Homework 5

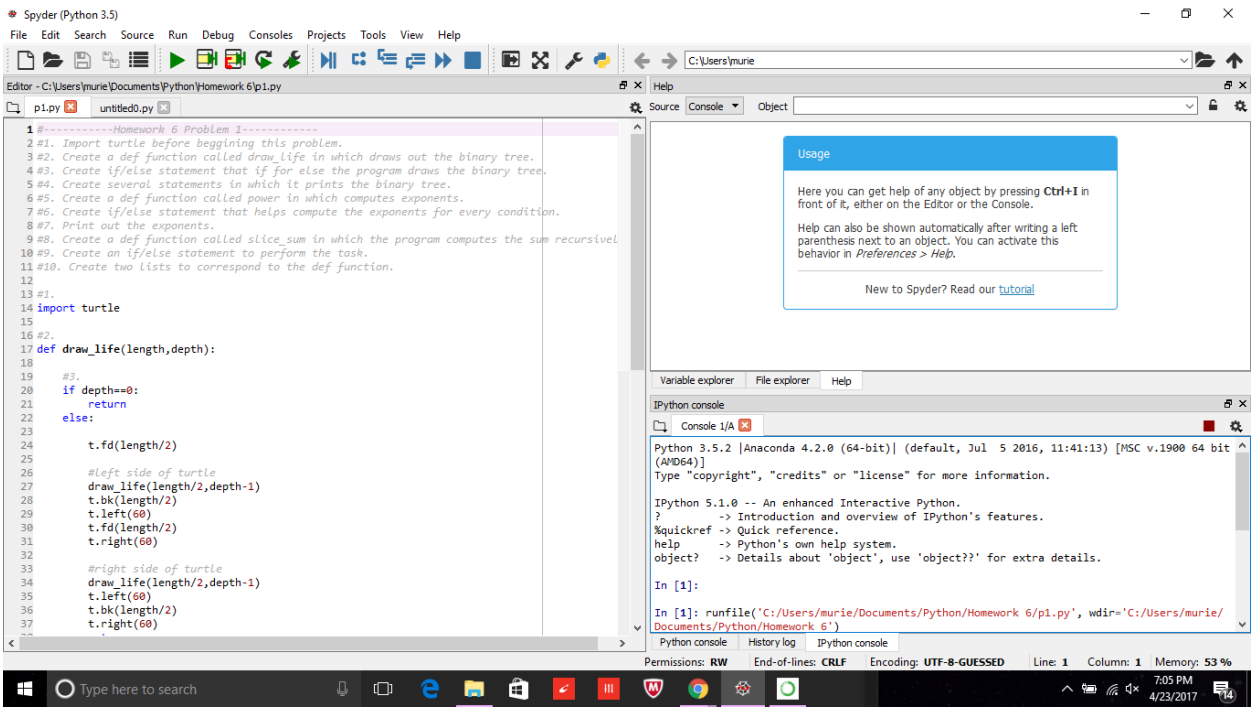
Jamie Andrews

Problem 1:

```
#----------Homework 6 Problem 1----------
#1. Import turtle before beggining this problem.
#2. Create a def function called draw_life in which draws out the binary tree.
#3. Create if/else statement that if for else the program draws the binary tree.
#4. Create several statements in which it prints the binary tree.
#5. Create a def function called power in which computes exponents.
#6. Create if/else statement that helps compute the exponents for every condition.
#7. Print out the exponents.
#8. Create a def function called slice_sum in which the program computes the sum recursively.
#9. Create an if/else statement to perform the task.
#10. Create two lists to correspond to the def function.

#1.
import turtle

#2.
def draw_life(length,depth):

    #3.
    if depth==0:
        return
    else:

        t.fd(length/2)
```

```
        #left side of turtle
        draw_life(length/2,depth-1)
        t.bk(length/2)
        t.left(60)
        t.fd(length/2)
        t.right(60)

        #right side of turtle
        draw_life(length/2,depth-1)
        t.left(60)
        t.bk(length/2)
        t.right(60)
        return
 #4.
window = turtle.Screen()
t = turtle.Turtle()
t.color("green")
t.right(120)
draw_life(160,5)
window.exitonclick()

#5.
def power(x,n):
    #6.
    if n == 0:
        return 1
    elif n == 1:
        return x
```

```
    elif n == 2:
        return x * x
    elif n % 2 != 0:
        return x * power(x, n-1)
    elif n % 2 == 0:
        return power(x,n//2) * power(x,n//2)

#7.
i = power(2,3)
print(i)

#8.
def slice_sum(lst, begin, end):

    #9.
    if end==0:
        return 0
    else:
        return lst[begin] + slice_sum(lst,begin+1,end-1)

#10.
X=[0,1,2,3,4,5]
i = [3,2,6,2,1]
```



Problem 2:

----------Homework 6 Problem 2----------

#1. Import itertools before beggining part A

#2. Create class called PrimeIter in which manages the prime integers.

#3. Begin Part A by building the constuctor in the class.

#4. Create a def function in the Class PrimeIter called __next__ which returns the next prime number.

#5. Create a def function in the Class PrimeIter called __iter__ which iterates the prime number.

#6. Begin Part B by Creating another class called PrimeGen which generates prime numbers.

#7. Create a constructor in the class PrimeGen.

#8. Create a def function called genPrime in the class PrimeGen which generates prime numbers.

#9. Create an if statement called __name__ which calles out the 2 classes.

#1.

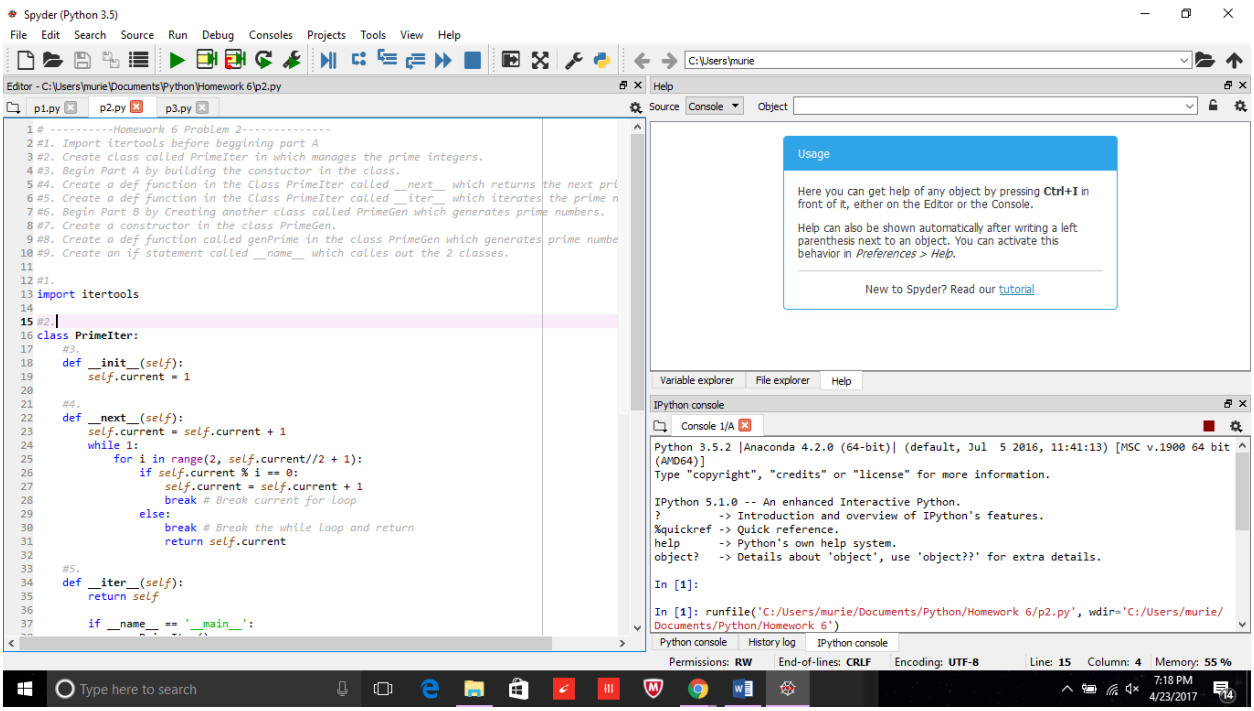
```
import itertools
```

#2.

```
class PrimeIter:
```

```
    #3.
    def __init__(self):
        self.current = 1

    #4.
    def __next__(self):
        self.current = self.current + 1
        while 1:
            for i in range(2, self.current//2 + 1):
                if self.current % i == 0:
                    self.current = self.current + 1
                    break # Break current for loop
                else:
                    break # Break the while loop and return
                    return self.current

    #5.
    def __iter__(self):
        return self

        if __name__ == '__main__':
            p = PrimeIter()
            for x in itertools.islice(p, 10):
                print (x)

#6.
class PrimeGen:
    #7.
```

```
    def __init__(self):
        self.current = 2

    #8.
    def genPrime(self, num):
        for i in range(num):
            while 1:
                for j in range(2, self.current//2 + 1):
                    if self.current % j == 0:
                        self.current = self.current + 1
                        break
                    else:
                        break
                    print (self.current)
                    self.current = self.current + 1

#9.
if __name__ == '__main__':
    p = PrimeGen()
    p.genPrime(10)
```



Problem 3:

#----------Homework 6 Problem 3 ----------

#1. Import the following libaries before doing part A.

#2. Create a def function called gen_rndtup(n) in which generates random sequence.

#3. Write a while loop that for when true it yeilds the random integer.

#4. Create a def function called answer_x in which it prints out the answer for the first tuple.

#5. Create a def function called answer_y in which it prints out the answer for the second tuple.

#6. Create a def function called answer_z in which it prints out the answer for the third tuple.

#7. Create an if statement called __main__ that calls out def functions answer_x, answer_y, and answer_z

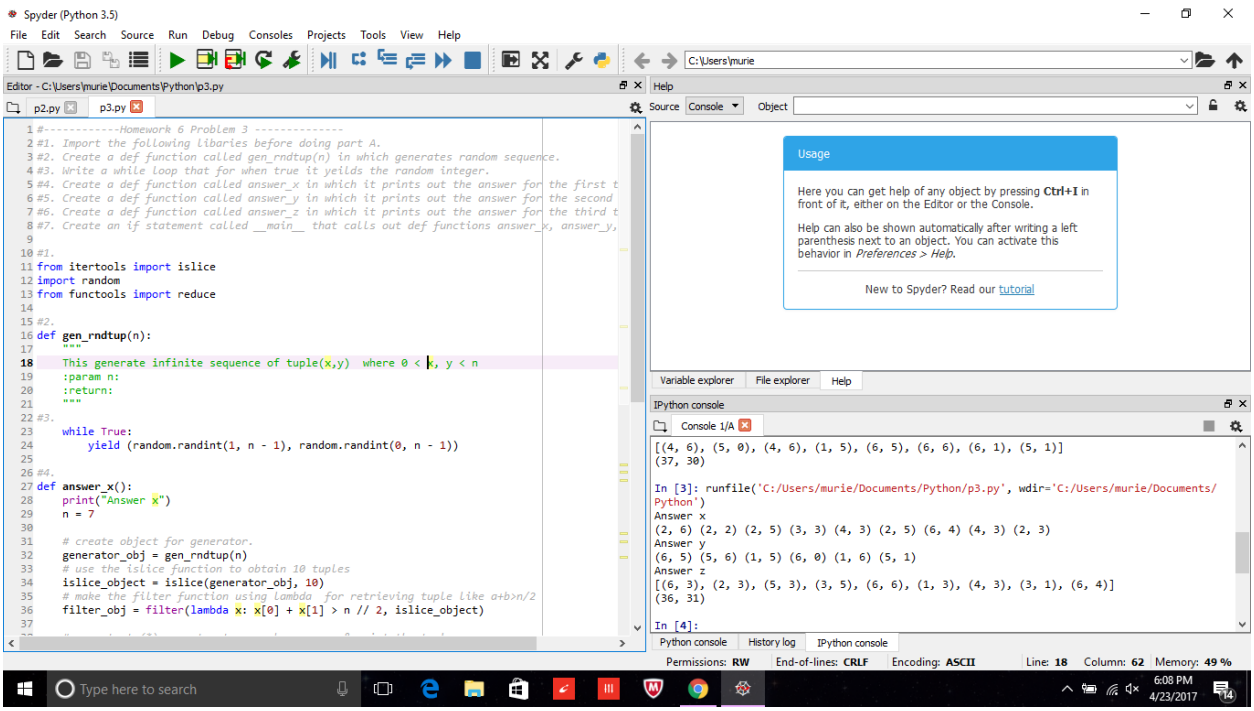
#1.

```
from itertools import islice
import random
from functools import reduce
```

#2.

```
def gen_rndtup(n):
    """
    This generate infinite sequence of tuple(x,y)  where 0 < x, y < n
    :param n:
    :return:
    """
#3.
    while True:
        yield (random.randint(1, n - 1), random.randint(0, n - 1))

#4.
def answer_x():
    print("Answer x")
    n = 7

    # create object for generator.
    generator_obj = gen_rndtup(n)
    # use the islice function to obtain 10 tuples
    islice_object = islice(generator_obj, 10)
    # make the filter function using lambda  for retrieving tuple like a+b>n/2
    filter_obj = filter(lambda x: x[0] + x[1] > n // 2, islice_object)

    # use start (*) operator to unpack sequence & print the tuples
    print(*filter_obj)

#5.
def answer_y():
    print("Answer y")
```

```
    n = 7
    generator_obj = ((random.randint(1, n - 1), random.randint(0, n - 1)) for i in range(10))
    for x in generator_obj:
        if x[0] + x[1] > n // 2:
            print(x, end=" ")
    print()


#6.
def answer_z():
    print("Answer z")
    n = 7
    map_obj = map(lambda x: (random.randint(1, n - 1), random.randint(0, n - 1)), range(10))
    filter_obj = list(filter(lambda x: x[0] + x[1] > n // 2, islice(map_obj, 10)))
    print(filter_obj)
    sum_of_tuples = reduce(lambda x, y: (x[0] + y[0], x[1] + y[1]), filter_obj)
    print(sum_of_tuples)
#7.
if __name__ == '__main__':
    answer_x()
    answer_y()
    answer_z()
```


Spyder (Python 3.5)
File Edit Search Source Run Debug Consoles Projects Tools View Help
Editor - C:\Users\murie\Documents\Python\p3.py
p2.py
p3.py
#----------Homework 6 Problem 3 ----------
#1. Import the following libaries before doing part A.
#2. Create a def function called gen_rndtup(n) in which generates random sequence.
#3. Write a while loop that for when true it yeilds the random integer.
#4. Create a def function called answer_x in which it prints out the answer for the first t
#5. Create a def function called answer_y in which it prints out the answer for the second
#6. Create a def function called answer_z in which it prints out the answer for the third t
#7. Create an if statement called __main__ that calls out def functions answer_x, answer_y,
#1.
from itertools import islice
import random
from functools import reduce
#2.
def gen_rndtup(n):
"""
This generate infinite sequence of tuple(x,y) where 0 < x, y < n
:param n:
:return:
"""
#3.
while True:
yield (random.randint(1, n - 1), random.randint(0, n - 1))
#4.
def answer_x():
print("Answer x")
n = 7
# create object for generator.
generator_obj = gen_rndtup(n)
# use the islice function to obtain 10 tuples
islice_object = islice(generator_obj, 10)
# make the filter function using Lambda for retrieving tuple like a+b>n/2
filter_obj = filter(lambda x: x[0] + x[1] > n // 2, islice_object)
Help
Source Console Object
Usage
Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.
Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.
New to Spyder? Read our tutorial
Variable explorer File explorer Help
IPython console
Console 1/A
[(4, 6), (5, 0), (4, 6), (1, 5), (6, 5), (6, 6), (6, 1), (5, 1)]
(37, 30)
In [3]: runfile('C:/Users/murie/Documents/Python/p3.py', wdir='C:/Users/murie/Documents/Python')
Answer x
(2, 6) (2, 2) (2, 5) (3, 3) (4, 3) (2, 5) (6, 4) (4, 3) (2, 3)
Answer y
(6, 5) (5, 6) (1, 5) (6, 0) (1, 6) (5, 1)
Answer z
[(6, 3), (2, 3), (5, 3), (3, 5), (6, 6), (1, 3), (4, 3), (3, 1), (6, 4)]
(36, 31)
In [4]:
Python console History log IPython console
Permissions: RW End-of-lines: CRLF Encoding: ASCII Line: 18 Column: 62 Memory: 49 %
Type here to search
6:08 PM
4/23/2017