Syllabus Exercise/Assignments

|  |  |
| --- | --- |
| Week 2 | Variables & Conditions |

## VARIABLES

- Write a program that asks the user to enter three numbers (use three separate input statements). Create variables called total and average that hold the sum and average of the three numbers and print out the values of total and average.

- A lot of cell phones have tip calculators. Write one. Ask the user for the price of the meal and the percent tip they want to leave. Then print both the tip amount and the total bill with the tip included.

- Write a program that asks the user for a weight in kilograms and converts it to pounds. There are 2.2 pounds in a kilogram.

- Write a program that outputs 100 lines, numbered 1 to 100, each with your name on it. The output should look like the output below.

- Ask the user to enter a number. Print out the square of the number, but use the sep optional argument to print it out in a full sentence that ends in a period.

## CONDITIONALS

- Write a program that asks the user to enter a length in centimeters .If the user enters a negative length, the program should tell the user that the entry is invalid. Otherwise, the program should convert the length to inches and print out the result. There are 2.54 centimeters in an inch.

- Ask the user for a temperature. Then ask them what units, Celsius or Fahrenheit, the temper- ature is in. Your program should convert the temperature to the other unit. The conversions are F = 95 C + 32 and C = 95 (F − 32).

- Ask the user to enter a temperature in Celsius. The program should print a message based on the temperature:

(a) If the temperature is less than -273.15, print that the temperature is invalid because it is below absolute zero.

(b) If it is exactly -273.15, print that the temperature is absolute 0.

(c) If the temperature is between-273.15 and 0,print that the temperature is below freezing.

(d) If it is 0, print that the temperature is at the freezing point.

(e) If it is between 0 and 100, print that the temperature is in the normal range.

(f) If it is 100, print that the temperature is at the boiling point.

(g) If it is above 100, print that the temperature is above the boiling point.

- Write a program that asks the user how many credits they have taken. If they have taken 23 or less, print that the student is a freshman. If they have taken between 24 and 53, print that they are a sophomore. The range for juniors is 54 to 83, and for seniors it is 84 and over.

- A store charges $12 per item if you buy less than 10 items. If you buy between 10 and 99 items, the cost is $10 per item. If you buy 100 or more items, the cost is $7 per item. Write a program that asks the user how many items they are buying and prints the total cost.

- Write a program that asks the user to enter a number and prints out all the divisors of that number. [Hint: the % operator is used to tell if a number is divisible by something.]

- Write a multiplication game program for kids. The program should give the player ten ran- domly generated multiplication questions to do. After each, the program should tell them whether they got it right or wrong and what the correct answer is.

|  |  |
| --- | --- |
| Week 3 | Loops & Functions |

## FOR LOOP

- The Fibonacci numbers are the sequence below, where the first two numbers are 1, and each number thereafter is the sum of the two preceding numbers. Write a program that asks the user how many Fibonacci numbers to print and then prints that many. eg-> 1,1,2,3,5,8,13,21,34,55,89...

- Use a for loop to print a triangle like the one below. Allow the user to specify how high the triangle should be.

\*

\*\*

\*\*\*

\*\*\*\*

- Use a for loop to print a triangle like the one below. Allow the user to specify how high the triangle should be.

\*\*\*\*

\*\*\*

\*\*

\*

- Ask which direction the user wants to count (up or down). If they select up, then ask them for the top number and then count from 1 to that number. If they select down, ask them to enter a number below 20 and then count down from 20 to that number. If they entered something other than up or down, display the message “I don’t understand”.

- Ask the user to enter their name and a number. If the number is less than 10, then display their name that number of times; otherwise display the message “Too high” three times.

- Ask how many people the user wants to invite to a party. If they enter a number below 10, ask for the names and after each name display “[name] has been invited”. If they enter a number which is 10 or higher, display the message “Too many people”.

## WHILE LOOP

- (a) Write a program that uses a while loop (not a for loop) to read through a string and print the characters of the string one-by-one on separate lines.

(b) Modify the program above to print out every second character of the string.

- A good program will make sure that the data its users enter is valid. Write a program that asks the user for a weight and converts it from kilograms to pounds. Whenever the user enters a weight below 0, the program should tell them that their entry is invalid and then ask them again to enter a weight. [Hint: Use a while loop, not an if statement].

- Write a program that asks the user to enter a password. If the user enters the right password, the program should tell them they are logged in to the system. Otherwise, the program should ask them to reenter the password. The user should only get five tries to enter the password, after which point the program should tell them that they are kicked off of the system.

- Recall that, given a string s, s.index('x') returns the index of the first x in s and an error if there is no x.

(a) Write a program that asks the user for a string and a letter. Using a while loop, the program should print the index of the first occurrence of that letter and a message if the string does not contain the letter.

(b) Write the above program using a for/break loop instead of a while loop.

- Write a program that allows the user to enter any number of test scores. The user indicates they are done by entering in a negative number. Print how many of the scores are A’s (90 or above). Also print out the average.

## 

## FUNCTIONS

- Write a function called change\_case that given a string, returns a string with each upper case letter replaced by a lower case letter and vice-versa.

- Write a function called is\_sorted that is given a list and returns True if the list is sorted and False otherwise.

- Write a function called root that is given a number *x* and an integer *n* and returns *x*1/*n*. In the function definition, set the default value of *n* to 2.

- Write a function called one\_away that takes two strings and returns True if the strings are of the same length and differ in exactly one letter, like bike/hike or water/wafer.

- Write a function called primes that is given a number *n* and returns a list of the first *n* primes. Let the default value of *n* be 100.

- Modify the function above so that there is an optional argument called start that allows the list to start at a value other than 2. The function should return the first *n* primes that are greater than or equal to start. The default value of start should be 2.

- Write a function called first\_diff that is given two strings and returns the first location in which the strings differ. If the strings are identical, it should return -1.

- The digital rootof a number *n* is obtained as follows: Add up the digits *n* to get a new number. Add up the digits of that to get another new number. Keep doing this until you get a number that has only one digit. That number is the digital root.For example, if *n* = 45893, we add up the digits to get 4+5+8+9+3 = 29. We then add up the digits of 29 to get 2+9=11. We then add up the digits of 11 to get 1+1=2. Since2has only one digit, 2 is our digital root.Write a function that returns the digital root of an integer *n*.   
   
- Write a function called closest that takes a list of numbers L and a number n and returns the largest element in L that is not larger than n. For instance, if L=[1,6,3,9,11] and n=8, then the function should return 6, because 6 is the closest thing in L to 8 that is not larger than 8. Don’t worry about if all of the things in L are smaller than n.

- Write a function called matches that takes two strings as arguments and returns how many matches there are between the strings. A match is where the two strings have the same char- acter at the same index. For instance, 'python' and 'path' match in the first, third, and fourth characters, so the function should return 3.

|  |  |
| --- | --- |
| Week 4 | Lists & Tuples |

## LISTS

- Write a program that asks the user to enter a list of integers. Do the following:

(a) Print the total number of items in the list.

(b) Print the last item in the list.

(c) Print the list in reverse order.

(d) Print Yes if the list contains a 5 and No otherwise.

(e) Print the number of fives in the list.

(f) Remove the first and last items from the list, sort the remaining items, and print the result.

(g) Print how many integers in the list are less than 5.

- Write a program that generates a list of 20 random numbers between 1 and 100.

(a) Print the list.

(b) Print the average of the elements in the list.

(c) Print the largest and smallest values in the list.

(d) Print the second largest and second smallest entries in the list

(e) Print how many even numbers are in the list.

- Start with the list [8,9,10]. Do the following:

(a) Set the second entry (index 1) to 17

(b) Add4,5,and 6 to the end of the list

(c) Remove the first entry from the list

(d) Sort the list

(e) Double the list

(f) Insert 25 at index 3

The final list should equal [4,5,6,25,10,17,4,5,6,10,17]

- Ask the user to enter a list containing numbers between 1 and 12. Then replace all of the entries in the list that are greater than 10 with 10.

- Write a program that takes any two lists L and M of the same size and adds their elements together to form a new list N whose elements are sums of the corresponding elements in L and M. For instance, if L=[3,1,4] and M=[1,5,9], then N should equal [4,6,13].

- Write a program that generates 100 random integers that are either 0 or 1. Then find the longest run of zeros, the largest number of zeros in a row. For instance, the longest run of zeros in [1,0,1,1,0,0,0,0,1,0,0] is 4.

- Write a program that removes any repeated items from a list so that each item appears at most once. For instance, the list [1,1,2,3,4,3,0,0] would become [1,2,3,4,0].

- Write a simple quiz game that has a list of ten questions and a list of answers to those ques- tions. The game should give the player four randomly selected questions to answer. It should ask the questions one-by-one, and tell the player whether they got the question right or wrong. At the end it should print out how many out of four they got right.

- Write a program that asks the user to enter a length in feet. The program should then give the user the option to convert from feet into inches, yards, miles, millimeters, centimeters, meters, or kilometers. Say if the user enters a 1, then the program converts to inches, if they enter a 2, then the program converts to yards, etc. While this can be done with if statements, it is much shorter with lists and it is also easier to add new conversions if you use lists.

## TUPLES

|  |  |
| --- | --- |
| Week 5 | Dictionaries & Sets |

## DICTIONARIES

- Dictionaries provide a convenient way to store structured data. Here is an example dictio- nary:

Write a program that reads through any dictionary like this and prints the following:

|  |
| --- |
| data=[{'name':'Kwadwo', 'phone':'555-1414', 'email':kwadwo@mail.net'},{'name':'Daniel', 'phone':'555-1618', 'email':'daniel@mail.net'}, {'name':'Akwasi', 'phone':'555-3141', 'email':''}, {'name':'Andy', 'phone':'555-2718', 'email':'andy@mail.net'}] |

(a) All the users whose phone number ends in an 8

(b) All the users that don’t have an email address listed

- Write a program that repeatedly asks the user to enter product names and prices. Store all of these in a dictionary whose keys are the product names and whose values are the prices. When the user is done entering products and prices, allow them to repeatedly enter a product name and print the corresponding price or a message if the product is not in the dictionary.

- Using the dictionary created in the previous problem, allow the user to enter a dollar amount and print out all the products whose price is less than that amount.

- For this problem, use the dictionary in the table below, whose keys are month names and whose values are the number of days in the corresponding months.

|  |
| --- |
| days = {'January':31, 'February':28, 'March':31, 'April':30, 'May':31, 'June':30, 'July':31, 'August':31, 'September':30, 'October':31, 'November':30, 'December':31} |

(a) Ask the user to enter a month name and use the dictionary to tell them how many days are in the month.

(b) Print out all of the keys in alphabetical order.

(c) Print out all of the months with 31 days.

(d) Print out the (key-value) pairs sorted by the number of days in each month

- Write a program that uses a dictionary that contains ten user names and passwords. The program should ask the user to enter their username and password. If the username is not in the dictionary, the program should indicate that the person is not a valid user of the system. If the username is in the dictionary, but the user does not enter the right password, the program should say that the password is invalid. If the password is correct, then the program should tell the user that they are now logged in to the system.[hints: Create your own data or make it dynamic just so the user can create his/her username and password and store them in a dictionary before starting to write your program.(optional)]

- Repeatedly ask the user to enter a team name and the how many games the team won and how many they lost. Store this information in a dictionary where the keys are the team names and the values are lists of the form [wins, losses].

(a) Using the dictionary created above, allow the user to enter a team name and print out the team’s winning percentage.

(b) Using the dictionary, create a list whose entries are the number of wins of each team.

(c) Using the dictionary, create a list of all those teams that have winning records.

- Repeatedly ask the user to enter game scores in a format like *team1 - team2*. Store this information in a dictionary where the keys are the team names and the values are lists of the form [*wins*, *losses*].

## SETS

|  |  |
| --- | --- |
| Week 6 | Object-oriented programming |

## OBJECT-ORIENTED PROGRAMMING

- Write a class called Password\_manager. The class should have a list called old\_passwords that holds all of the user’s past passwords. The last item of the list is the user’s current password. There should be a method called get\_password that returns the current password and a method called set\_password that sets the user’s password. The set\_password method should only change the password if the attempted password is different from all the user’s past passwords. Finally, create a method called is\_correct that receives a string and returns a boolean True or False depending on whether the string is equal to the current password or not.

- Write a class called Wordplay. It should have a field that holds a list of words. The user of the class should pass the list of words they want to use to the class. There should be the following methods:

• words\_with\_length(length) — returns a list of all the words of length length

• starts\_with(s) — returns a list of all the words that start with s

• ends\_with(s) — returns a list of all the words that end with s

• palindromes() — returns a list of all the palindromes in the list

• only(L) — returns a list of the words that contain only those letters in L

• avoids(L) — returns a list of the words that contain none of the letters in L

- Write a class called Product. The class should have fields called name,amount, and price, holding the product’s name, the number of items of that product in stock, and the regular price of the product. There should be a method get\_price that receives the number of items to be bought and returns a the cost of buying that many items, where the regular price is charged for orders of less than 10 items, a 10% discount is applied for orders of between 10 and 99 items, and a 20% discount is applied for orders of 100 or more items. There should also be a method called make\_purchase that receives the number of items to be bought and decreases amount by that much.

- Write a class called Converter. The user will pass a length and a unit when declaring an object from the class—for example, c = Converter(9,'inches'). The possible units are inches, feet, yards, miles, kilometers, meters, centimeters, and millimeters. For each of these units there should be a method that returns the length converted into those units. For exam- ple, using the Converter object created above, the user could call c.feet() and should get 0.75 as the result.