```
#Try to guess a random 3 digit number within 4 guesses. The digits will not be repeated.
import random
int_list = list(range(10))
random.shuffle(int_list)
if(int_list[0]==0):
   res="".join(map(str,int_list[1:4]))
else:
   res = "".join(map(str,int_list[0:3]))
print(res)
cow=bull =0

def check(inp,res):
   global cow,bull
   for i in range(3):
      if (inp[i] == res[i]):
         bull +=1
      elif inp[i] in res:
         cow +=1
   if bull==3:
      print("You guessed it right. You have won!!!")
      exit(0)
   elif cow ==0 and bull==0:
      print("No match")
   else:
      print("Match Found: %d correct position, %d wrong position"%(bull,cow))


print("Guessing Game:")
for i in range(5):
   inp=list(input("Enter a 3 digit number: "))
   cow=bull=0
   check(inp,res)
print("You have lost")
```

```
#Read and Write to CSV File
# Import pandas as pd
import pandas as pd
# Import the sales.csv data: sales
sales = pd.read_csv('sales.csv')
# Print out sales
print(sales)
#write to csv files
c= {'Roll Number': [10,20,30],
```

```
    'Student Name':['Kala','Vimala','Harish'],
    'Age':[30,28,17],
    'Maths':[100,34,90],
    'Science':[39,89,29],
    'Date of Birth':['4/5/1990','9/12/1992','17/10/2003']
    }
df = pd.DataFrame(c,columns=['Roll Number','Student
Name','Age','Maths','Science','Date of Birth'])
export_csv = df.to_csv
(r'C:\Users\jasmathi\Documents\KloudOne\Python\Class4Assignment\pandaresul
t.csv', index = None, header=True)
```

```
#Two player Chess Game
import chess
import chess.svg
from IPython.display import SVG
board=chess.Board()
SVG(chess.svg.board(board=board,size=400))
#Prints the chess board
print(board)
#Loop continus while game is not over or not stale mate or not check mate
while(board.is_game_over()==False or board.is_stalemate()==False or
board.is_checkmate()== False):
    #print(board.legal_moves)
    #GEt input from user
    inp = input("\nEnter the move: ")
    #check if move is a valid move and if so make the move or else print invalid move
    if chess.Move.from_uci(inp) in board.legal_moves:
        board.push_uci(inp)
        print(board)
        if board.turn:
            print("\nWhite has to Move")
        else:
            print("\nBlack has to Move")
        #Checks for a check to the King and if so Alerts the King
        if board.is_check():
            print("\nCheck to the King")
    else:
        print(board)
        print("\nInvalid Move")

if(board.is_game_over()):
    print("The game is over")
```

```
#Checks for a check mate. then sees if white is mated or black is mated
if(board.is_checkmate()):
   if board.turn():
      print("White is mated. White has lost the game")
   else:
      print("Black is mated. Black has lost the game")
#checks for a stale mate to see if game is a draw
elif(board.is_stalemate()):
   print("The game is a draw")
```
