Python Programming

Homework 4

Jordan Diaz

Problem 1

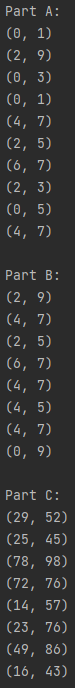
# This program tests random number generation using iterators and generators  
  
class RndSeq(object):  
 *""" Class uses the linear congruential generator to generate a sequence of random ints"""* def \_\_init\_\_(self, x0, n):  
 self.m = 2 \*\* 32  
 self.a = 22695477  
 self.c = 1  
 self.n = n  
 self.x0 = x0  
 self.loop\_forever = False  
  
 if self.n < 0:  
 self.loop\_forever = True  
  
 def \_\_iter\_\_(self):  
 *""" Iterator"""* return self  
  
 def \_\_next\_\_(self):  
 *""" Returns the next iteration"""* if self.loop\_forever:  
 self.x0 = (self.a \* self.x0 + self.c) % self.m  
 return self.x0  
 else:  
 if self.n == 0:  
 raise StopIteration  
 self.x0 = (self.a \* self.x0 + self.c) % self.m  
 self.n -= 1  
 return self.x0  
  
  
def rnd\_gen(x0, n):  
 *""" Generates a sequence of random ints"""* if n >= 0:  
 for i in range(0, n):  
 x0 = (22695477 \* x0 + 1) % 2 \*\* 32  
 yield x0  
 else:  
 while True:  
 x0 = (22695477 \* x0 + 1) % 2 \*\* 32  
 yield x0  
  
  
def main():  
 *""" Driver function"""* rnd = RndSeq(1, 10)  
 print([i for i in rnd])  
  
 print(list(rnd\_gen(1, 3)))  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Terminal Session for problem 1:

Problem 2

# This program is used to test out functional programming in python  
import itertools  
  
  
def rnd\_gen(x0, n):  
 *""" Generates a sequence of random ints"""* if n >= 0:  
 for i in range(0, n):  
 x0 = (22695477 \* x0 + 1) % 2 \*\* 32  
 yield x0  
 else:  
 while True:  
 x0 = (22695477 \* x0 + 1) % 2 \*\* 32  
 yield x0  
  
  
def gen\_rndtup(m):  
 *""" Creates an infinite sequence of tuples (a, b) where a and b are  
two random integers obtained using the rnd\_gen(1, -1) generator from Problem 2 and 0 ≤ a ≤ b < m."""* rand\_num = rnd\_gen(1, -1)  
 while True:  
 a = next(rand\_num) % m  
 b = next(rand\_num) % m  
 if a < b:  
 yield a, b  
  
  
def main():  
 *""" Main Driver Function"""* # Part A Code  
 print("Part A:")  
 tuples = gen\_rndtup(10)  
 final = itertools.islice(tuples, 0, 10)  
 for element in final:  
 print(element)  
  
 print()  
  
 # Part B Code  
 print("Part B:")  
 ans = itertools.islice(itertools.filterfalse(lambda x: x[0] + x[1] < 6, gen\_rndtup(10)), 0, 8)  
 it = iter(ans)  
 for element in it:  
 print(element)  
  
 print()  
  
 # Part C Code  
 print("Part C:")  
 ans = itertools.islice((res for res in zip(rnd\_gen(1, -1), rnd\_gen(2, -1)) if (res[0] % 101) < res[1] % 100), 0, 8)  
 for element in ans:  
 print((element[0] % 101, element[1] % 100))  
  
 print()  
  
 # Part D Code  
 print("Part D:")  
 ans = rnd\_gen(1, -1)  
 elements\_less\_than\_100 = map(lambda x: x % 100, ans)  
 elements\_divisible\_by\_13 = filter(lambda x: x % 13 == 0, elements\_less\_than\_100)  
 final\_lst = itertools.islice(elements\_divisible\_by\_13, 0, 10)  
 for element in final\_lst:  
 print(element)  
  
 print()  
  
 # Part E Code  
 print("Part E:")  
 ans = gen\_rndtup(10)  
 elements\_that\_total\_5 = filter(lambda x: (x[0] + x[1]) >= 5, ans)  
 final\_lst = itertools.islice(elements\_that\_total\_5, 0, 10)  
 for element in final\_lst:  
 print(element)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

Terminal Session for problem 2

Graphical user interface, text

Description automatically generated