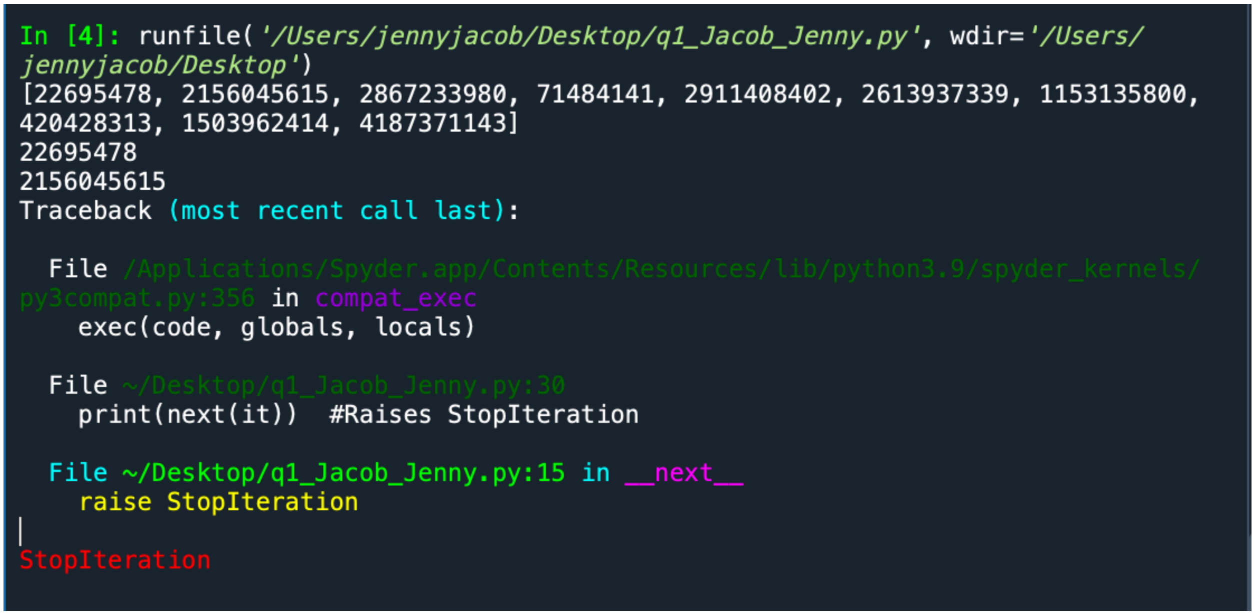
[redacted]

[redacted]

Problem 1. Random Number Generator:

Part a). Iterator Class

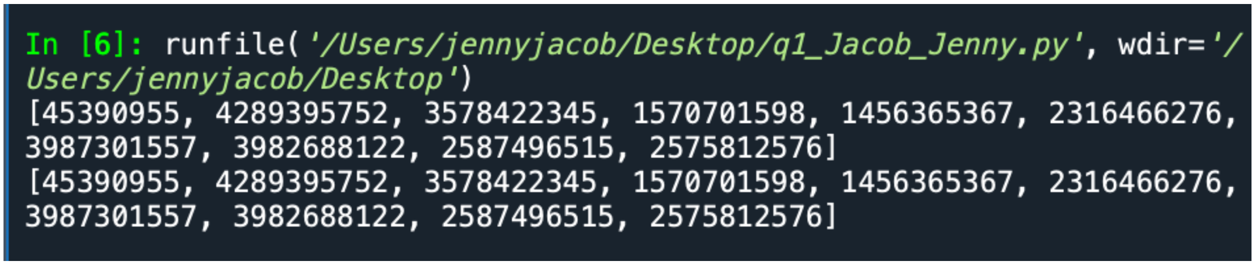
```
class RndSeq:
    def __init__(self, x0, n):
        self.x0 = x0
        self.n = n
        self.count = 0
        self.m = 2**32
        self.a = 22695477
        self.c = 1

    def __iter__(self):
        return self

    def __next__(self):
        if self.n >= 0 and self.count >= self.n:
            raise StopIteration
        else:
            self.count += 1
            self.x0 = (self.a * self.x0 + self.c) % self.m
        return self.x0



if __name__ == "__main__":
    rnd = RndSeq(1, 10)
    print([i for i in rnd])

    rnd = RndSeq(1, 2)
    it = iter(rnd)
    print(next(it))
    print(next(it))
    print(next(it))  #Raises StopIteration
```



```
In [4]: runfile('/Users/jennyjacob/Desktop/q1_Jacob_Jenny.py', wdir='/Users/
jennyjacob/Desktop')
[22695478, 2156045615, 2867233980, 71484141, 2911408402, 2613937339, 1153135800,
420428313, 1503962414, 4187371143]
22695478
2156045615
Traceback (most recent call last):

  File /Applications/Spyder.app/Contents/Resources/lib/python3.9/spyder_kernels/
py3compat.py:356 in compat_exec
    exec(code, globals, locals)

  File ~/Desktop/q1_Jacob_Jenny.py:30
    print(next(it))  #Raises StopIteration

  File ~/Desktop/q1_Jacob_Jenny.py:15 in __next__
    raise StopIteration

StopIteration
```

Part b). Generator

```
class RndSeq:
    def __init__(self, x0, n):
        self.x0 = x0
        self.n = n
        self.count = 0
        self.m = 2**32
        self.a = 22695477
        self.c = 1

    def __iter__(self):
        return self

    def __next__(self):
        if self.n >= 0 and self.count >= self.n:
            raise StopIteration
        else:
            self.count += 1
            self.x0 = (self.a * self.x0 + self.c) % self.m
            return self.x0


def rnd_gen(x0, n):
    count = 0
    m = 2**32
    a = 22695477
    c = 1

    while True:
        if n >= 0 and count >= n:
            break
        else:
            count += 1
            x0 = (a * x0 + c) % m
            yield x0


def main():
    #RndSeq class
    rnd_seq = RndSeq(2, 10)
    print([i for i in rnd_seq])

    #rnd_gen generator
    print([i for i in rnd_gen(2, 10)])


if __name__ == "__main__":
    main()
```

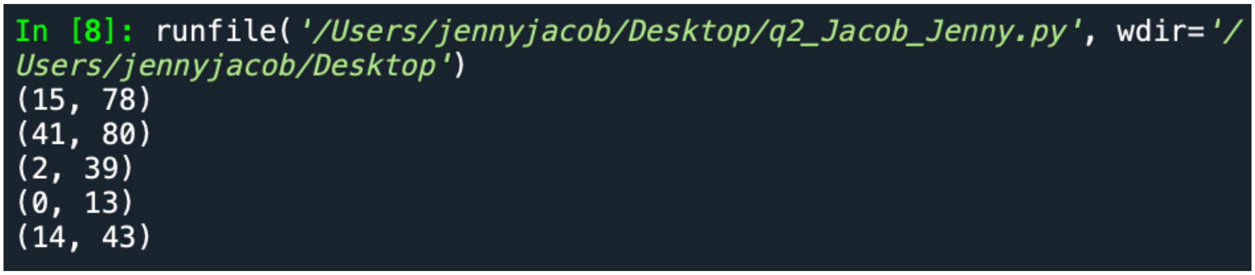


```
In [6]: runfile('/Users/jennyjacob/Desktop/q1_Jacob_Jenny.py', wdir='/
Users/jennyjacob/Desktop')
[45390955, 4289395752, 3578422345, 1570701598, 1456365367, 2316466276,
3987301557, 3982688122, 2587496515, 2575812576]
[45390955, 4289395752, 3578422345, 1570701598, 1456365367, 2316466276,
3987301557, 3982688122, 2587496515, 2575812576]
```

Problem 2. Functional Programming:

a.)

```
def rnd_gen(x0, n):
    count = 0
    m = 2**32
    a = 22695477
    c = 1

    while True:
        if n >= 0 and count >= n:
            break
        else:
            count += 1
            x0 = (a * x0 + c) % m
            yield x0

def gen_rndtup(m):
    rnd_generator = rnd_gen(1, -1)
    while True:
        a = next(rnd_generator) % m
        b = next(rnd_generator) % m
        if a > b:
            a, b = b, a
        yield (a, b)


def main():
    #gen_rndtup
    gen = gen_rndtup(100)
    for _ in range(5):
        print(next(gen))


if __name__ == "__main__":
    main()
```

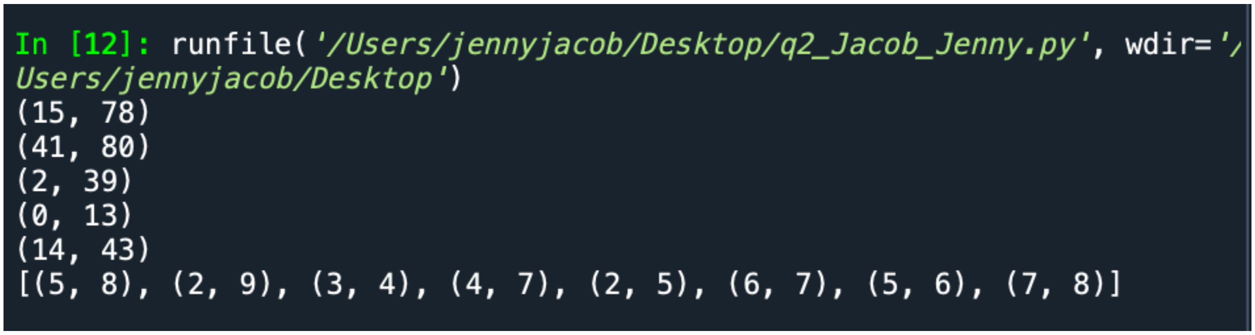


```
In [8]: runfile('/Users/jennyjacob/Desktop/q2_Jacob_Jenny.py', wdir='/
Users/jennyjacob/Desktop')
(15, 78)
(41, 80)
(2, 39)
(0, 13)
(14, 43)
```

b.)

```
from itertools import islice, filterfalse
```

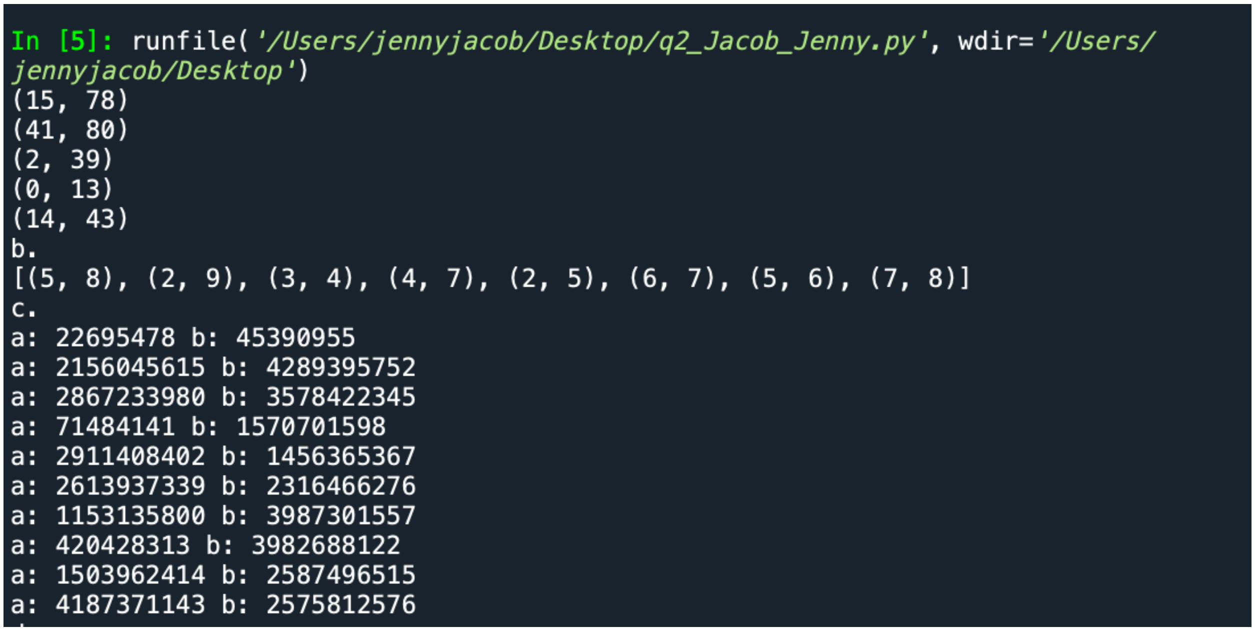
```
def rnd_gen(x0, n):
    count = 0
    m = 2**32
    a = 22695477
    c = 1

    while True:
        if n >= 0 and count >= n:
            break
        else:
            count += 1
            x0 = (a * x0 + c) % m
            yield x0

def gen_rndtup(m):
    rnd_generator = rnd_gen(1, -1)

    while True:
        a = next(rnd_generator) % m
        b = next(rnd_generator) % m
        if a > b:
            a, b = b, a
        yield (a, b)


def main():
    #use gen_rndtup
    gen = gen_rndtup(100)
    for _ in range(5):
        print(next(gen))

    #(b)
    print("b.")
    gen = gen_rndtup(10)
    filtered_gen = filterfalse(lambda tup: sum(tup) < 6, gen)
    result = list(islice(filtered_gen, 8))
    print(result)


if __name__ == "__main__":
    main()
```



```
In [12]: runfile('/Users/jennyjacob/Desktop/q2_Jacob_Jenny.py', wdir='/
Users/jennyjacob/Desktop')
(15, 78)
(41, 80)
(2, 39)
(0, 13)
(14, 43)
[(5, 8), (2, 9), (3, 4), (4, 7), (2, 5), (6, 7), (5, 6), (7, 8)]
```

c.)

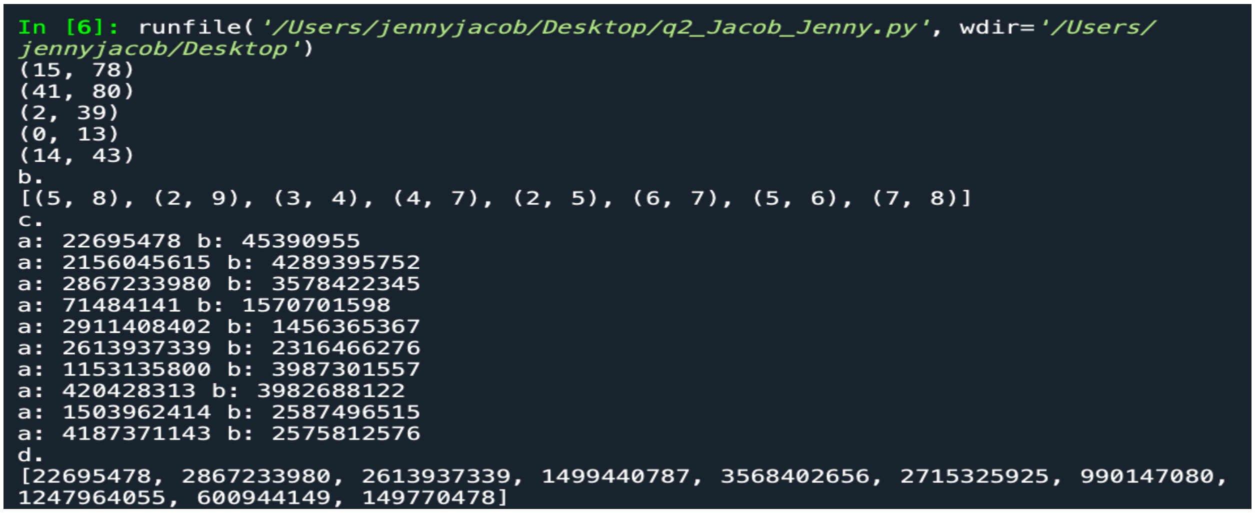
```
from itertools import islice, filterfalse
from functools import reduce


def rnd_gen(x0, n, max_value=None):
    count = 0
    m = 2**32
    a = 22695477
    c = 1

    while True:
        if max_value is not None and count >= max_value:
            break
        elif n >= 0 and count >= n:
            break
        else:
            count += 1
            x0 = (a * x0 + c) % m
            yield x0


def gen_rndtup(m):
    rnd_generator = rnd_gen(1, -1)
    while True:
        a = next(rnd_generator) % m
        b = next(rnd_generator) % m
        if a > b:
            a, b = b, a
        yield (a, b)


def main():
    #use gen_rndtup
    gen = gen_rndtup(100)
    for _ in range(5):
        print(next(gen))


    #(b)
    print("b.")
    gen = gen_rndtup(10)
    filtered_gen = filterfalse(lambda tup: sum(tup) < 6, gen)
    result = list(islice(filtered_gen, 8))
    print(result)



    #(c)
    print("c.")

    rnd_gen_a = rnd_gen(1, -1)

    rnd_gen_b = rnd_gen(2, -1)

    for _ in range(10):

        a = next(rnd_gen_a)

        b = next(rnd_gen_b)

        print("a:", a, "b:", b)

        if a <= b <= 100:

            print((a, b))


if __name__ == "__main__":

    main()
```



```
In [5]: runfile('/Users/jennyjacob/Desktop/q2_Jacob_Jenny.py', wdir='/Users/
jennyjacob/Desktop')
(15, 78)
(41, 80)
(2, 39)
(0, 13)
(14, 43)
b.
[(5, 8), (2, 9), (3, 4), (4, 7), (2, 5), (6, 7), (5, 6), (7, 8)]
c.
a: 22695478 b: 45390955
a: 2156045615 b: 4289395752
a: 2867233980 b: 3578422345
a: 71484141 b: 1570701598
a: 2911408402 b: 1456365367
a: 2613937339 b: 2316466276
a: 1153135800 b: 3987301557
a: 420428313 b: 3982688122
a: 1503962414 b: 2587496515
a: 4187371143 b: 2575812576
```

d.)

```
from itertools import islice, filterfalse
from functools import reduce


def rnd_gen(x0, n, max_value=None):
    count = 0
    m = 2**32
    a = 22695477
    c = 1

    while True:
        if max_value is not None and count >= max_value:
            break
        elif n >= 0 and count >= n:
            break
        else:
            count += 1
            x0 = (a * x0 + c) % m
            yield x0


def gen_rndtup(m):
    rnd_generator = rnd_gen(1, -1)
    while True:
        a = next(rnd_generator) % m
        b = next(rnd_generator) % m
        if a > b:
            a, b = b, a
        yield (a, b)

def main():
    #use gen_rndtup
    gen = gen_rndtup(100)
    for _ in range(5):
        print(next(gen))

    #(b)
    print("b.")
    gen = gen_rndtup(10)
    filtered_gen = filterfalse(lambda tup: sum(tup) < 6, gen)
    result = list(islice(filtered_gen, 8))
    print(result)


    #(c)
    print("c.")
    rnd_gen_a = rnd_gen(1, -1)

    rnd_gen_b = rnd_gen(2, -1)

    for _ in range(10):

        a = next(rnd_gen_a)

        b = next(rnd_gen_b)

        print("a:", a, "b:", b)

        if a <= b <= 100:

            print((a, b))


    #(d)

    print("d.")

    rnd_gen_13 = rnd_gen(1, -1)

    divisible_by_13 = filter(lambda x: x % 13 == 0, rnd_gen_13)

    first_10_numbers = islice(divisible_by_13, 10)

    print(list(first_10_numbers))


if __name__ == "__main__":

    main()
```



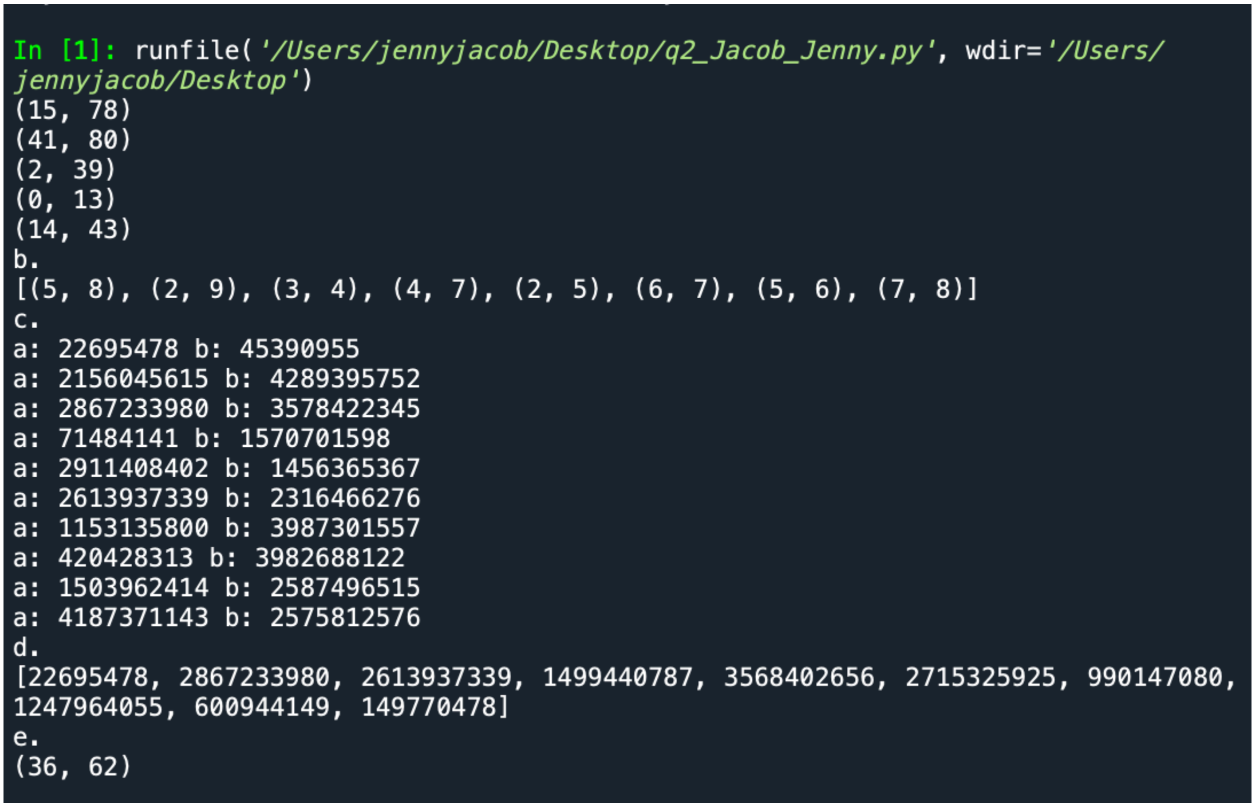
```
In [6]: runfile('/Users/jennyjacob/Desktop/q2_Jacob_Jenny.py', wdir='/Users/
jennyjacob/Desktop')
(15, 78)
(41, 80)
(2, 39)
(0, 13)
(14, 43)
b.
[(5, 8), (2, 9), (3, 4), (4, 7), (2, 5), (6, 7), (5, 6), (7, 8)]
c.
a: 22695478 b: 45390955
a: 2156045615 b: 4289395752
a: 2867233980 b: 3578422345
a: 71484141 b: 1570701598
a: 2911408402 b: 1456365367
a: 2613937339 b: 2316466276
a: 1153135800 b: 3987301557
a: 420428313 b: 3982688122
a: 1503962414 b: 2587496515
a: 4187371143 b: 2575812576
d.
[22695478, 2867233980, 2613937339, 1499440787, 3568402656, 2715325925, 990147080,
1247964055, 600944149, 149770478]
```

e.)

```
from itertools import islice, filterfalse
from functools import reduce


def rnd_gen(x0, n, max_value=None):
    count = 0
    m = 2**32
    a = 22695477
    c = 1

    while True:
        if max_value is not None and count >= max_value:
            break
        elif n >= 0 and count >= n:
            break
        else:
            count += 1
            x0 = (a * x0 + c) % m
            yield x0


def gen_rndtup(m):
    rnd_generator = rnd_gen(1, -1)
    while True:
        a = next(rnd_generator) % m
        b = next(rnd_generator) % m
        if a > b:
            a, b = b, a
        yield (a, b)

def main():
    #use gen_rndtup
    gen = gen_rndtup(100)
    for _ in range(5):
        print(next(gen))

    #(b)
    print("b.")
    gen = gen_rndtup(10)
    filtered_gen = filterfalse(lambda tup: sum(tup) < 6, gen)
    result = list(islice(filtered_gen, 8))
    print(result)


    #(c)
    print("c.")
```

```
rnd_gen_a = rnd_gen(1, -1)

rnd_gen_b = rnd_gen(2, -1)

for _ in range(10):

    a = next(rnd_gen_a)

    b = next(rnd_gen_b)

    print("a:", a, "b:", b)

    if a <= b <= 100:

        print((a, b))


#(d)

print("d.")

rnd_gen_13 = rnd_gen(1, -1)

divisible_by_13 = filter(lambda x: x % 13 == 0, rnd_gen_13)

first_10_numbers = islice(divisible_by_13, 10)

print(list(first_10_numbers))


#(e)

print("e.")

gen = gen_rndtup(10)

filtered_gen = filter(lambda tup: sum(tup) >= 5, gen)

first_10_tuples = islice(filtered_gen, 10)

result = reduce(lambda x, y: (x[0] + y[0], x[1] + y[1]), first_10_tuples)

print(result)
```

```
if __name__ == "__main__":

    main()
```



```
In [1]: runfile('/Users/jennyjacob/Desktop/q2_Jacob_Jenny.py', wdir='/Users/
jennyjacob/Desktop')
(15, 78)
(41, 80)
(2, 39)
(0, 13)
(14, 43)
b.
[(5, 8), (2, 9), (3, 4), (4, 7), (2, 5), (6, 7), (5, 6), (7, 8)]
c.
a: 22695478 b: 45390955
a: 2156045615 b: 4289395752
a: 2867233980 b: 3578422345
a: 71484141 b: 1570701598
a: 2911408402 b: 1456365367
a: 2613937339 b: 2316466276
a: 1153135800 b: 3987301557
a: 420428313 b: 3982688122
a: 1503962414 b: 2587496515
a: 4187371143 b: 2575812576
d.
[22695478, 2867233980, 2613937339, 1499440787, 3568402656, 2715325925, 990147080,
1247964055, 600944149, 149770478]
e.
(36, 62)
```