Fundamentals of Python programing:

chapter 5, section 9 (Exercises):

1. In Listing 5.4 (addnonnegatives.py) could the condition of the if statement have used > instead of >= and achieved the same results? Why?

=>yes, we could use if entry > -1, this works because just like entry >= 0, this too includes 0 and excludes all negative number.

2. In Listing 5.4 (addnonnegatives.py) could the condition of the while statement have used > instead of >= and achieved the same results? Why?

=>yes, like said above we could use while entry > -1, and it would work as intended.

3. In Listing 5.4 (addnonnegatives.py) what would happen if the statement

entry = int(input()) # Get the value

were moved out of the loop? Is moving the assignment out of the loop a good or bad thing to do? Why?

=>if we write this line above the loop, then according to the entry the

user gives, we could have either a infinite loop (if entry >= 0) or we could have no loop at all (entry < 0).

but if moved under the loop, we wont take any input at all because the while loop would work infinitly.

4. How many asterisks does the following code fragment print?

```
a = 0
while a < 100:
    print('*', end='')
    a += 1
print()
```

=>100 asterisks.

5. How many asterisks does the following code fragment print?

```
a = 0
while a < 100:
    print('*', end='')
print()
```

=>infinite asterisks.

6. How many asterisks does the following code fragment print?

```
a = 0
while a > 100:
    print('*', end='')
    a += 1
print()
```

=>no asterisks.

7. How many asterisks does the following code fragment print?

```
a = 0
while a < 100:
    b = 0
    while b < 55:
        print('*', end='')
        b += 1
    print()
    a += 1
```

=>100 lines, each line 55 asterisks.

8. How many asterisks does the following code fragment print?

```
a = 0
while a < 100:
    if a  % 5 == 0:
        print('*', end='')
    a += 1
print()
```

=>20 asterisks.

9. How many asterisks does the following code fragment print?

```python
a = 0
while a < 100:
    b = 0
    while b < 40:
        if (a + b) % 2 == 0:
            print('*', end='')
        b += 1
    print()
    a += 1
```

=>adding two odd values gives us a even one,

and adding two even values, will always give us a even value.

in 40 we have 20 odd values, and 20 even values.

so for each line, if we get a odd value for a, 20 odd b values will be added to it and make it even,

and if we get a even value for a, 20 even b values will be added to it and make it even again.

in conclusion:

100 lines, each line 20 asterisks.

10. How many asterisks does the following code fragment print?

```
a = 0
while a < 100:
    b = 0
    while b < 100:
        c = 0
        while c < 100:
            print('*', end='')
            c += 1
        b += 1
    a += 1
print()
```

=>100 to the power of 3 so 1,000,000.

11. What is minimum number of arguments acceptable to the range expression?

=>One

12. What is maximum number of arguments acceptable to the range expression?

=>Three

13. Provide the exact sequence of integers specified by each of the following range expressions.

(a) range(5)

=> 0, 1, 2, 3, 4

(b) range(5, 10)

=>5, 6, 7, 8, 9

(c) range(5, 20, 3)

=>5 ,8 ,11 ,14 ,17

(d) range(20, 5, -1)

=>20 ,19 ,18 ,17 ,16 ,15 ,14 ,13 ,12 ,11 ,10 ,9 ,8 ,7 ,6

(e) range(20, 5, -3)

=>20 ,17 ,14 ,11 ,8

(f) range(10, 5)

=>nothing, since we have to add -1 if we are going back wards.

(g) range(0)

=>nothing.

(h) range(10, 101, 10)

=>10 ,20 ,30 ,40 ,50 ,60 ,70 ,80 ,90 ,100

(i) range(10, -1, -1)

=>10 ,9 ,8 ,7 ,6 ,5 ,4 ,3 ,2 ,1 ,0

(j) range(-3, 4)

=>-3 ,-2 ,-1 ,0 ,1 ,2 ,3

(k) range(0, 10, 1)

=>0 ,1 ,2 ,3 ,4 ,5 ,6 ,7 ,8 ,9

14. What is a shorter way to express range(0, 5, 1)?

=>range(5)

15. Provide an equivalent Python range expression for each of the following integer sequences.

(a) 1,2,3,4,5

=>range(1, 6)

(b) 5,4,3,2,1

=>range(5, 0, -1)

(c) 5,10,15,20,25,30

=>range(5, 31, 5)

(d) 30,25,20,15,10,5

=>range(30, 4, -5)

(e) -3,-2,-1,0,1,2,3

=>range(-3, 4)

(f) 3,2,1,0,-1,-2,-3

=>range(3, -4, -1)

(g) -50,-40,-30,-20,-10

=>range(-50, -9, 10)

(h) Empty sequence

=>range(0)

16. If x is bound to the integer value 2, what integer sequence does range(x, 10*x, x) represent?

=>range(2, 20, 2) so:

2, 4, 6, 8, 10, 12, 14, 16, 18

17. If x is bound to the integer value 2 and y is bound to the integer 5, what integer sequence does range(x, x + y) represent?

=>range(2, 7) so:

2, 3, 4, 5, 6

18. Is it possible to represent the following sequence with a Python range expression: 1,−1,2,−2,3,−3,4,−4?

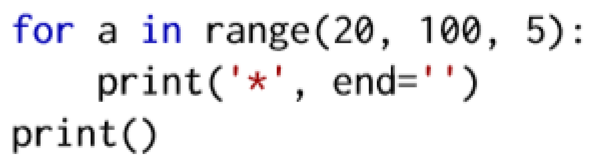
=>No.

19. How many asterisks does the following code fragment print?

```
for a in range(100):
    print('*', end='')
print()
```

=>100.

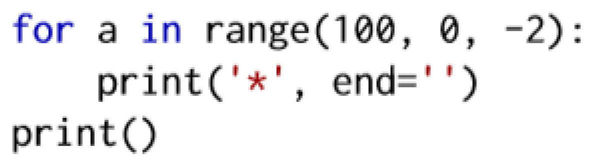
20. How many asterisks does the following code fragment print?



```
for a in range(20, 100, 5):
    print('*', end='')
print()
```

=>16. starts from 20 ends in 95.

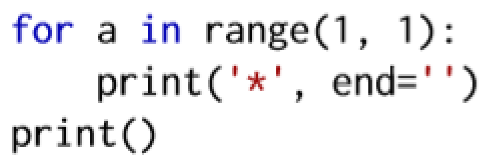
21. How many asterisks does the following code fragment print?



```
for a in range(100, 0, -2):
    print('*', end='')
print()
```

=>50, starting at 100 ending at 2.

22. How many asterisks does the following code fragment print?



```
for a in range(1, 1):
    print('*', end='')
print()
```

=>nothing.

23. How many asterisks does the following code fragment print?

```
for a in range(-100, 100):
    print('*', end='')
print()
```

=>200, from -100 to 99.

24. How many asterisks does the following code fragment print?

```
for a in range(-100, 100, 10):
    print('*', end='')
print()
```

=>20, from -100 to 90.

25. Rewrite the code in the previous question so it uses a while instead of a for. Your code should behave identically.

=>

```
a = -100;
while a < 100:
    print(a)
    a += 10;
```

26. What does the following code fragment print?

```
a = 0
while a < 100:
    print(a)
    a += 1
print()
```

=>prints 100 numbers from 0 to 99.

27. Rewrite the code in the previous question so it uses a for instead of a while. Your code should behave identically.

=>

```
for a in range(100):
        print(a);
```

28. What is printed by the following code fragment?

```
a = 0
while a > 100:
    print(a)
    a += 1
print()
```

=>nothing.

29. Rewrite the following code fragment using a break statement and eliminating the done variable. Your code should behave identically to this code fragment.

```
done = False
n, m = 0, 100
while not done and n != m:
    n = int(input())
    if n < 0:
        done = True
    print("n =", n)
```

=>

```
n, m = 0, 100;
while n != m:
    n = int(input('enter number: '));
    print("n =", n);
    if n < 0:
        break;
```

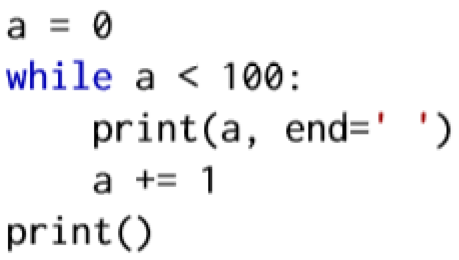
30. Rewrite the following code fragment so it eliminates the continue statement. Your new code's logic should be simpler than the logic of this fragment.

```
x = 5
while x > 0:
    y = int(input())
    if y == 25:
        continue
    x -= 1
    print('x =', x)
```

=>x = 5;

```
while x > 0:
    y = int(input("please enter a number: "));
    if y != 25:
        x -= 1;
        print("x =", x);
```

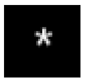
31. What is printed by the following code fragment?



```
a = 0
while a < 100:
    print(a, end=' ')
    a += 1
print()
```

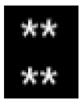
=>100 numbers, from 0 to 99.

32. Write a Python program that accepts a single integer value entered by the user. If thevalue entered is less than one, the program prints nothing. If the user enters a positive integer, n, the program prints an n×n box drawn with * characters. If the users enters 1, for example, the program prints :



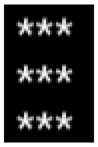
```
*
```

If the user enters a 2, it prints:



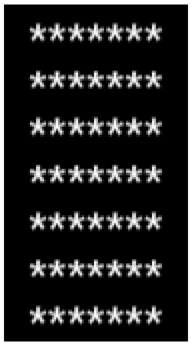
```
**
**
```

An entry of three yields:



```
***
***
***
```

and so forth. If the user enters 7, it prints:



```
*******
*******
*******
*******
*******
*******
*******
```

that is, a 7×7 box of * symbols.

=>

```
num = int(input('Please enter the number of rows and columns: '));

for row in range(num):
    for column in range(num):
        print('*', end='');
    print();
```

33. Write a Python program that allows the user to enter exactly twenty floating-point values. The program then prints the sum , average (arithmetic mean) , maximum , and minimum of the values entered.

```
=>

max = None;

min = None;

sum = 0;

for i in range(5):

    num = float(input('Please enter a floating-point number: '));

    if i == 0:

        max, min = num, num;


    if(num > max):

        max = num;

    elif(num < min):

        min = num;


    sum += num;


print("Max is:", max, "\nMin is:", min, "\nSum is:", sum, "\nAVG is:",
sum/5);
```

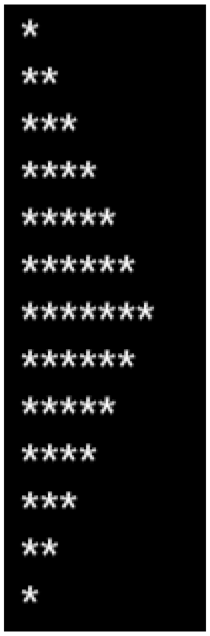
34. Write a Python program that allows the user to enter any number of non-negative floating-point values. The user terminates the input list with any negative value. The program then prints the sum, average (arithmetic mean), maximum, and

minimum of the values entered. The terminating negative value is notused in the computations.

=>

```
max = 0;

min = 0;

sum = 0;

for i in range(5):

    num = float(input('Please enter a floating-point number: '));


    if num < 0:

        break;

    if i == 0:

        max, min = num, num;


    if(num > max):

        max = num;

    elif(num < min):

        min = num;


    sum += num;


print("Max is:", max, "\nMin is:", min, "\nSum is:", sum, "\nAVG is:",
sum/5);
```

35. Redesign Listing 5.34 (startree.py) so that it draws a sideways tree pointing right; for example, if the user enters 7, the program would print

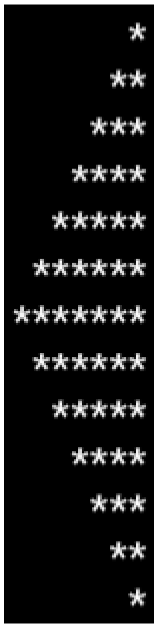


```
*
**
***
****
*****
******
*******
******
*****
****
***
**
*
```

=>

```
length = int(input('Please enter the length: '));


star = 0;
for line in range(length*2-1):
    if line < length:
        star += 1;
    else:
        star -= 1;
    for s in range(star):
        print(end="*");
    print();
```

36. Redesign Listing 5.34 (startree.py) so that it draws a sideways tree pointing left; for example, if the user enters 7, the program would print



```
      *
     **
    ***
   ****
  *****
 ******
*******
 ******
  *****
   ****
    ***
     **
      *
```

=>

```
length = int(input('Please enter the length: '));

space = length;
star = 0;

for line in range(2*length-1):
```

```
if line < length:
        star += 1;
        space -= 1;
else:
        star -= 1;
        space += 1;

for sp in range(space):
        print(end=" ");
for st in range(star):
        print(end="*");
print();
```