

CS 111 – Introduction to Computer Science – Fall 2017

Lab Assignment #6

*Repetition Statements * (30pts)*

Due Date: at 11:59pm on Saturday, March 17.

In this lab, you will be working with Python loops to gain experience in the use of repetition statements. Some of the program examples will also use the selection statements introduced in the previous lab.



Before getting started with the lab, copy the entire `lab6` folder from the course folder (`H:\Compsci\givens\cs111`) to your `U:\cs111` folder.

Debugging

Open the `squares.py` program from your `lab6` folder. The program is suppose to print all squares less than n . For example, if n is 100, it should print 0 1 4 9 16 25 36 49 64 81 (one per line). But it contains several logical errors. Find and correct the logical errors and add your name to the file prolog.

Conversion Table



You are to design and implement a program (`conversion.py`) that prints a Celsius to Fahrenheit conversion table as shown below.

Celsius	Fahrenheit
0	32
5	41
10	50
15	59
...	...
95	203
100	212

The program should produce entries in the table for degrees Celsius from 0 to 100 inclusive, in increments of 5 degrees. Your program should be written to the following specifications:

- Include an appropriate file prolog and appropriate comments throughout the program.
- The format of the output should look very similar to that shown above (but with all 21 values shown).


After completing the program, be sure to fully test the program and verify the results.



*Based on the labs of Dr. Rance Necaise

Factorial

The factorial of a nonnegative integer n is the product of all the integers less than or equal to n , written $n!$. For instance $5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$.

 Design and write a program called `factorial.py` that gets an integer from the user, and if it is greater than, or equal to, 0, returns the factorial of that number. (Note that $0! = 1$.)

The following are example program runs:

```
Enter an integer: 6
6! = 720
```

```
Enter an integer: -5
"No negative numbers, please."
```

Your program should be written to the following specifications:

- Include an appropriate file prolog and appropriate comments throughout the program.
- Your program should prompt the user for an integer and then give appropriate print statement based on the user input.

After completing the program, be sure to fully test the program and verify the results.


■



Note

In many programming languages, this program would quickly cause overflow because factorial is a very fast growing function. For instance, in Java, $13!$ would cause overflow if you were using ints, and $21!$ would cause overflow if you were using longs (a type of integer that is given twice the number of bits). However, Python automatically increases the memory space of an integer to avoid overflow. So, as long as you have the memory space available, you can calculate the factorial of larger numbers in Python.

Guessing a Random Number

 Design and implement a program, `guess.py`, in which the program chooses a random number between 1 and 30 (inclusive to both), and allows the user to guess what the random number is, up to 5 guesses. The program will provide hints for the user when the user guesses incorrectly.

Your code should follow the format below:

1. The program picks a random number between 1 and 30 (inclusive for both).
2. The user guesses a number, and program says “higher” if the program’s number is higher and “lower” if the program’s number is lower
3. Guessing and prompting continues until the user guesses the correct number or the user has guessed incorrectly 5 times.
4. If the user guessed the number, print “You guessed it!”, otherwise, print the number for the user.

Example output is as follows:

```
Guess a number between 1 and 30: 15
higher
Guess again: 25
lower
Guess again: 20
higher
Guess again: 22

You guessed it!

Guess a number between 1 and 30: 2
higher
Guess again: 4
lower
Guess again: 6
higher
Guess again: 8
higher
Guess again: 10

The number was 18!
```

Your program must be written to the following specifications:

- Include an appropriate file prolog and appropriate comments throughout the program.
- Your program should continually prompt the user for a guess until the user guesses the correct answer or until the user has guessed incorrectly 5 times. This means your loop will have 2 conditions (while the guess is incorrect and the user has provided less than 5 guesses, continue looping). Two conditions means your loop will have two updates: a new guess and a new count of the guesses.
- One way to solve the problem is to start with just one condition, while the guess is incorrect. Once your code works for this single condition, add the second one (and the user has provided less than 5 guesses), and don’t forget all your updates!

■

Finishing Up

When you are finished with the lab, you need to show me that your code runs and correctly computes the solution for each part of the lab. Also, you need to submit the source files for grading. To submit the files, find the lab assignment on Canvas and upload the four files:

- `squares.py`
- `conversion.py`
- `factorial.py`
- `guess.py`

Remember, all of the files must be named exactly as indicated above, with the same case and with no spaces or special characters.