#String\_formatting\_examples.py

name = "Gibson L-5 CES"  
year = 1922  
cost = 16035.4  
  
# The 'old' manual way to format text with string concatenation:  
print("My guitar: " + name + ", first made in " + str(year))  
  
# A better way - using str.format():  
print("My guitar: {}, first made in {}".format(name, year))  
print("My guitar: {0}, first made in {1}".format(name, year))  
print("My {0} was first made in {1} (that's right, {1}!)".format(name, year))  
  
# Formatting currency (grouping with comma, 2 decimal places):  
print("My {} would cost ${:,.2f}".format(name, cost))  
  
# Aligning columns:  
numbers = [1, 19, 123, 456, -25]  
for number in numbers:  
 print("Number is {:>5}".format(number))  
  
# A version of the above loop using the enumerate function, useful when you want the index and value  
for i, number in enumerate(numbers):  
 print("Number {0} is {1:>5}".format(i + 1, number))  
  
print("{} {} for about ${:,.0f}!".format(year, name, cost)) # Use format to value the {}  
for number in range(0, 151, 50): #print number from 0 to 151 for every 50 numbers  
 print("{:3}".format(number))

#Radoms.py

import random  
print(random.randint(1,100))

#capitalist\_conrad.py

import random  
  
max\_increase = 0.1  
max\_decrease = 0.05  
MAX = 1000  
MIN = 0.01  
price = 10  
OUTPUT\_FILE = "capitalist\_conrad.txt"  
out\_file = open(OUTPUT\_FILE, 'w')  
# Please change to 'a' if you already have a file  
  
day = 0  
print("Starting price: ${:,.2f}".format(price), file=out\_file)  
while price >= MIN and price <= MAX:  
 change = 0  
 day += 1  
 # Random 1, 2 for 50% chance of increase or decrease  
 if random.randint(1, 2) == 1:  
 change = random.uniform(0, max\_increase)  
 else:  
 # - max\_decrease to decrease 5% to 0%  
 change = random.uniform(-max\_decrease, 0)  
 price \*= (1 + change)  
 print("On day {} price is: ${:,.2f}".format(day, price), file=out\_file)  
out\_file.close()

#Exceptions\_demo.py

try:  
 numerator = int(input("Enter the numerator: "))  
 denominator = int(input("Enter the denominator: "))  
 #Answer for question 3  
 while denominator == 0:  
 print("Invalid")  
 denominator = int(input("Enter the denominator: "))  
 fraction = numerator / denominator  
 print(fraction)  
except ValueError:  
 print("Numerator and denominator must be valid numbers!")  
except ZeroDivisionError:  
 print("Cannot divide by zero!")  
print("Finished.")  
# 1. ValueError occurs when enter words or numbers with words  
# 2. when the denominator is Zero

#Exceptions\_to\_complete.py

finished = False  
result = 0  
while not finished:  
 try:  
 result = int(input("PLease enter an integer: "))  
 finished = str  
 except ValueError:  
 print("Please enter a valid integer.")  
print("Valid result is:", result)

#files.py

# Question 1  
out\_file = open('name.txt', 'w')  
name = input("Enter a name: ")  
print(name, file=out\_file)  
out\_file.close()  
  
# Question 2  
in\_file = open("name.txt", "r")  
name = in\_file.read().strip()  
in\_file.close()  
print("Your name is", name)  
  
# Question 3  
in\_file = open("numbers.txt", "r")  
number1 = int(in\_file.readline())  
number2 = int(in\_file.readline())  
in\_file.close()  
print(number1 + number2)  
  
# Question 4  
in\_file = open("numbers.txt", "r")  
total = 0  
for i in in\_file:  
 numbers = int(i)  
 total += numbers  
in\_file.close()  
print(total)

#Password\_checker.py

MIN\_LENGTH = 2  
MAX\_LENGTH = 6  
SPECIAL\_CHARS\_REQUIRED = False

#Please change to True for special character requirements  
SPECIAL\_CHARACTERS = "!@#$%^&\*()\_-=+`~,./'[]<>?{}|\\"  
  
  
def main():  
 *"""Program to get and check a user's password."""* print("Please enter a valid password")  
 print("Your password must be between", MIN\_LENGTH, "and", MAX\_LENGTH, "characters, and contain:")  
 print("\t1 or more uppercase characters")  
 print("\t1 or more lowercase characters")  
 print("\t1 or more numbers")  
 if SPECIAL\_CHARS\_REQUIRED:  
 print("\tand 1 or more special characters: ", SPECIAL\_CHARACTERS)  
 password = input("> ")  
 while not is\_valid\_password(password):  
 print("Invalid password!")  
 password = input("> ")  
 print("Your {}-character password is valid: {}".format(len(password), password))  
  
def is\_valid\_password(password):  
 *"""Determine if the provided password is valid."""* # *TODO: if length is wrong, return False* if len(password) < MIN\_LENGTH or len(password) > MAX\_LENGTH:  
 return False  
  
 count\_lower = 0  
 count\_upper = 0  
 count\_digit = 0  
 count\_special = 0  
 for char in password:  
 # *TODO: count each kind of character (use str methods like isdigit)* if char.isdigit():  
 count\_digit += 1  
 elif char.islower():  
 count\_lower += 1  
 elif char.isupper():  
 count\_upper += 1  
 elif char in SPECIAL\_CHARACTERS:  
 count\_special += 1  
  
 # *TODO: if any of the 'normal' counts are zero, return False* if count\_lower == 0 or count\_upper == 0 or count\_digit == 0:  
 return False  
 # *TODO: if special characters are required, then check the count of those* # and return False if it's zero  
 if SPECIAL\_CHARS\_REQUIRED:  
 if count\_special == 0:  
 return False  
 # if we get here (without returning False), then the password must be valid  
 return True  
  
main()