**Exercises pg 70**

1. Write the outputs of the following loops:

a. for count in range(5):

print(count + 1, end = " ")

**Output: 1 2 3 4 5**

b. for count in range(1, 4):

print(count, end = " ")

**Output: 1 2 3**

c. for count in range(1, 6, 2):

print(count, end = " ")

**Output: 1 3 5**

d. for count in range(6, 1, –1):

print(count, end = " ")

**Output: 6 5 4 3 2**

2. Write a loop that prints your name 100 times. Each output should begin on a

new line.

**for eachpass in range (100):**

**print("Jacob Kolb", end = "")**

3. Explain the role of the variable in the header of a for loop.

**The role of the variable is to define the loops value.**

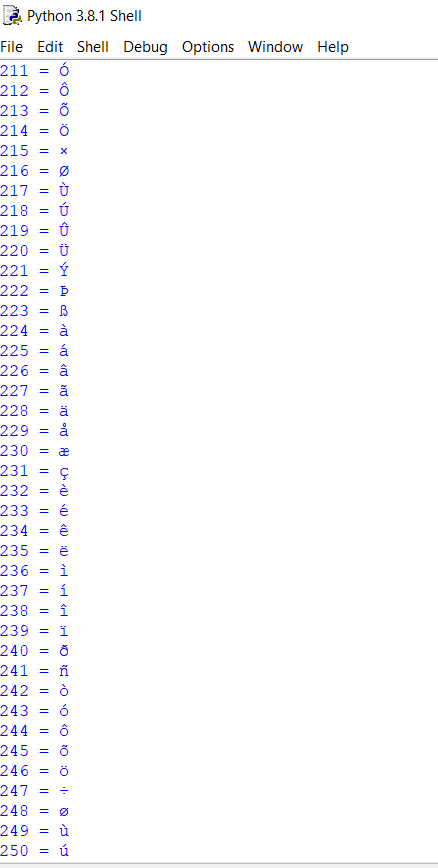
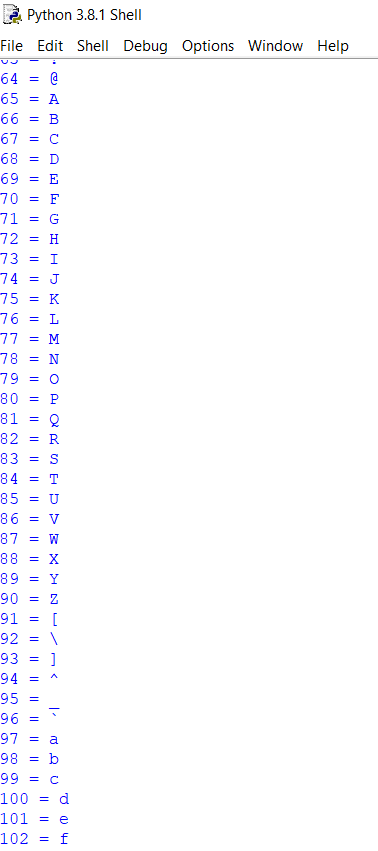
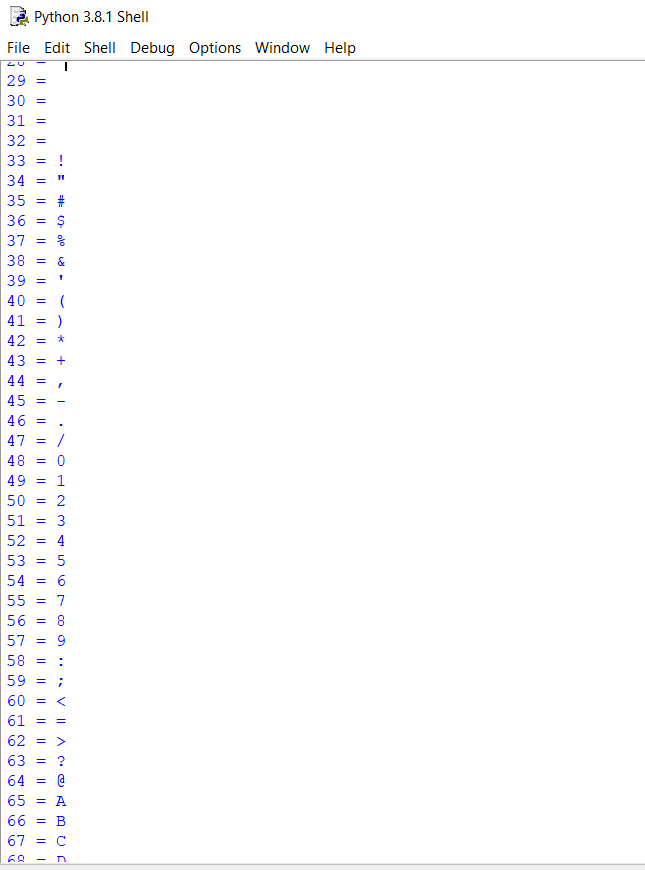
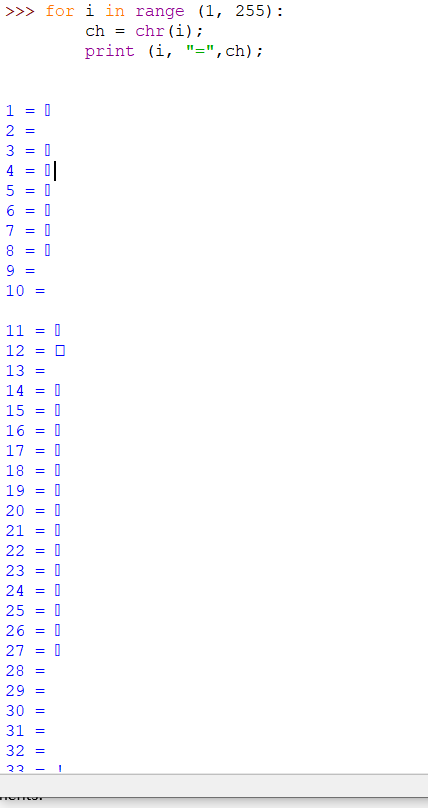
4. Write a loop that prints the first 128 ASCII values followed by the corresponding

characters (see the section on characters in Chapter 2). Be aware that most of the

ASCII values in the range “0..31” belong to special control characters with no standard

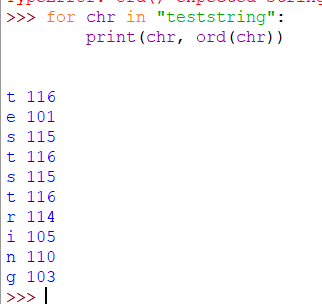
print representation, so you might see strange symbols in the output for these

values.



5. Assume that the variable teststring refers to a string. Write a loop that prints

each character in this string, followed by its ASCII value.



**Exercises pg 72**

1. Assume that the variable amount refers to 24.325. Write the outputs of the following

statements:

a. print("Your salary is $%0.2f" % amount)

b. print("The area is %0.1f" % amount)

c. print("%7f" % amount)

**>>> amount = 24.325**

**>>> print("Your salary is $%0.2f" % amount)**

**Your salary is $24.32**

**>>> print("The area is $%0.1f" % amount)**

**The area is $24.3**

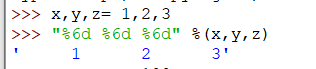
**>>> print("%4f" % amount)**

**24.325000**

**>>>**

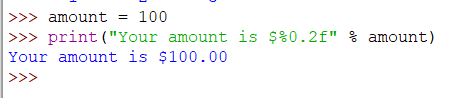
2. Write a code segment that displays the values of the integers x, y, and z on a single

line, such that each value is right-justified with a field width of 6.



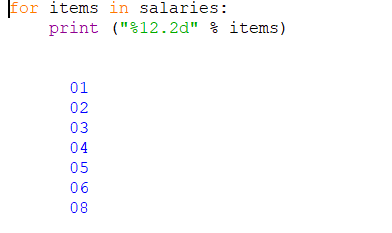
3. Write a format operation that builds a string for the float variable amount that has

exactly two digits of precision and a field width of zero.



4. Write a loop that outputs the numbers in a list named salaries. The outputs should be

formatted in a column that is right-justified, with a field width of 12 and a precision of 2.



**Exercises pg 85**

1. Assume that x is 3 and y is 5. Write the values of the following expressions:

a. x == y

**False**

b. x > y – 3

**True**

c. x <= y – 2

**True**

d. x == y or x > 2

**True**

e. x != 6 and y > 10

**False**

f. x > 0 and x < 100

**True**

2. Assume that x refers to a number. Write a code segment that prints the number’s

absolute value without using Python’s abs function.

**if x< 0:**

**x= -x**

3. Write a loop that counts the number of space characters in a string. Recall that the

space character is represented as ' '.

**count = 0**

**>>> s = "Hello I am Jacob"**

**>>> for ch in s:**

**if ch == ' ':**

**count = count + 1**

**print count**

4. Assume that the variables x and y refer to strings. Write a code segment that prints

these strings in alphabetical order. You should assume that they are not equal.

**if x < y:**

**print(x)**

**print(y)**

**else:**

**print(y)**

**print(x)**

5. Explain how to check for an invalid input number and prevent it being used in a

program. You may assume that the user enters a number.

**Use an if-else statement to check if the number is valid.**

6. Construct truth tables for the following Boolean expressions:

a. not (A or B)

|  |  |  |
| --- | --- | --- |
| A | B | not(A or B) |
| T | T | F |
| T | F | F |
| F | T | F |
| F | F | T |

b. not A and not B

|  |  |  |
| --- | --- | --- |
| A | B | not A and not B |
| T | T | F |
| T | F | F |
| F | T | F |
| F | F | T |

7. Explain the role of the trailing else part of an extended if statement.

**The else part executes code when none of the if/elif statements are true.**

8. The variables x and y refer to numbers. Write a code segment that prompts the user for

an arithmetic operator and prints the value obtained by applying that operator to x and y.

**op = input(“Enter operation: “)**

**if op == ‘+’:**

**print(x + y)**

**elif op == ‘-’:**

**print(x - y)**

**elif op == ‘\*’:**

**print(x \* y)**

**elif op == ‘/’:**

**print(x / y)**

**else:**

**print(“Invalid operation”)**

9. Does the Boolean expression count > 0 and total // count > 0 contain a potential error? If not, why not?

**No, since count is evaluated first there can’t be division by 0**

**Exercises pg 92**

1. Translate the following for loops to equivalent while loops:

a. for count in range(100):

print(count)

**i = 0**

**while i < 100:**

**print(i)**

**i += 1**

b. for count in range(1, 101):

print(count)

**count = 1**

**while count < 101:**

**print(count)**

**count += 1**

c. for count in range(100, 0, –1):

print(count)

**count = 100**

**while count > 0:**

**print(count)**

**count -= 1**

2. The factorial of an integer N is the product of the integers between 1 and N, inclusive.

Write a while loop that computes the factorial of a given integer N.

**prod = 1**

**count = 2**

**while count <= n:**

**prod \*= count**

**count += 1**

**print(prod)**

3. The log2 of a given number N is given by M in the equation N 5 2M. Using integer

arithmetic, the value of M is approximately equal to the number of times N can be

evenly divided by 2 until it becomes 0. Write a loop that computes this approximation

of the log2 of a given number N. You can check your code by importing the

math.log function and evaluating the expression round(math.log(N, 2)) (note

that the math.log function returns a floating-point value).

**count = 0**

**n = int(input(“Enter num: “)**

**while n > 0:**

**n //= 2**

**count += 1**

**print(count)**

4. Describe the purpose of the break statement and the type of problem for which it is

well suited.

**The break statement will cause the loop to exit. It is used when you need the loop to run at least once.**

5. What is the maximum number of guesses necessary to guess correctly a given number

between the numbers N and M?

**log2(M - N + 1)**

6. What happens when the programmer forgets to update the loop control variable in a while loop?

**They could get caught in an infinite loop.**