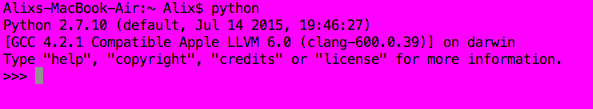
Assignment 1: Interactive Python

1. In your terminal, type ‘python’. You should see something like this:

This is the interactive Python shell.

1. Try typing each of the following, pressing ‘enter’ after each:
   1. "Hello World"
   2. 3
   3. 129.401

The shell is echoing your entries back to you!

1. Try typing each of the following, pressing ‘enter’ after each:
   1. 2 + 2
   2. 10 + 3
   3. 6 - 4
   4. 9 - 18
   5. 7 \* 8
   6. -4 \* 11
   7. 18 / 6
   8. 9 / 4
      1. Did this answer surprise you? Try 9.0 / 4.

The shell does addition, subtraction, multiplication and division… though division with integers (whole numbers) will always return integers.

1. Try typing each of the following, pressing ‘enter’ after each:
   1. print("Hello World")
   2. 4.567 \* 8.323 \* 17
   3. 3 + 2 \* 4
      1. Is Python using the order of operations from math class? Multiplication, division, addition, and subtraction?
   4. \_ (you type the underscore by pressing the shift key and holding it while you press the dash key to the right of the 0 key at the top right of the keyboard).
      1. The underscore gets you the most recent output from the shell.
   5. \_ + 4
   6. "Hello" + " " + "World"
   7. "la" \* 4

Assignment 2: Variables

1. Create a new python project called Person.py.
2. Create the following variables:
   1. An int called age which has the value of your age.
   2. A double called height which has the value of your height.
   3. A String called firstName which has the value of your first name.
   4. A String called lastName which has the value of your last name.
   5. A boolean called isAStudent which is true if you are a student and false if you are not a student.
3. Make the program print all of this information using the variables. For example, when you want to print your name, do not do:
   1. print “My name is Alix Feinsod”

Instead, do:

b. print "My name is " + firstName + " " + lastName

1. Add 2 more variables of your choice with 2 different types that represent something about you. Then print those variables as well.
2. Run the program and make sure the output works as expected.
3. Save the project and email to the instructor

Assignment 3: Booleans

1. Create a new python project called Booleans.py.
2. Create the following variables:
   1. Ints x and y, equal to two values of your choice.
   2. 6 booleans called a,b,c,d,e,f,g that represent all the possible comparisons between x and y: ==, !=, >, <, >=, <=.
3. Make the program print all of this information using the variables. For example, when printing if x and y are equal, use the following:

print x, " is equal to ", y , ": ", a

1. Repeat steps 2 and 3 for two Strings of your choice called s1 and s2, with booleans named h,i,j,k,l,m.
2. Using compound logic operators, create and print booleans b1, b2, and b3 that represent:
   1. x is greater than y and s1 is greater than s2
   2. x is equal to y or s1 is not equal to s2
   3. The opposite of (b)
3. Save the project and email to the instructor.

Assignment 4: User Input

1. Create a new python project called AboutYou.py.
2. Prompt the user for their name and store the name in a String variable called ‘name’ using this code:

name = str(raw\_input("What's your name? "))

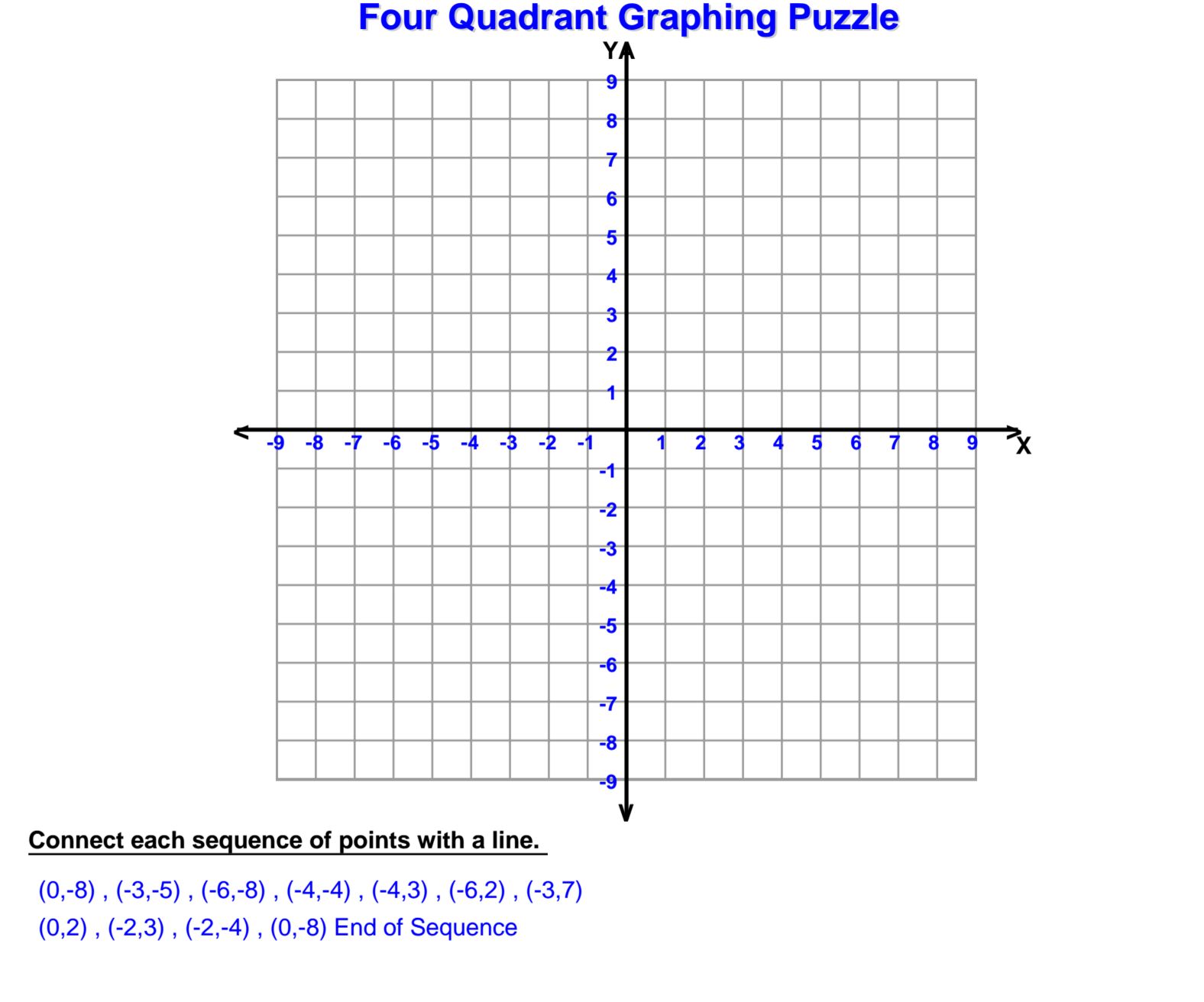
1. Next, use the name to make a personalized prompt for the age, which you should store in an int variable. For example, if the name was Alix, the next line should print something like “Hi, Alix! How old are you?”.
2. Continue prompting and storing the following information from the user.
   1. Birthday (day as an int, month as a String, year as an int)
   2. Favorite Book (as a String)
   3. Hometown (as a String)
   4. Lucky number (as a double)
   5. Shoe size (as a double)
   6. On a scale of 1 to 5, how excited they are to learn Python (as a double)
   7. One cool fact about them (as a String)
3. Run the program and make sure the dialogue works as expected.
4. Next, have the program print out all the information in the following format:

Alix Feinsod, age 22, was born on November 22, 1994 in Berkeley. Alix has lucky number 7, favorite book Harry Potter, and a shoe size of 7.5. On a scale of 1 to 5, Alix is this excited to learn about Python: 10! A fun fact about Alix is: I love cats!

1. Run the program and make sure the output works as expected.
2. Now, we will do some math. Create new variables, using the existing ones, to represent the following information:
   1. An int that is their age in 5 years from now
   2. A boolean that is true if their age is equal to their birthday (day of the month they were born)
   3. A boolean that is true if their birth year divided by 2000 is greater than 1, which will mean they were born in or after the year 2000
   4. A double that is their shoe size mutliplied by their excitement level
3. Have all of the new information printed as well, with explanations of each one.
4. Run the program and make sure the output works as expected.
5. Save the project and email to the instructor.

Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

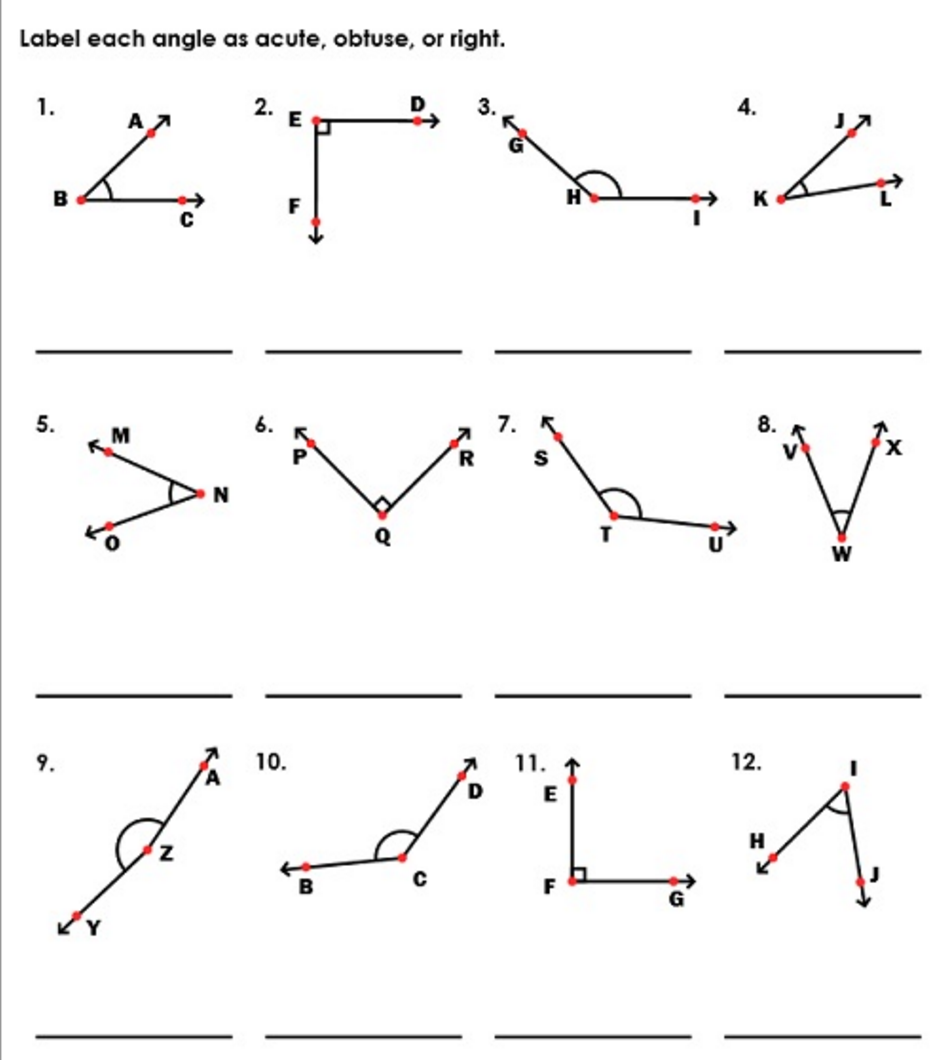
Assignment 5: Graphing Puzzle



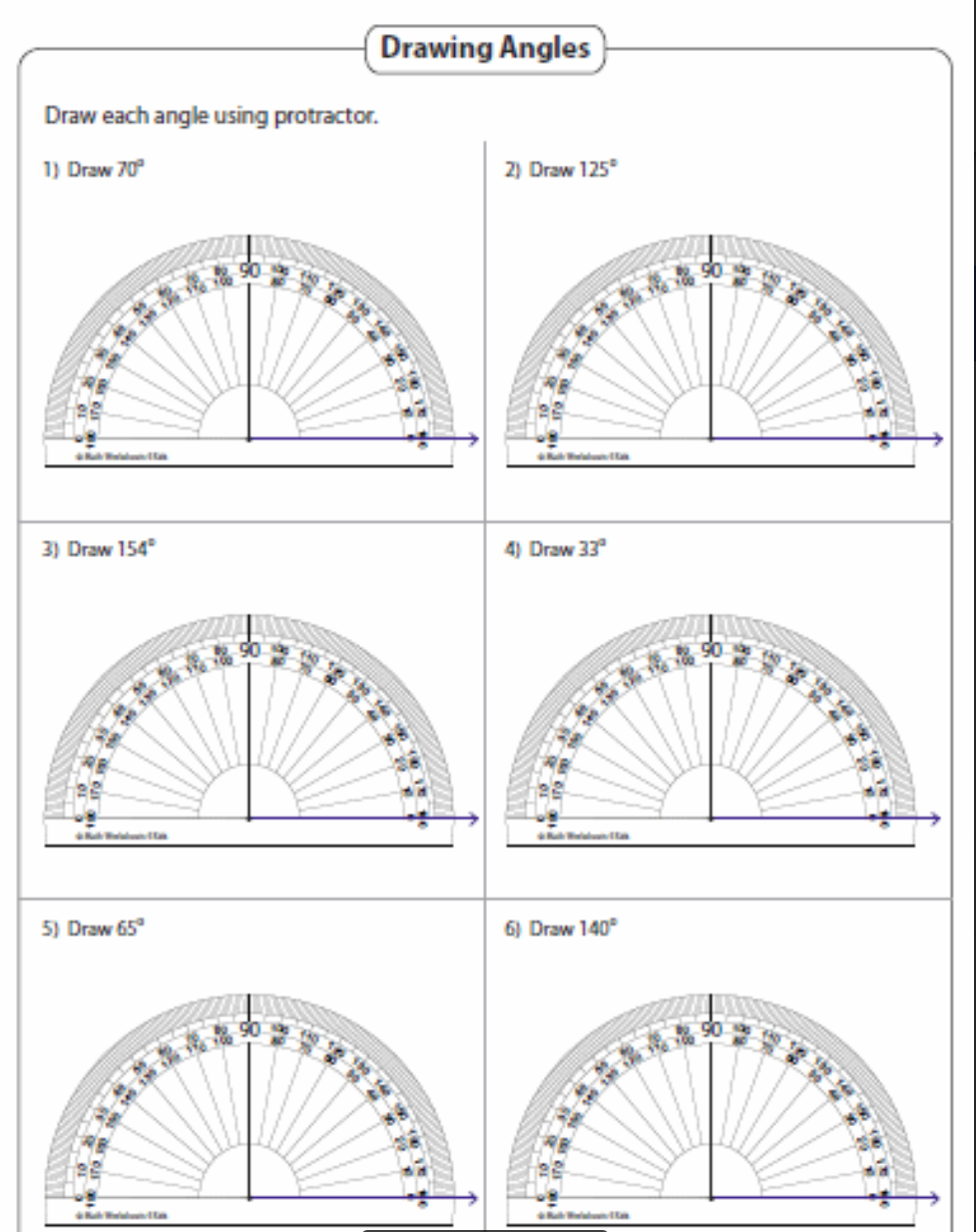
What is the shape?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Assignment 6: Angle Types and Measuring

Page 1 of 2



Page 2 of 2



Assignment 7: Turtle Practice

1. You are given the Turtle.py class
2. Read through the current code and run it. Make sure the code and the turtle’s motions make sense to you.
3. Have the turtle draw all of the following, next to each other:
   1. An equilateral triangle
   2. A square
   3. An octagon (eight sides)
   4. For a challenge, try to draw a 5-pointed star. What angle will you need to turn by each time?
4. Save the program and email it to the instructor.

Assignment 8: Shapes

1. Modify your previous project Turtle.py so that the shapes are not connected by a line using the penup() and pendown() methods.
2. Make each shape have a different color and a different pen size.
3. Save the program and email it to the instructor.

Assignment 9: Conditionals

1. You are given the Quiz.py class
2. Read through the current code and run it. Make sure the code and the quiz make sense to you.
3. Add 5 new questions to the quiz. They can be multiple choice or true/false, with at least one of each. One question should be about math, and one should be about science, and one should be about Python programming. The rest can be whatever you want! Make sure you keep track of the number of right answers and print them at the end.
4. Run the program and make sure the quiz works as expected.
5. Save the program and email to instructor.
6. Trade with a partner and take each others quizzes!

Assignment 10: Shape Shifter

1. Create a project called ShapeShifter.py.
2. Create a program that prompts the user to enter a color, a side length, and a pen size. Ask if they want to draw a square. If yes, draw a square with that color, side length, and pen size. If no, quit the program.
3. Save the program and email it to the instructor.

Assignment 11: While Loops

1. You are given the Repeater.py class
2. Read through the current code and run it. Make sure the code and the while loop make sense to you.
3. Make the program print the message back 10 times instead of 3.
4. Put numbers in front of the message each time it prints. For example, if the message is ‘Hi!’, it should look like this:

1. Hi!

2. Hi!

3. Hi!

… etc

This is tricky because the numbers start at 0, not 1. You can either change the starting number and the ending number by 1 each, or compute the number to print based on the count value.

1. Change the program so that it asks the user how many times they want the message repeated, and then repeats the message that number of times.
2. Run the program and make sure the output works as expected.
3. Save the program and email it to the instructor.

Assignment 12: Shape Shifter Revisited

1. Reopen your project ShapeShifter.py.
2. Rewrite the square drawing part so that it uses a while loop with a counter to draw the 4 sides of the square.
3. Now, rewrite the program so that it asks the user for the number of sides of a shape, and draws a polygon with that number of sides. You’ll need to compute the number of degrees to turn based on the number of sides, and the condition of the while loop will also depend on that number.
4. Save the program and email it to the instructor.

Assignment 13: Doubles

1. Create a file called Doubles.py.
2. Create a program that rolls 2 dice (using random numbers between 1 and 6) and prints the results.
3. Use a while loop to keep going until it rolls doubles. If they roll doubles, print a message of congratulations.
4. This all happens so fast! Change the program so that each time it asks the user if they want to roll the dice and waits for user input to go on. If the user does not want to roll the dice, you can break out of the loop.
5. Run the program and make sure it works as expected.
6. Save the program and email to the instructor.

Assignment 14: Random Shapes

1. Create a project called RandomShapes.py
2. Make your project print 10 different shapes, separated without lines in between. For each shape, your code should randomly select:
   1. The x and y location where the shape begins, both between -300 and 300
   2. The number of sides of the shape, between 3 and 15
   3. The side length, between 10 and 100
   4. The pen width, between 1 and 10
   5. The color, using this code:

random.choice([“red”, “orange”, “yellow”, “green”, “blue”, “purple”])

1. If you’re having trouble, start by making a program that makes 1 random shape. Then, use a while loop to repeat that 10 times.
2. Save the program and email it to the instructor.

Assignment 15: High Or Low

1. Create a project HighOrLow.py.
2. Your program will play the High-Low game with the user. It should pick a random number between 1 and 100 and prompt the user to guess. If the guess is lower than the secret number, tell the user they are too low. If the guess if higher than the secret number, tell the user they are too low. If they guess the secret number, print a message of congrats!
3. Run the program and make sure it works as expected.
4. Add a counter to keep track of the number of guesses. At the end, print how many guesses they took to get the number.
5. Run the program and make sure it works as expected.
6. Save the program and email to the instructor.
7. Swap with a partner and play their High-Low game.

Assignment 16: Lists

1. Create a program called ListFun.py.
2. Create an empty list. Fill up the list with odd numbers (1,3,5,...). Append the values to the list one at a time. Then, print them on the screen one by one using the index. Don’t use a while loop. This part of the code will be pretty long.
3. Create another empty list. Fill up the list with the number 1.25, then print all the numbers in the list. You must use a while loop to fill this list and to print. Use 10 for the while condition, so the list will have 10 elements in it.
4. Create another empty list. Fill it with 1000 random numbers between 1 and 1000. Use a while loop!
5. Challenge: print all the numbers in the random list out with a space between each one, making them start on a new line every 20 numbers. You can use a double while loop to do that! This is tricky!
6. Run the program and make sure it works as expected.
7. Email the file to the instructor.

Assignment 17: For Loops

1. Create a project called ForLoopFun.py.
2. In between each of the following parts, print a blank line using print “”
3. First, use a for loop to print “I love Python” 10 times.
4. Then, prompt the user to enter a number, and then use a for loop to count from 0 to that number. Print each number separated by a space. To do that, create an empty string (“”) and in the for loop, attach the number to that string, and then a space. At the end, print the string.
5. Create a list of ints with values 1-10. You can use a for or while loop to fill it. Then, use a for loop to create a copy of the list, where all the values are changed by 2 (add 2 to each number). Create the second list before you fill it using a for loop. Print out the values of the second list separated by spaces.
6. Bonus: Create n list of ints with values 1-100. Use a for loop to print the numbers line by line, but print an exclamation point after multiples of 5. Use an if statement and the math concept of mod:
   1. x mod y gives the remainder when x is divided by y. For example, 5 mod 2 is 1 because 5/2 is 2 remainder 1. In Python, % is mod. So 12 % 5 is 2. Use this to find multiples of 5.
7. Run the program and make sure it works as expected.
8. Email the file to the instructor.

Assignment 18: Choose Your Own Adventure

1. Create a program called ChooseYourOwnAdventure.py.
2. Make a "Choose Your Own Adventure" game. The starting room should give the user two choices. Then the second room they travel to should give them two more choices. Finally the third room should give them two choices. Each ending room should have a different message. Your game will have a total of fifteen rooms: 1 starting room, 2 possibilities from there, 4 from those, 8 final rooms from those.
3. Run the program and make sure it works as expected.
4. Email to the instructor.
5. Swap with a partner and play their game!

Assignment 19: Functions

1. Create a project called HatShop.py.
2. This program should pull up a menu for a Hat Shop with 4 options: Add a hat to cart, see your cart, remove hats from the order, and check out (which should end the program). These 4 options should call 4 different functions. The program should keep track of how many hats are in the order at any time. For an extra challenge, allow hats to have different colors and allow for adding and removing different colors, and print the colors when you see the cart.
3. Run the program and make sure it works as expected.
4. Email the file to the instructor.

Assignment 20: Final Project Options

Project 1. Hangman

1. Create a file called Hangman.py.
2. It must randomly choose a word from a list of words, stop when all the letters in the word are guessed. It must give them limited tries and stop after they run out. It must display the letters they have already guessed. Suggestion: make a list of the letters in the word.
3. You will be given a list of words to use.

Project 2: Tic Tac Toe

1. You are given starter code in a file called TicTacToe.py.
2. Complete the project so that it makes a 2-player game of Tic Tac Toe, alternatively prompting X and O to choose their location, which is one int representing the row on the board and one int representing the column. The game should end if someone wins (gets 3 in a row) or if the board is full and it is a tie.

Project 3: Password Checker

1. Create a program called PasswordChecker.py
2. The program should prompt the user for a password, then check that it fulfils all the following:
   1. Contains one lowercase letter [a-z]
   2. Contains one number [0-9]
   3. Contains one capital letter [A-Z]
   4. Contains at least one of these: [$#@]
   5. Is at least 8 characters long
   6. Is no more than 16 characters long
3. Print ‘valid’ if this is a valid password and ‘invalid’ if not.

Project 4: Fibonacci Numbers

1. Create a program called Fibonacci.py
2. Prompt the user to enter the number of Fibonacci numbers they want generated. Then, print that many Fibonacci Numbers.
3. The Fibonacci numbers are generated by adding the previous 2 numbers to get the next. The sequence starts with 1 and 1.

1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, …

Harder Projects:

**Address Book -** Write an application that acts like an address book. Each entry in it should contain first and last name, phone number, and email. It should be able to sort by at least one field (like last name) - more would be excellent! Next, make it be able to only print entries that match certain criteria (like certain area codes or first names beginning with A).

**Calculator -** Write a program that acts as a calculator. It should take integers and decimals, and do addition, subtraction, multiplication, division, exponents, and give error messages when something is typed wrong. An excellent project will also do modulus, factorial, square roots, and if you press a certain button, it displays a manual of how to use it.

**Quiz Maker** – Make an application which takes various questions form a file, picked randomly, and puts together a quiz for students. Each quiz can be different and then reads a key to grade the quizzes.

**Product Inventory Project** – Create an application which manages an inventory of products. Create a product class which has a price, id, and quantity on hand. Then create an inventory class which keeps track of various products and can sum up the inventory value.

**Movie Store** – Manage video rentals and controls when videos are checked out, due to return, overdue fees and for added complexity create a summary of those accounts which are overdue for contact.

**Student Grade Book Application** – Keep track of students (with a student class that has their name, average, and scores) in a class and their grades. Assign their scores on tests and assignments to the students and figure out their average and grade for the class. For added complexity put the students on a bell curve.

**Bank Account Manager** - Create a class called “Account” which will be an abstract class for three other classes called “CheckingAccount”, “SavingsAccount” and “BusinessAccount”. Manage credits and debits from these accounts through an ATM style program.

**Library Catalog** – Create a book class with a title, page count, ISBN and whether or not it is checked out or not. Manage a collection of various books and allow the user to check out books or return books. For added complexity generate a report of those books overdue and any fees. Also allow users to put books on reserve.