1

w.add(dw.Drop(column=["extract1"],

table=0,

status="active",

drop=True))

# Drop extract

w.add(dw.Drop(column=["extract"],

table=0,

status="active",

drop=True))

# Extract from data between '[\[' and ' '

w.add(dw.Extract(column=["data"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before=" ",

after="\\[\\[",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Extract from data between '{ any lowercase word |' and '}'

w.add(dw.Extract(column=["data"],

table=0,

status="active",

drop=False,

result="column",

update=False,

insert\_position="right",

row=None,

on=".\*",

before="}",

after="{[a-z]+\\|",

ignore\_between=None,

which=1,

max=1,

positions=None))

# Fill extract2 with values from above

w.add(dw.Fill(column=["extract2"],

table=0,

status="active",

drop=False,

direction="down",

method="copy",

row=None))

# Delete rows where data is null

w.add(dw.Filter(column=[],

table=0,

status="active",

drop=False,

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.IsNull(column=[],

table=0,

status="active",

drop=False,

lcol="data",

value=None,

op\_str="is null")])))

# Delete rows where extract is null

w.add(dw.Filter(column=[],

table=0,

status="active",

drop=False,

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.IsNull(column=[],

table=0,

status="active",

drop=False,

lcol="extract",

value=None,

op\_str="is null")])))

# Drop data

w.add(dw.Drop(column=["data"],

table=0,

status="active",

drop=True))

# Edit merge row 2 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[1])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 1 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[0])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 12 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[11])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 8 to ' 3 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[7])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="3",

update\_method=None))

# Edit merge row 10 to ' 4 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[9])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="4",

update\_method=None))

# Edit merge row 16 to ' 2 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[15])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="2",

update\_method=None))

# Edit merge row 20 to ' 3 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[19])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="3",

update\_method=None))

# Edit merge row 24 to ' 4 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[23])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="4",

update\_method=None))

# Edit merge row 25 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[24])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 26 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[25])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 31 to ' 3 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[30])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="3",

update\_method=None))

# Edit merge row 32 to ' 4 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[31])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="4",

update\_method=None))

# Edit merge row 33 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[32])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 35 to ' 2 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[34])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="2",

update\_method=None))

# Edit merge row 38 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[37])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Edit merge row 43 to ' 1 '

w.add(dw.Edit(column=["merge"],

table=0,

status="active",

drop=False,

result="column",

update=True,

insert\_position="right",

row=dw.Row(column=[],

table=0,

status="active",

drop=False,

conditions=[dw.RowIndex(column=[],

table=0,

status="active",

drop=False,

indices=[42])]),

on=None,

before=None,

after=None,

ignore\_between=None,

which=1,

max=1,

positions=None,

to="1",

update\_method=None))

# Fill merge with values from above

w.add(dw.Fill(column=["merge"],

table=0,

status="active",

drop=False,

direction="down",

method="copy",

row=None))

# Unfold extract on merge

w.add(dw.Unfold(column=["extract"],

table=0,

status="active",

drop=False,

measure="merge"))

w.apply\_to\_file(sys.argv[1]).print\_csv(sys.argv[2])

**UNIX Tools Command: CMSC**

cat cmsc.txt | sed 's/Seats (Total: //g'| sed 's/Open: //g'| sed 's/Waitlist: //g'| sed 's/)//g'

| sed 's/CSI /CSI, /g' | sed 's/AVW /AVW, /g' |sed 's/HBK /HBK, /g' |sed 's/JMP /JMP, /g' |sed 's/MTH /MTH, /g'

| sed 's/M /M, /g' | sed 's/MW /MW, /g' | sed 's/MWF /MWF, /g' | sed 's/TuTh /TuTh, /g' | sed 's/F /F,/g' |

awk '/^CMSC/ {print combined; combined = $0}

!/^CMSC/ {combined = combined", "$0;}

END {print combined}'

CMSC100, 0101, Charles Kassir, 45, 4, 0, M, 4:00pm - 4:50pm, CSI, 2117,

CMSC106, 0101, Jianwu Wang, 45, 0, 5, TuTh, 9:30am - 10:45am, CSI, 2117,

CMSC131, 0101, Evan Golub , 31, 0, 0, MWF, 2:00pm - 2:50pm, CSI, 1115, 0102, Evan Golub , 31, 0, 0, MWF, 2:00pm - 2:50pm, CSI, 1115, 0103, Evan Golub , 31, 1, 0, MWF, 2:00pm - 2:50pm, CSI, 1115, 0104, Evan Golub , 31, 1, 0, MWF, 2:00pm - 2:50pm, CSI, 1115, 0201, Evan Golub , 31, 1, 0, MWF, 3:00pm - 3:50pm, CSI, 1115, 0202, Evan Golub , 31, 1, 0, MWF, 3:00pm - 3:50pm, CSI, 1115, 0203, Evan Golub , 31, 0, 0, MWF, 3:00pm - 3:50pm, CSI, 1115, 0204, Evan Golub , 31, 0, 1, MWF, 3:00pm - 3:50pm, CSI, 1115, 0301, Thomas Reinhardt, 31, 0, 0, MWF, 11:00am - 11:50am, CSI, 1115, 0302, Thomas Reinhardt, 31, 0, 0, MWF, 11:00am - 11:50am, CSI, 1115, 0303, Thomas Reinhardt, 31, 0, 0, MWF, 11:00am - 11:50am, CSI, 1115, 0304, Thomas Reinhardt, 31, 0, 0, MWF, 11:00am - 11:50am, CSI, 1115, 0401, Thomas Reinhardt, 31, 0, 0, MWF, 12:00pm - 12:50pm, CSI, 1115, 0402, Thomas Reinhardt, 31, 7, 0, MWF, 12:00pm - 12:50pm, CSI, 1115, 0403, Thomas Reinhardt, 31, 0, 0, MWF, 12:00pm - 12:50pm, CSI, 1115, 0404, Thomas Reinhardt, 31, 7, 0, MWF, 12:00pm - 12:50pm, CSI, 1115,

CMSC132, 0101, Laurence Herman , 34, 0, 2, MWF, 10:00am - 10:50am, CSI, 1115, 0102, Laurence Herman , 34, 0, 0, MWF, 10:00am - 10:50am, CSI, 1115, 0103, Laurence Herman , 34, 0, 0, MWF, 10:00am - 10:50am, CSI, 1115, 0104, Laurence Herman , 34, 0, 2, MWF, 10:00am - 10:50am, CSI, 1115, 0201, Laurence Herman , 34, 6, 0, MWF, 1:00pm - 1:50pm, CSI, 1115, 0202, Laurence Herman , 34, 1, 0, MWF, 1:00pm - 1:50pm, CSI, 1115, 0203, Laurence Herman , 34, 0, 0, MWF, 1:00pm - 1:50pm, CSI, 1115, 0204, Laurence Herman , 34, 0, 0, MWF, 1:00pm - 1:50pm, CSI, 1115, 0301, Laurence Herman , 30, 3, 0, MWF, 2:00pm - 2:50pm, CSI, 2117, 0302, Laurence Herman , 29, 0, 1, MWF, 2:00pm - 2:50pm, CSI, 2117, 0303, Laurence Herman , 29, 8, 0, MWF, 2:00pm - 2:50pm, CSI, 2117,

CMSC216, 0101, Nelson Padua-Perez , 28, 2, 0, TuTh, 9:30am - 10:45am, CSI, 1115, 0102, Nelson Padua-Perez , 28, 0, 0, TuTh, 9:30am - 10:45am, CSI, 1115, 0103, Nelson Padua-Perez , 28, 0, 1, TuTh, 9:30am - 10:45am, CSI, 1115, 0104, Nelson Padua-Perez , 28, 0, 0, TuTh, 9:30am - 10:45am, CSI, 1115, 0201, Nelson Padua-Perez , 28, 0, 0, TuTh, 11:00am - 12:15pm, CSI, 1115, 0202, Nelson Padua-Perez , 28, 0, 1, TuTh, 11:00am - 12:15pm, CSI, 1115, 0203, Nelson Padua-Perez , 28, 0, 1, TuTh, 11:00am - 12:15pm, CSI, 1115, 0204, Nelson Padua-Perez , 28, 0, 0, TuTh, 11:00am - 12:15pm, CSI, 1115, 0301, Nelson Padua-Perez , 28, 0, 0, TuTh, 2:00pm - 3:15pm, CSI, 1115, 0302, Nelson Padua-Perez , 28, 2, 0, TuTh, 2:00pm - 3:15pm, CSI, 1115, 0303, Nelson Padua-Perez , 28, 0, 0, TuTh, 2:00pm - 3:15pm, CSI, 1115, 0304, Nelson Padua-Perez , 28, 0, 0, TuTh, 2:00pm - 3:15pm, CSI, 1115,

CMSC250, 0101, Clyde Kruskal , 29, 0, 4, TuTh, 2:00pm - 3:15pm, CSI, 2117, 0102, Clyde Kruskal , 29, 0, 5, TuTh, 2:00pm - 3:15pm, CSI, 2117, 0103, Clyde Kruskal , 29, 0, 3, TuTh, 2:00pm - 3:15pm, CSI, 2117, 0201, Clyde Kruskal , 29, 0, 5, TuTh, 3:30pm - 4:45pm, CSI, 1115, 0202, Clyde Kruskal , 29, 0, 3, TuTh, 3:30pm - 4:45pm, CSI, 1115, 0203, Clyde Kruskal , 29, 0, 3, TuTh, 3:30pm - 4:45pm, CSI, 1115, 0204, Clyde Kruskal , 29, 14, 0, TuTh, 3:30pm - 4:45pm, CSI, 1115, 0301, Thomas Reinhardt, 29, 0, 1, TuTh, 11:00am - 12:15pm, CSI, 3117, 0302, Thomas Reinhardt, 29, 1, 0, TuTh, 11:00am - 12:15pm, CSI, 3117, 0303, Thomas Reinhardt, 29, 0, 5, TuTh, 11:00am - 12:15pm, CSI, 3117,

CMSC250, 0101, Thomas Goldstein, 25, 0, 0, TuTh, 12:30pm - 1:45pm, CSI, 3118,

CMSC289, 0101, James Reggia , 60, 0, 36, TuTh, 11:00am - 12:15pm, CSI, 2117,

CMSC330, 0101, Chau-Wen Tseng , 27, 0, 0, TuTh, 3:30pm - 4:45pm, CSI, 3117, 0102, Chau-Wen Tseng , 27, 0, 0, TuTh, 3:30pm - 4:45pm, CSI, 3117, 0103, Chau-Wen Tseng , 27, 1, 0, TuTh, 3:30pm - 4:45pm, CSI, 3117, 0201, Chau-Wen Tseng , 27, 0, 0, TuTh, 12:30pm - 1:45pm, CSI, 3117, 0202, Chau-Wen Tseng , 27, 0, 1, TuTh, 12:30pm - 1:45pm, CSI, 3117, 0203, Chau-Wen Tseng , 27, 13, 0, TuTh, 12:30pm - 1:45pm, CSI, 3117, 0301, Chau-Wen Tseng , 27, 0, 0, TuTh, 2:00pm - 3:15pm, CSI, 3117, 0302, Chau-Wen Tseng , 27, 0, 0, TuTh, 2:00pm - 3:15pm, CSI, 3117, 0303, Chau-Wen Tseng , 27, 1, 0, TuTh, 2:00pm - 3:15pm, CSI, 3117,

CMSC351, 0101, Hamid Mahini, 88, 0, 2, MWF, 10:00am - 10:50am, CSI, 3117, 0201, Hamid Mahini, 88, 0, 4, MWF, 11:00am - 11:50am, CSI, 3117, 0301, Hamid Mahini, 88, 0, 0, MWF, 3:00pm - 3:50pm, CSI, 2117,

CMSC396, 0101, Atif Memon , Neil Spring, 25, 0, 0, W 1:00pm - 1:50pm, AVW, 4172,

CMSC411, 0101, Michelle Hugue , 45, 0, 3, TuTh, 3:30pm - 4:45pm, CSI, 1122, 0201, Michelle Hugue , 45, 0, 3, TuTh, 2:00pm - 3:15pm, CSI, 1122,

CMSC412, 0101, Neil Spring, 25, 0, 4, TuTh, 11:00am - 12:15pm, CSI, 1122, 0102, Neil Spring, 25, 0, 3, TuTh, 11:00am - 12:15pm, CSI, 1122,

CMSC414, 0101, A.U. Shankar , 40, 0, 6, TuTh, 12:30pm - 1:45pm, CSI, 1122, 0201, Elaine Shi, 50, 1, 0, MW, 3:30pm - 4:45pm, CSI, 1122,

CMSC417, 0101, Ashok Agrawala , 40, 5, 0, TuTh, 11:00am - 12:15pm, CSI, 3120, 0201, Instructor: TBA, 40, 27, 0, TuTh, 3:30pm - 4:45pm, CSI, 2107,

CMSC420, 0101, Hanan Samet , 40, 0, 0, TuTh, 12:30pm - 1:45pm, CSI, 2120, 0201, Michelle Hugue , 40, 0, 10, MW, 3:30pm - 4:45pm, CSI, 3120,

CMSC421, 0101, Donald Perlis , 50, 0, 2, TuTh, 2:00pm - 3:15pm, CSI, 1121, 0201, Donald Perlis , 46, 0, 3, TuTh, 9:30am - 10:45am, CSI, 3117,

CMSC423, 0101, Hector Corrada Bravo, 48, 20, 0, MW, 3:30pm - 4:45pm, CSI, 1121,

CMSC424, 0101, Nicholas Roussopoulos , 40, 0, 2, TuTh, 2:00pm - 3:15pm, CSI, 3120,

CMSC426, 0101, John Aloimonos, 40, 7, 0, TuTh, 11:00am - 12:15pm, CSI, 2120,

CMSC427, 0101, Zia Khan, 40, 2, 0, MW, 2:00pm - 3:15pm, CSI, 1121,

CMSC430, 0101, Jeffrey Foster , 40, 1, 0, MW, 3:30pm - 4:45pm, CSI, 2107,

CMSC433, 0101, Michael Hicks , 50, 0, 15, TuTh, 3:30pm - 4:45pm, CSI, 1121,

CMSC434, 0101, Jon Froehlich, 44, 0, 8, TuTh, 9:30am - 10:45am, CSI, 1122, 0201, Vibha Sazawal, 44, 0, 6, TuTh, 11:00am - 12:15pm, CSI, 1121,

CMSC435, 0101, James Purtilo , 50, 0, 4, TuTh, 12:30pm - 1:45pm, CSI, 1121,

CMSC436, 0101, Adam Porter , 50, 0, 14, TuTh, 9:30am - 10:45am, CSI, 1121, 0201, Atif Memon , 60, 0, 11, TuTh, 12:30pm - 1:45pm, CSI, 2117,

CMSC451, 0101, Aravind Srinivasan, 40, 0, 1, TuTh, 12:30pm - 1:45pm, CSI, 3120,

CMSC452, 0101, William Gasarch , 32, 11, 0, TuTh, 2:00pm - 3:15pm, CSI, 3118,

CMSC456, 0101, Jonathan Katz, 40, 0, 3, MWF, 12:00pm - 12:50pm, CSI, 1121,

CMSC460, 0101, Harland Glaz , 33, 0, 2, TuTh, 12:30pm - 1:45pm, MTH, 0304, 0201, Changhui Tan, 25, 2, 0, TuTh, 2:00pm - 3:15pm, MTH, 0409,

CMSC466, 0101, Maria Cameron, 25, 11, 0, MWF, 11:00am - 11:50am, MTH, 1311,

CMSC474, 0101, Dana Nau , 40, 9, 0, TuTh, 3:30pm - 4:45pm, CSI, 2120,

CMSC498, 0101, Uzi Vishkin , 10, 7, 0, MW, 11:00am - 12:15pm, ITV 1100,

CMSC498, 0101, Amol Deshpande, 40, 8, 0, TuTh, 11:00am - 12:15pm, CSI, 2107,

CMSC498, 0101, Marshini Chetty , 10, 0, 4, Th 6:00pm - 8:45pm, HBK, 0123,

CMSC631, 0101, David Van Horn, 20, 3, 0, TuTh, 3:30pm - 4:45pm, CSI, 3118,

CMSC660, 0101, Ramani Duraiswami, 25, 0, 3, TuTh, 3:30pm - 4:45pm, CSI, 3120,

CMSC663, 0101, Howard Elman, 20, 15, 0, TuTh, 5:00pm - 6:15pm, CSI, 4122,

CMSC666, 0101, Tobias von Petersdorff , 25, 8, 0, TuTh, 2:00pm - 3:15pm, CSI, 4122,

CMSC701, 0101, Mihai Pop, 40, 9, 0, TuTh, 9:30am - 10:45am, CSI, 3120,

CMSC712, 0101, A.U. Shankar , 20, 0, 0, TuTh, 2:00pm - 3:30pm, AVW, 3258,

CMSC723, 0101, Hal Daume, 40, 0, 10, TuTh, 2:00pm - 3:15pm, CSI, 2120,

CMSC733, 0101, David Jacobs , 32, 0, 17, TuTh, 11:00am - 12:15pm, CSI, 3118,

CMSC737, 0101, Atif Memon , 20, 0, 0, TuTh, 9:30am - 10:45am, CSI, 2118,

CMSC754, 0101, David Mount , 32, 0, 15, TuTh, 12:30pm - 1:45pm, CSI, 2118,

CMSC798, 0101, Jeffrey Foster , 40, 10, 0, F,11:00am - 12:50pm, CSI, 2117,

CMSC818, 0101, Peter Keleher , 20, 1, 0, TuTh, 11:00am - 12:15pm, AVW, 3258,

CMSC818, 0101, Instructor: TBA, 145, 145, 0, MW, 4:00pm - 5:00pm, CSI, 1115,

CMSC818, 0101, Tudor Dumitras, 15, 13, 0, MW, 12:30pm - 1:45pm, JMP, 1202,

CMSC828, 0101, Larry Davis , 25, 2, 0, TuTh, 2:00pm - 3:15pm, CSI, 2118,

CMSC828, 0101, Venkatramanan Subrahmanian , 25, 12, 0, TuTh, 9:30am - 10:45am, CSI, 3118,

CMSC858, 0101, Mohammad Hajiaghayi, 20, 3, 0, TuTh, 3:30pm - 4:45pm, CSI, 2118

UNIX Tools Command: World Cup 1

cat worldcup.txt | tail -n +3 | sed 's/|style="background:#fff68f"|{{fb|//;' | sed 's/|align=center|{{sort dash}}/0/;' | sed 's/|{{fb|//;' | sed 's/}}//;' | sed 's/|-//;' | sed 's/|style="background:#fff68f"|//;'| sed 's/{{sort dash//;' | sed 's/|1||||{{sort dash}}//;' |sed 's/|1||1||//;' | sed 's/|2||1||//;' | sed 's/|4||3||1//;' |sed 's/|2||2||//;' | sed 's/|2||2||2//;' | sed 's/|5||4||3//;' |sed 's/2||||{{sort dash}}//;' | sed 's/|||||{{sort dash}}//;' | sed 's/|1||9||7//;' | sed 's/|3||1||8//;' | sed 's/|8||7||6//;' | sed 's/|5||5||5//;' | sed 's/|5||4||2//;' | sed 's/|5||2||2//;' | sed 's/|13||12||8//;' | sed 's/|11||9||7//;' | awk -F ',' '/^[A-Z][A-Z][A-Z]/ { country = $0; place = 0} !/^[A-Z][A-Z][A-Z]/ { if ($0 != "|}"&& $0 != "|" && $0 != 2 && $0 !=1){ split($0, arr, ","); place = place +1; for(i = 0; i<6; i++){ if(arr[i]!=""&& arr[i]!=0){ ind = match(arr[i], '/[0-9][0-9][0-9][0-9]/'); if(ind){print country,",", substr(arr[i], ind, 4),",", place} }}} } END {}'

BRA , 1958 , 1

BRA , 1962 , 1

BRA , 1970 , 1

BRA , 1994 , 1

BRA , 2002 , 1

BRA , 1950 , 2

BRA , 1998 , 2

BRA , 1938 , 3

BRA , 1978 , 3

BRA , 1974 , 4

BRA , 2014 , 4

GER , 1954 , 1

GER , 1974 , 1

GER , 1990 , 1

GER , 2014 , 1

GER , 1966 , 2

GER , 1982 , 2

GER , 1986 , 2

GER , 2002 , 2

GER , 1934 , 3

GER , 1970 , 3

GER , 2006 , 3

GER , 2010 , 3

GER , 1958 , 4

ITA , 1934 , 1

ITA , 1938 , 1

ITA , 1982 , 1

ITA , 2006 , 1

ITA , 1970 , 2

ITA , 1994 , 2

ITA , 1990 , 3

ITA , 1978 , 4

ARG , 1978 , 1

ARG , 1986 , 1

ARG , 1930 , 2

ARG , 1990 , 2

ARG , 2014 , 2

URU , 1930 , 1

URU , 1950 , 1

URU , 1954 , 4

URU , 1970 , 4

URU , 2010 , 4

FRA , 1998 , 1

FRA , 2006 , 2

FRA , 1958 , 3

FRA , 1986 , 3

FRA , 1982 , 4

ENG , 1966 , 1

ENG , 1990 , 4

ESP , 2010 , 1

ESP , 1950 , 4

NED , 1974 , 2

NED , 1978 , 2

NED , 2010 , 2

NED , 2014 , 3

NED , 1998 , 4

TCH , 1934 , 2

TCH , 1962 , 2

HUN , 1938 , 2

HUN , 1954 , 2

SWE , 1958 , 2

SWE , 1950 , 3

SWE , 1994 , 3

SWE , 1938 , 4

POL , 1974 , 3

POL , 1982 , 3

AUT , 1954 , 3

AUT , 1934 , 4

POR , 1966 , 3

POR , 2006 , 4

USA , 1930 , 3

CHI , 1962 , 3

CRO , 1998 , 3

TUR , 2002 , 3

YUG , 1930 , 4

YUG , 1962 , 4

URS , 1966 , 4

BEL , 1986 , 4

BUL , 1994 , 4

KOR , 2002 , 4

**Python Script: CMSC**

# Import pandas and numpy

import pandas as pd

import numpy as np

import subprocess

import re

cmsc = open('cmsc.txt', 'r')

fullLine = ""

for line in cmsc:

line = line.rstrip('\r\n')

line = line.replace("Seats (Total: ", "")

line = line.replace("Open: ", "")

line = line.replace("Waitlist: ","")

line = line.replace(")","")

#buildings

line = line.replace("CSI ", "CSI,")

line = line.replace("AVW ", "AVW,")

line = line.replace("HBK ", "HBK,")

line = line.replace("JMP ", "JMP,")

line = line.replace("MTH ", "MTH,")

#days of week

line = line.replace("M ", "M, ")

line = line.replace("MW ", "MW, ")

line = line.replace("MWF ", "MWF, ")

line = line.replace("TuTh ", "TuTh, ")

line = line.replace("F ", "F,")

#line = re.sub(r'([A-Z]{3})\s\s(\d{4})', r'([A-Z]{3}),\s(\d{4})', line)

if "CMSC" in line:

print fullLine

fullLine = line

else:

fullLine = fullLine+","+line

CMSC100,0101,Charles Kassir,45, 4, 0,M, 4:00pm - 4:50pm,CSI, 2117,

CMSC106,0101,Jianwu Wang,45, 0, 5,TuTh, 9:30am - 10:45am,CSI, 2117,

CMSC131,0101,Evan Golub ,31, 0, 0,MWF, 2:00pm - 2:50pm,CSI, 1115,0102,Evan Golub ,31, 0, 0,MWF, 2:00pm - 2:50pm,CSI, 1115,0103,Evan Golub ,31, 1, 0,MWF, 2:00pm - 2:50pm,CSI, 1115,0104,Evan Golub ,31, 1, 0,MWF, 2:00pm - 2:50pm,CSI, 1115,0201,Evan Golub ,31, 1, 0,MWF, 3:00pm - 3:50pm,CSI, 1115,0202,Evan Golub ,31, 1, 0,MWF, 3:00pm - 3:50pm,CSI, 1115,0203,Evan Golub ,31, 0, 0,MWF, 3:00pm - 3:50pm,CSI, 1115,0204,Evan Golub ,31, 0, 1,MWF, 3:00pm - 3:50pm,CSI, 1115,0301,Thomas Reinhardt,31, 0, 0,MWF, 11:00am - 11:50am,CSI, 1115,0302,Thomas Reinhardt,31, 0, 0,MWF, 11:00am - 11:50am,CSI, 1115,0303,Thomas Reinhardt,31, 0, 0,MWF, 11:00am - 11:50am,CSI, 1115,0304,Thomas Reinhardt,31, 0, 0,MWF, 11:00am - 11:50am,CSI, 1115,0401,Thomas Reinhardt,31, 0, 0,MWF, 12:00pm - 12:50pm,CSI, 1115,0402,Thomas Reinhardt,31, 7, 0,MWF, 12:00pm - 12:50pm,CSI, 1115,0403,Thomas Reinhardt,31, 0, 0,MWF, 12:00pm - 12:50pm,CSI, 1115,0404,Thomas Reinhardt,31, 7, 0,MWF, 12:00pm - 12:50pm,CSI, 1115,

CMSC132,0101,Laurence Herman ,34, 0, 2,MWF, 10:00am - 10:50am,CSI, 1115,0102,Laurence Herman ,34, 0, 0,MWF, 10:00am - 10:50am,CSI, 1115,0103,Laurence Herman ,34, 0, 0,MWF, 10:00am - 10:50am,CSI, 1115,0104,Laurence Herman ,34, 0, 2,MWF, 10:00am - 10:50am,CSI, 1115,0201,Laurence Herman ,34, 6, 0,MWF, 1:00pm - 1:50pm,CSI, 1115,0202,Laurence Herman ,34, 1, 0,MWF, 1:00pm - 1:50pm,CSI, 1115,0203,Laurence Herman ,34, 0, 0,MWF, 1:00pm - 1:50pm,CSI, 1115,0204,Laurence Herman ,34, 0, 0,MWF, 1:00pm - 1:50pm,CSI, 1115,0301,Laurence Herman ,30, 3, 0,MWF, 2:00pm - 2:50pm,CSI, 2117,0302,Laurence Herman ,29, 0, 1,MWF, 2:00pm - 2:50pm,CSI, 2117,0303,Laurence Herman ,29, 8, 0,MWF, 2:00pm - 2:50pm,CSI, 2117,

CMSC216,0101,Nelson Padua-Perez ,28, 2, 0,TuTh, 9:30am - 10:45am,CSI, 1115,0102,Nelson Padua-Perez ,28, 0, 0,TuTh, 9:30am - 10:45am,CSI, 1115,0103,Nelson Padua-Perez ,28, 0, 1,TuTh, 9:30am - 10:45am,CSI, 1115,0104,Nelson Padua-Perez ,28, 0, 0,TuTh, 9:30am - 10:45am,CSI, 1115,0201,Nelson Padua-Perez ,28, 0, 0,TuTh, 11:00am - 12:15pm,CSI, 1115,0202,Nelson Padua-Perez ,28, 0, 1,TuTh, 11:00am - 12:15pm,CSI, 1115,0203,Nelson Padua-Perez ,28, 0, 1,TuTh, 11:00am - 12:15pm,CSI, 1115,0204,Nelson Padua-Perez ,28, 0, 0,TuTh, 11:00am - 12:15pm,CSI, 1115,0301,Nelson Padua-Perez ,28, 0, 0,TuTh, 2:00pm - 3:15pm,CSI, 1115,0302,Nelson Padua-Perez ,28, 2, 0,TuTh, 2:00pm - 3:15pm,CSI, 1115,0303,Nelson Padua-Perez ,28, 0, 0,TuTh, 2:00pm - 3:15pm,CSI, 1115,0304,Nelson Padua-Perez ,28, 0, 0,TuTh, 2:00pm - 3:15pm,CSI, 1115,

CMSC250,0101,Clyde Kruskal ,29, 0, 4,TuTh, 2:00pm - 3:15pm,CSI, 2117,0102,Clyde Kruskal ,29, 0, 5,TuTh, 2:00pm - 3:15pm,CSI, 2117,0103,Clyde Kruskal ,29, 0, 3,TuTh, 2:00pm - 3:15pm,CSI, 2117,0201,Clyde Kruskal ,29, 0, 5,TuTh, 3:30pm - 4:45pm,CSI, 1115,0202,Clyde Kruskal ,29, 0, 3,TuTh, 3:30pm - 4:45pm,CSI, 1115,0203,Clyde Kruskal ,29, 0, 3,TuTh, 3:30pm - 4:45pm,CSI, 1115,0204,Clyde Kruskal ,29, 14, 0,TuTh, 3:30pm - 4:45pm,CSI, 1115,0301,Thomas Reinhardt,29, 0, 1,TuTh, 11:00am - 12:15pm,CSI, 3117,0302,Thomas Reinhardt,29, 1, 0,TuTh, 11:00am - 12:15pm,CSI, 3117,0303,Thomas Reinhardt,29, 0, 5,TuTh, 11:00am - 12:15pm,CSI, 3117,

CMSC250,0101,Thomas Goldstein,25, 0, 0,TuTh, 12:30pm - 1:45pm,CSI, 3118,

CMSC289,0101,James Reggia ,60, 0, 36,TuTh, 11:00am - 12:15pm,CSI, 2117,

CMSC330,0101,Chau-Wen Tseng ,27, 0, 0,TuTh, 3:30pm - 4:45pm,CSI, 3117,0102,Chau-Wen Tseng ,27, 0, 0,TuTh, 3:30pm - 4:45pm,CSI, 3117,0103,Chau-Wen Tseng ,27, 1, 0,TuTh, 3:30pm - 4:45pm,CSI, 3117,0201,Chau-Wen Tseng ,27, 0, 0,TuTh, 12:30pm - 1:45pm,CSI, 3117,0202,Chau-Wen Tseng ,27, 0, 1,TuTh, 12:30pm - 1:45pm,CSI, 3117,0203,Chau-Wen Tseng ,27, 13, 0,TuTh, 12:30pm - 1:45pm,CSI, 3117,0301,Chau-Wen Tseng ,27, 0, 0,TuTh, 2:00pm - 3:15pm,CSI, 3117,0302,Chau-Wen Tseng ,27, 0, 0,TuTh, 2:00pm - 3:15pm,CSI, 3117,0303,Chau-Wen Tseng ,27, 1, 0,TuTh, 2:00pm - 3:15pm,CSI, 3117,

CMSC351,0101,Hamid Mahini,88, 0, 2,MWF, 10:00am - 10:50am,CSI, 3117,0201,Hamid Mahini,88, 0, 4,MWF, 11:00am - 11:50am,CSI, 3117,0301,Hamid Mahini,88, 0, 0,MWF, 3:00pm - 3:50pm,CSI, 2117,

CMSC396,0101,Atif Memon , Neil Spring,25, 0, 0,W 1:00pm - 1:50pm,AVW, 4172,

CMSC411,0101,Michelle Hugue ,45, 0, 3,TuTh, 3:30pm - 4:45pm,CSI, 1122,0201,Michelle Hugue ,45, 0, 3,TuTh, 2:00pm - 3:15pm,CSI, 1122,

CMSC412,0101,Neil Spring,25, 0, 4,TuTh, 11:00am - 12:15pm,CSI, 1122,0102,Neil Spring,25, 0, 3,TuTh, 11:00am - 12:15pm,CSI, 1122,

CMSC414,0101,A.U. Shankar ,40, 0, 6,TuTh, 12:30pm - 1:45pm,CSI, 1122,0201,Elaine Shi,50, 1, 0,MW, 3:30pm - 4:45pm,CSI, 1122,

CMSC417,0101,Ashok Agrawala ,40, 5, 0,TuTh, 11:00am - 12:15pm,CSI, 3120,0201,Instructor: TBA,40, 27, 0,TuTh, 3:30pm - 4:45pm,CSI, 2107,

CMSC420,0101,Hanan Samet ,40, 0, 0,TuTh, 12:30pm - 1:45pm,CSI, 2120,0201,Michelle Hugue ,40, 0, 10,MW, 3:30pm - 4:45pm,CSI, 3120,

CMSC421,0101,Donald Perlis ,50, 0, 2,TuTh, 2:00pm - 3:15pm,CSI, 1121,0201,Donald Perlis ,46, 0, 3,TuTh, 9:30am - 10:45am,CSI, 3117,

CMSC423,0101,Hector Corrada Bravo,48, 20, 0,MW, 3:30pm - 4:45pm,CSI, 1121,

CMSC424,0101,Nicholas Roussopoulos ,40, 0, 2,TuTh, 2:00pm - 3:15pm,CSI, 3120,

CMSC426,0101,John Aloimonos,40, 7, 0,TuTh, 11:00am - 12:15pm,CSI, 2120,

CMSC427,0101,Zia Khan,40, 2, 0,MW, 2:00pm - 3:15pm,CSI, 1121,

CMSC430,0101,Jeffrey Foster ,40, 1, 0,MW, 3:30pm - 4:45pm,CSI, 2107,

CMSC433,0101,Michael Hicks ,50, 0, 15,TuTh, 3:30pm - 4:45pm,CSI, 1121,

CMSC434,0101,Jon Froehlich,44, 0, 8,TuTh, 9:30am - 10:45am,CSI, 1122,0201,Vibha Sazawal,44, 0, 6,TuTh, 11:00am - 12:15pm,CSI, 1121,

CMSC435,0101,James Purtilo ,50, 0, 4,TuTh, 12:30pm - 1:45pm,CSI, 1121,

CMSC436,0101,Adam Porter ,50, 0, 14,TuTh, 9:30am - 10:45am,CSI, 1121,0201,Atif Memon ,60, 0, 11,TuTh, 12:30pm - 1:45pm,CSI, 2117,

CMSC451,0101,Aravind Srinivasan,40, 0, 1,TuTh, 12:30pm - 1:45pm,CSI, 3120,

CMSC452,0101,William Gasarch ,32, 11, 0,TuTh, 2:00pm - 3:15pm,CSI, 3118,

CMSC456,0101,Jonathan Katz,40, 0, 3,MWF, 12:00pm - 12:50pm,CSI, 1121,

CMSC460,0101,Harland Glaz ,33, 0, 2,TuTh, 12:30pm - 1:45pm,MTH, 0304,0201,Changhui Tan,25, 2, 0,TuTh, 2:00pm - 3:15pm,MTH, 0409,

CMSC466,0101,Maria Cameron,25, 11, 0,MWF, 11:00am - 11:50am,MTH, 1311,

CMSC474,0101,Dana Nau ,40, 9, 0,TuTh, 3:30pm - 4:45pm,CSI, 2120,

CMSC498,0101,Uzi Vishkin ,10, 7, 0,MW, 11:00am - 12:15pm,ITV 1100,

CMSC498,0101,Amol Deshpande,40, 8, 0,TuTh, 11:00am - 12:15pm,CSI, 2107,

CMSC498,0101,Marshini Chetty ,10, 0, 4,Th 6:00pm - 8:45pm,HBK, 0123,

CMSC631,0101,David Van Horn,20, 3, 0,TuTh, 3:30pm - 4:45pm,CSI, 3118,

CMSC660,0101,Ramani Duraiswami,25, 0, 3,TuTh, 3:30pm - 4:45pm,CSI, 3120,

CMSC663,0101,Howard Elman,20, 15, 0,TuTh, 5:00pm - 6:15pm,CSI, 4122,

CMSC666,0101,Tobias von Petersdorff ,25, 8, 0,TuTh, 2:00pm - 3:15pm,CSI, 4122,

CMSC701,0101,Mihai Pop,40, 9, 0,TuTh, 9:30am - 10:45am,CSI, 3120,

CMSC712,0101,A.U. Shankar ,20, 0, 0,TuTh, 2:00pm - 3:30pm,AVW, 3258,

CMSC723,0101,Hal Daume,40, 0, 10,TuTh, 2:00pm - 3:15pm,CSI, 2120,

CMSC733,0101,David Jacobs ,32, 0, 17,TuTh, 11:00am - 12:15pm,CSI, 3118,

CMSC737,0101,Atif Memon ,20, 0, 0,TuTh, 9:30am - 10:45am,CSI, 2118,

CMSC754,0101,David Mount ,32, 0, 15,TuTh, 12:30pm - 1:45pm,CSI, 2118,

CMSC798,0101,Jeffrey Foster ,40, 10, 0,F,11:00am - 12:50pm,CSI, 2117,

CMSC818,0101,Peter Keleher ,20, 1, 0,TuTh, 11:00am - 12:15pm,AVW, 3258,

CMSC818,0101,Instructor: TBA,145, 145, 0,MW, 4:00pm - 5:00pm,CSI, 1115,

CMSC818,0101,Tudor Dumitras,15, 13, 0,MW, 12:30pm - 1:45pm,JMP, 1202,

CMSC828,0101,Larry Davis ,25, 2, 0,TuTh, 2:00pm - 3:15pm,CSI, 2118,

CMSC828,0101,Venkatramanan Subrahmanian ,25, 12, 0,TuTh, 9:30am - 10:45am,CSI, 3118,

**Python Script: World Cup 1**

# Import pandas and numpy

import pandas as pd

import numpy as np

import subprocess

import re

world = open('worldcup.txt', 'r')

fullLine = ""

count = 0

placement = 0

country = ""

for line in world:

if(count<2):

line = ""

else:

line = line.rstrip('\r\n')

line = line.replace("align=center|{{sort dash}}", "0")

line = line.replace("|style=\"background:#fff68f\"|{{fb|","")

line = line.replace("|style=\"background:#fff68f\"","")

line = line.replace("{{fb", "")

line = line.replace("}}", "")

line = line.replace("{{sort dash", "")

line = line.replace("|", "")

line = line.replace("-", "")

line = line.replace("(", "")

line = line.replace("[[", "")

match = re.search('^[A-Z]{3}', line)

#split into array

arr = line.split(",")

if match:

country = line[0:3]

placement = 0

else:

if (placement<4):

placement = placement+1

else:

placement = 1

for i in arr:

if(i != "0" and i != ""):

year1 = re.search('^\d\s\d{4}\s', i)

year2 = re.search('^\s\d{4}',i)

if year1:

print country+", "+i[2:6]+", "+str(placement)

elif year2:

print country+", "+i[1:5]+", "+str(placement)

count=count+1

BRA, 1958, 1

BRA, 1962, 1

BRA, 1970, 1

BRA, 1994, 1

BRA, 2002, 1

BRA, 1950, 2

BRA, 1998, 2

BRA, 1938, 3

BRA, 1978, 3

BRA, 1974, 4

BRA, 2014, 4

GER, 1954, 1

GER, 1974, 1

GER, 1990, 1

GER, 2014, 1

GER, 1966, 2

GER, 1982, 2

GER, 1986, 2

GER, 2002, 2

GER, 1934, 3

GER, 1970, 3

GER, 2006, 3

GER, 2010, 3

GER, 1958, 4

ITA, 1934, 1

ITA, 1938, 1

ITA, 1982, 1

ITA, 2006, 1

ITA, 1970, 2

ITA, 1994, 2

ITA, 1990, 3

ITA, 1978, 4

ARG, 1978, 1

ARG, 1986, 1

ARG, 1930, 2

ARG, 1990, 2

ARG, 2014, 2

URU, 1930, 1

URU, 1950, 1

URU, 1954, 4

URU, 1970, 4

URU, 2010, 4

FRA, 1998, 1

FRA, 2006, 2

FRA, 1958, 3

FRA, 1986, 3

FRA, 1982, 4

ENG, 1966, 1

ENG, 1990, 4

ESP, 2010, 1

ESP, 1950, 4

NED, 1974, 2

NED, 1978, 2

NED, 2010, 2

NED, 2014, 3

NED, 1998, 4

TCH, 1934, 2

TCH, 1962, 2

HUN, 1938, 2

HUN, 1954, 2

SWE, 1958, 2

SWE, 1950, 3

SWE, 1994, 3

SWE, 1938, 4

POL, 1974, 3

POL, 1982, 3

AUT, 1954, 3

AUT, 1934, 4

POR, 1966, 3

POR, 2006, 4

USA, 1930, 3

CHI, 1962, 3

CRO, 1998, 3

TUR, 2002, 3

YUG, 1930, 4

YUG, 1962, 4

URS, 1966, 4

BEL, 1986, 4

BUL, 1994, 4

KOR, 2002, 4

**Python Script: World Cup 2**

import pandas as pd

import numpy as np

piv = pd.read\_csv('output.csv')

print piv.pivot('Country', 'Year', 'Rank')

Year 1930 1934 1938 1950 1954 1958 1962 1966 1970 1974 1978 \

Country

ARG 2 NaN NaN NaN NaN NaN NaN NaN NaN NaN 1

AUT NaN 4 NaN NaN 3 NaN NaN NaN NaN NaN NaN

BEL NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

BRA NaN NaN 3 2 NaN 1 1 NaN 1 4 3

BUL NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

CHI NaN NaN NaN NaN NaN NaN 3 NaN NaN NaN NaN

CRO NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

ENG NaN NaN NaN NaN NaN NaN NaN 1 NaN NaN NaN

ESP NaN NaN NaN 4 NaN NaN NaN NaN NaN NaN NaN

FRA NaN NaN NaN NaN NaN 3 NaN NaN NaN NaN NaN

GER NaN 3 NaN NaN 1 4 NaN 2 3 1 NaN

HUN NaN NaN 2 NaN 2 NaN NaN NaN NaN NaN NaN

ITA NaN 1 1 NaN NaN NaN NaN NaN 2 NaN 4

KOR NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

NED NaN NaN NaN NaN NaN NaN NaN NaN NaN 2 2

POL NaN NaN NaN NaN NaN NaN NaN NaN NaN 3 NaN

POR NaN NaN NaN NaN NaN NaN NaN 3 NaN NaN NaN

SWE NaN NaN 4 3 NaN 2 NaN NaN NaN NaN NaN

TCH NaN 2 NaN NaN NaN NaN 2 NaN NaN NaN NaN

TUR NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

URS NaN NaN NaN NaN NaN NaN NaN 4 NaN NaN NaN

URU 1 NaN NaN 1 4 NaN NaN NaN 4 NaN NaN

USA 3 NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

YUG 4 NaN NaN NaN NaN NaN 4 NaN NaN NaN NaN

Year 1930 1934 1938 1950 1954 1958 1962 1966 1970 1974 1978 \

Country

ARG 2 NaN NaN NaN NaN NaN NaN NaN NaN NaN 1

AUT NaN 4 NaN NaN 3 NaN NaN NaN NaN NaN NaN

BEL NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

BRA NaN NaN 3 2 NaN 1 1 NaN 1 4 3

BUL NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

CHI NaN NaN NaN NaN NaN NaN 3 NaN NaN NaN NaN

CRO NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

ENG NaN NaN NaN NaN NaN NaN NaN 1 NaN NaN NaN

ESP NaN NaN NaN 4 NaN NaN NaN NaN NaN NaN NaN

FRA NaN NaN NaN NaN NaN 3 NaN NaN NaN NaN NaN

GER NaN 3 NaN NaN 1 4 NaN 2 3 1 NaN

HUN NaN NaN 2 NaN 2 NaN NaN NaN NaN NaN NaN

ITA NaN 1 1 NaN NaN NaN NaN NaN 2 NaN 4

KOR NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

NED NaN NaN NaN NaN NaN NaN NaN NaN NaN 2 2

POL NaN NaN NaN NaN NaN NaN NaN NaN NaN 3 NaN

POR NaN NaN NaN NaN NaN NaN NaN 3 NaN NaN NaN

SWE NaN NaN 4 3 NaN 2 NaN NaN NaN NaN NaN

TCH NaN 2 NaN NaN NaN NaN 2 NaN NaN NaN NaN

TUR NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

URS NaN NaN NaN NaN NaN NaN NaN 4 NaN NaN NaN

URU 1 NaN NaN 1 4 NaN NaN NaN 4 NaN NaN

USA 3 NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN

YUG 4 NaN NaN NaN NaN NaN 4 NaN NaN NaN NaN

Year 1982 1986 1990 1994 1998 2002 2006 2010 2014

Country

ARG NaN 1 2 NaN NaN NaN NaN NaN 2

AUT NaN NaN NaN NaN NaN NaN NaN NaN NaN

BEL NaN 4 NaN NaN NaN NaN NaN NaN NaN

BRA NaN NaN NaN 1 2 1 NaN NaN 4

BUL NaN NaN NaN 4 NaN NaN NaN NaN NaN

CHI NaN NaN NaN NaN NaN NaN NaN NaN NaN

CRO NaN NaN NaN NaN 3 NaN NaN NaN NaN

ENG NaN NaN 4 NaN NaN NaN NaN NaN NaN

ESP NaN NaN NaN NaN NaN NaN NaN 1 NaN

FRA 4 3 NaN NaN 1 NaN 2 NaN NaN

GER 2 2 1 NaN NaN 2 3 3 1

HUN NaN NaN NaN NaN NaN NaN NaN NaN NaN

ITA 1 NaN 3 2 NaN NaN 1 NaN NaN

KOR NaN NaN NaN NaN NaN 4 NaN NaN NaN

NED NaN NaN NaN NaN 4 NaN NaN 2 3

POL 3 NaN NaN NaN NaN NaN NaN NaN NaN

POR NaN NaN NaN NaN NaN NaN 4 NaN NaN

SWE NaN NaN NaN 3 NaN NaN NaN NaN NaN

TCH NaN NaN NaN NaN NaN NaN NaN NaN NaN

TUR NaN NaN NaN NaN NaN 3 NaN NaN NaN

URS NaN NaN NaN NaN NaN NaN NaN NaN NaN

URU NaN NaN NaN NaN NaN NaN NaN 4 NaN

USA NaN NaN NaN NaN NaN NaN NaN NaN NaN

YUG NaN NaN NaN NaN NaN NaN NaN NaN NaN

[24 rows x 20 columns]