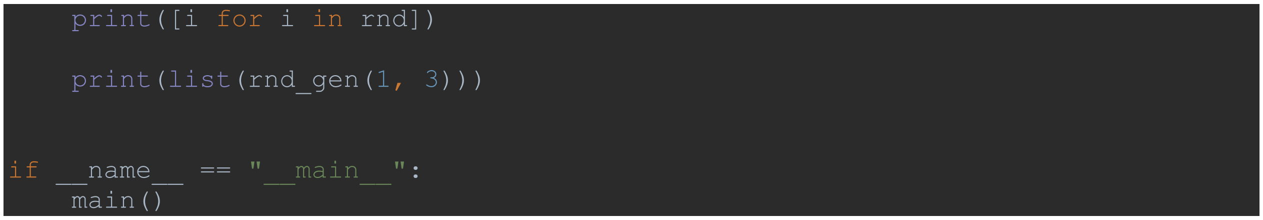
Python Programming

Homework 4

Jordan Diaz

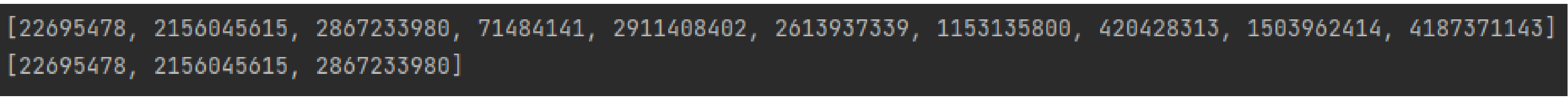
Problem 1

```
# 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()
```

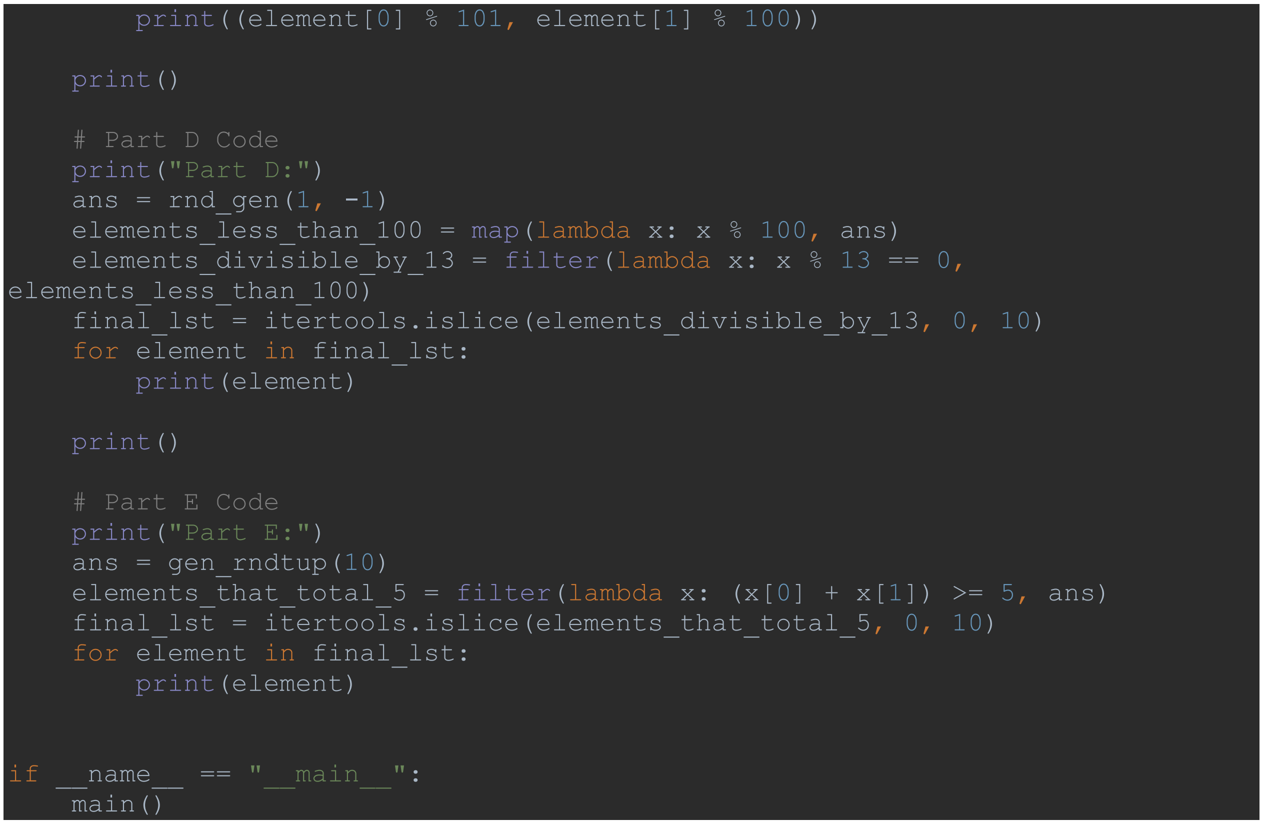
Terminal Session for problem 1:



```
[22695478, 2156045615, 2867233980, 71484141, 2911408402, 2613937339, 1153135800, 420428313, 1503962414, 4187371143]
[22695478, 2156045615, 2867233980]
```

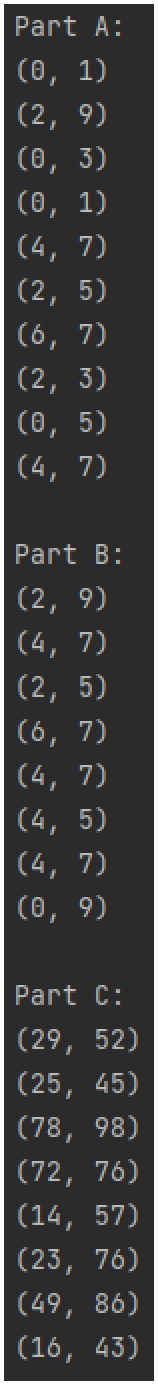
Problem 2

```
# 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:
```

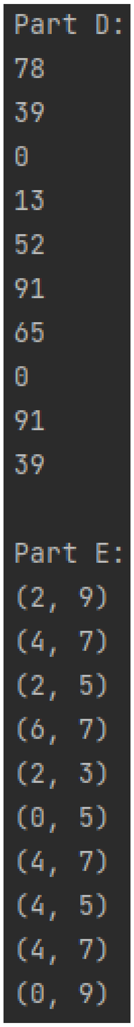


```
        print((element[0] % 101, element[1] % 100))

    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



```
Part A:
(0, 1)
(2, 9)
(0, 3)
(0, 1)
(4, 7)
(2, 5)
(6, 7)
(2, 3)
(0, 5)
(4, 7)

Part B:
(2, 9)
(4, 7)
(2, 5)
(6, 7)
(4, 7)
(4, 5)
(4, 7)
(0, 9)

Part C:
(29, 52)
(25, 45)
(78, 98)
(72, 76)
(14, 57)
(23, 76)
(49, 86)
(16, 43)
```



```
Part D:
78
39
0
13
52
91
65
0
91
39

Part E:
(2, 9)
(4, 7)
(2, 5)
(6, 7)
(2, 3)
(0, 5)
(4, 7)
(4, 5)
(4, 7)
(0, 9)
```