Chapter3_Exercises

May 27, 2017

1 Chapter 3 Exercises

1.1 Lumberjack Program

```
In [6]: def repeat_lyrics():
            print_lyrics()
            print_lyrics()

        def print_lyrics():
            print("I'm a lumberjack, and I'm okay.")
            print("I sleep all night and I work all day.")

        repeat_lyrics()
```

```
I'm a lumberjack, and I'm okay.
I sleep all night and I work all day.
I'm a lumberjack, and I'm okay.
I sleep all night and I work all day.
```

1.2 Exercise 3.1

Write a function named right_justify that takes a string named s as a parameter and prints the string with enough leading spaces so that the last letter of the string is in column 70 of the display.

```
In [7]: def right_justify(s):
            print(" "*(70-len(s)) + s)

        right_justify("monty")
```

```
                                                                     monty
```

1.3 Exercise 3.2

A function object is a value you can assign to a variable or pass as an argument. For example, do_twice is a function that takes a function object as an argument and calls it twice: def do_twice(f): f() f() Here's an example that uses do_twice to call a function named print_spam

twice. def print_spam(): print('spam') do_twice(print_spam) 1. Type this example into a script and test it. 2. Modify do_twice so that it takes two arguments, a function object and a value, and calls the function twice, passing the value as an argument. 3. Copy the definition of print_twice from earlier in this chapter to your script. 4. Use the modified version of do_twice to call print_twice twice, passing 'spam' as an argument. 5. Define a new function called do_four that takes a function object and a value and calls the function four times, passing the value as a parameter. There should be only two statements in the body of this function, not four.

```
In [13]: def print_twice(bruce):
             print(bruce)
             print(bruce)

         def do_twice(f, val):
             f(val)
             f(val)

         def do_four(f, val):
             do_twice(f, val)
             do_twice(f, val)

         def print_spam():
             print('spam')

         do_four(print_twice, 'spam')
```

```
spam
spam
spam
spam
spam
spam
spam
spam
```

1.4 Exercise 3.3

1. Write a function that draws a grid like the following: (excluded here) Hint: to print more than one value on a line, you can print a comma-separated sequence of values: print('+', '-') By default, print advances to the next line, but you can override that behavior and put a space at the end, like this: print('+', end='') print('-') 28 Chapter 3. Functions The output of these statements is '+ -'. A print statement with no argument ends the current line and goes to the next line.
2. Write a function that draws a similar grid with four rows and four columns.

```
In [33]: def print_top():
             print('+', '- '*4, '+','- '*4,'+')
         def print_tops():
             print('+', '- '*4, '+','- '*4, end='')
```

```
        print('+', '- '*4, '+','- '*4, end='')
        print('+')

    def print_middle():
        print('|', ' '*8, '|', ' '*8, '|')
    def print_middles():
        print('|', ' '*8, '|', ' '*8, end='')
        print('|', ' '*8, '|', ' '*8, end='')
        print('|')

    def print_bigrow():
        print_middles()
        print_middles()
        print_middles()
        print_middles()
        print_tops()

    print_top()
    print_middle()
    print_middle()
    print_middle()
    print_middle()
    print_top()
    print_middle()
    print_middle()
    print_middle()
    print_middle()
    print_top()

    print()

    def print_bigger():
        print_tops()
        print_bigrow()
        print_bigrow()
        print_bigrow()
        print_bigrow()

    print_bigger()
```

```
+ - - - - -  + - - - - -  +
|           |           |
|           |           |
|           |           |
|           |           |
+ - - - - -  + - - - - -  +
|           |           |
|           |           |
```