CP1404 Practical 2

string\_formatting\_examples.py

*"""  
CP1404/CP5632 - Practical  
Various examples of using Python string formatting with the str.format() method  
Want to read more about it? https://docs.python.org/3/library/string.html#formatstrings  
"""*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))  
  
# *TODO: Use string formatting to produce the output:*# 1922 Gibson L-5 CES for about $16,035!  
print("{} {} for about ${:,.0f}!".format(year, name, cost))  
  
# *TODO: Using a for loop with the range function and string formatting,*# produce the following right-aligned output (do not use a list):  
# 0  
# 50  
# 100  
# 150  
numbers\_1 = [0, 50, 100, 150]  
for i in numbers\_1:  
 print("{:>4}".format(i))

randoms.py

# 20  
# 3  
# 3.6424564914295723  
  
# The smallest number on line 1 is 5, the greatest is 20  
  
# The smallest number on line 2 is 2, the greatest is 10, can not be 4  
  
# The smallest number on line 3 is 2.5, the greatest is 5.5  
  
import random  
print(random.uniform(1, 100))

capitalist\_conrad.py

*"""  
CP1404/CP5632 - Practical  
Capitalist Conrad wants a stock price simulator for a volatile stock.  
The price starts off at $10.00, and, at the end of every day there is  
a 50% chance it increases by 0 to 10%, and  
a 50% chance that it decreases by 0 to 5%.  
If the price rises above $1000, or falls below $0.01, the program should end.  
The price should be displayed to the nearest cent (e.g. $33.59, not $33.5918232901)  
"""*import random  
  
MAX\_INCREASE = 0.175 # 17.5%  
MAX\_DECREASE = 0.05 # 5%  
MIN\_PRICE = 1.0  
MAX\_PRICE = 100.0  
INITIAL\_PRICE = 10.0  
out\_file = "stock\_prices.txt"  
price = INITIAL\_PRICE  
  
day\_counter = 1  
  
print("Starting price : ${:,.2f}".format(price), file=out\_file)  
  
while MIN\_PRICE <= price <= MAX\_PRICE:  
 price\_change = 0  
 # generate a random integer of 1 or 2  
 # if it's 1, the price increases, otherwise it decreases  
 if random.randint(1, 2) == 1:  
 # generate a random floating-point number  
 # between 0 and MAX\_INCREASE  
 price\_change = random.uniform(0, MAX\_INCREASE)  
 else:  
 # generate a random floating-point number  
 # between negative MAX\_DECREASE and 0  
 price\_change = random.uniform(-MAX\_DECREASE, 0)  
 day\_counter = day\_counter + 1  
 price \*= (1 + price\_change)  
 print("On day {}, price is ${:,.2f}".format(day\_counter, price), file=out\_file)  
  
out\_file.close()

exceptions\_demo.py

*"""  
CP1404/CP5632 - Practical  
Answer the following questions:  
1. When will a ValueError occur?  
2. When will a ZeroDivisionError occur?  
3. Could you change the code to avoid the possibility of a ZeroDivisionError?  
"""*try:  
 numerator = int(input("Enter the numerator: "))  
 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.")  
  
# Answers: ValueError occurs when the input are anything but not valid numbers.  
  
# ZeroDivisionError occurs when the denominator is 0.

exceptions\_to\_complete.py

*"""  
CP1404/CP5632 - Practical  
Fill in the TODOs to complete the task  
"""*finished = False  
result = 0  
while not finished:  
 try:  
 result = int(input("Enter an integer: "))  
 finished = True  
  
 except ValueError:  
 print("Please enter a valid integer.")  
print("Valid result is:", result)

 files.py

# 1  
out\_file = "names.txt"  
out\_file = open("names.txt", 'w')  
name = str(input("Please enter your name:"))  
print("your name is:", name, file=out\_file)  
  
out\_file.close()  
  
# 2  
in\_file = open("numbers.txt", "r")  
number\_1 = int(in\_file.readline())  
number\_2 = int(in\_file.readline())  
result = number\_1 + number\_2  
print(result)  
  
in\_file.close()  
  
# 3  
in\_file = open("numbers.txt", "r")  
total = 0  
for line in in\_file:  
 number = int(line)  
 total += number  
print(total)  
  
in\_file.close()

password\_checker

*"""  
CP1404/CP5632 - Practical  
Password checker "skeleton" code to help you get started  
"""*MIN\_LENGTH = 2  
MAX\_LENGTH = 6  
SPECIAL\_CHARS\_REQUIRED = False  
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."""* 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 character in password:  
 if character.islower():  
 count\_lower += 1  
 elif character.isupper():  
 count\_upper += 1  
 elif character.isdigit():  
 count\_digit += 1  
 elif character in SPECIAL\_CHARACTERS():  
 count\_special += 1  
  
 if count\_digit or count\_lower or count\_upper == 0:  
 return False  
  
 if character in SPECIAL\_CHARACTERS():  
 if count\_special == 0:  
 return False  
  
 return True  
  
  
main()

ascii\_table.py

letter = str(input("Enter a letter: "))  
print("{} ".format(letter), ord(letter), "\n", "...")  
  
number = int(input("Enter a number between 33 and 127: "))  
while number < 33 or number > 127:  
 print("Only between 33 to 127!")  
 number = int(input("Enter a number between 33 and 127: "))  
  
print("{} ".format(number), chr(number),"\n", "...")

word\_generator.py

*"""  
CP1404/CP5632 - Practical  
Random word generator - based on format of words  
  
Another way to get just consonants would be to use string.ascii\_lowercase  
(all letters) and remove the vowels.  
"""*import random  
  
  
VOWELS = "aeiou"  
CONSONANTS = "bcdfghjklmnpqrstvwxyz"  
word\_format = "%#\*"  
  
word = ""  
  
letter\_random = random.choice(word\_format)  
  
for kind in letter\_random.lower():  
 if kind == "%" or "\*":  
 word += random.choice(CONSONANTS)  
 elif kind == "#":  
 word += random.choice(VOWELS)  
 else:  
 word = word  
  
print(word)