Beginner Problems

Problem 1: Sort Integers

100 points
Filename: prob01 *(e.g. prob01.js, prob01.c, prob01.cpp, prob01.java, prob01.py2, prob01.py3)*

Description
Given a single line of input that contains N integers (where N will always be 1 to 100) each separated by a single space, remove any duplicate integers and output the integers sorted from smallest to largest, each separated by a single space.

Sample Input

```
1 9 2 8 8
```

Sample Output

```
1 2 8 9
```

Sample Input

```
20 1 2 3 15 4 5 6 10 7 8 9 19
```

Sample Output

```
1 2 3 4 5 6 7 8 9 10 15 19 20
```

Sample Input

```
111 222 333 444 555 666 777 888 999
```

Sample Output

```
111 222 333 444 555 666 777 888 999
```

Consider These Edge Cases

- Does your program properly handle negative integers?
- Does your program properly handle a single integer?
- Does your program properly handle a list of integers if they are all the same value?

Learn More
Programming languages have built in structures you can use to more easily solve this problem. Consider storing the values as the keys for an array and then sorting the array by key value. Since each key in the array must be unique, this method will automatically eliminate duplicate values.

Problem 2: Largest Integers

100 points
Filename: prob02 *(e.g. prob02.js, prob02.c, prob02.cpp, prob02.java, prob02.py2, prob02.py3)*

Description
Given a single line of input that contains N integers (where N will always be 1 to 100) each separated by a single space, remove any duplicate integers and output the 3 largest value integers sorted from largest to smallest, each separated by a single space.

Sample Input

```
1 9 2 8 8
```

Sample Output

```
9 8 2
```

Sample Input

```
20 1 2 3 15 4 5 6 10 7 8 9 19
```

Sample Output

```
20 19 15
```

Sample Input

```
111 222 333 444 555 666 777 888 999
```

Sample Output

```
999 888 777
```

Consider These Edge Cases

- Does your program properly handle negative integers?
- Does your program properly handle a list of 1 or 2 integers?
- Does your program properly handle a list of integers if there are only 1 or 2 unique integers?

Learn More
This problem builds on the prior problem. Rather than sorting the array by key value and outputting all of the keys, you could reverse sort the array by key value, then output the first 3 keys.

Problem 3: Duplicate Integers

100 points
Filename: prob03 *(e.g. prob03.js, prob03.c, prob03.cpp, prob03.java, prob03.py2, prob03.py3)*

Description
We have received a random list of page numbers that we need to review. We would like to review the pages in order and we would like to avoid reviewing any pages twice.

Given a single line of input that contains N positive integers representing page numbers (where N will always be 1 to 100) each separated by a single space, remove any duplicate page numbers and return the list of page numbers in ascending order, separated by a comma and a space.

Sample Input

```
9 2 10 10 1 9 7 9 9 7 9 4 10 1 9 9 9 6 9 4 9 8
```

Sample Output

```
1, 2, 4, 6, 7, 8, 9, 10
```

Sample Input

```
4 9 4 8 10 10 2 6 2 7 9 4 4 8 3 7 7 10 2 6
```

Sample Output

```
2, 3, 4, 6, 7, 8, 9, 10
```

Consider These Edge Cases

- Does your program properly handle a single integer?
- Does your program properly handle a list of integers if they are all the same value?

Learn More
This problem builds on the first problem. Rather than joining each key by a space (“ “), you could join each element of an array by a comma and space (“, “).

Problem 4: Does the Boat Float?

100 points
Filename: prob04 *(e.g. prob04.js, prob04.c, prob04.cpp, prob04.java, prob04.py2, prob04.py3)*

Description
How buoyant your boat is depends, in part, on how large it is. Buoyancy is a balancing act between buoyant force, which pushes your boat out of the water, and weight, which sinks your boat into the water. The lighter your boat -- just you and a fishing pole -- the lower the buoyant force required to keep your boat above the water and the more buoyant the boat. As you add things --- a cooler full of food and drink, a stereo, tackle boxes and fishing buddies --- the boat becomes heavier, rides lower in the water and is less buoyant.

Formulas
Volume of Boat in Cubic Feet
V = Length (feet) x Width (feet) x Height (feet)

Desired Positive Buoyancy (%)
B = Decide how much of the boat, also called the amount of positive buoyancy, you want your boat to have out of the water. If you want to have 50 percent of your boat's sides out of the water, use 0.50. If you want 75 percent of your boat's sides out of the water, use 0.75.

Weight of 1 Cubic Foot of Fresh Water
C = 62.4 pounds

Number of pounds of buoyant force keeping half of your boat out of the water.
F = V x B x C

Given the boat's volume (V) in cubic feet, the boat's weight in pounds, and the boat's desired positive buoyancy percentage (B), determine if the boat's actual buoyancy is same or greater than the desired buoyancy. If so, return the string "safe" if same or greater, else return the string "heavy".

Sample Input

```
54 1500 0.50
```

Sample Output

```
safe
```

Sample Input

```
54 2000 0.50
```

Sample Output

```
heavy
```

Sample Calculation
54 x 0.50 x 62.4 = 1684.8 weight supported at 50%, 1500 is less than 1684.8 so actual buoyancy is greater than desired buoyancy of 50%, which means the boat is "safe".

Problem 5: Balanced Letters

100 points
Filename: prob05 *(e.g. prob05.js, prob05.c, prob05.cpp, prob05.java, prob05.py2, prob05.py3)*

Description
Given a string, does the string contain an equal number of uppercase and lowercase letters? Ignore whitespace, numbers, and punctuation. Return the string “true” if balanced or the string “false” if not balanced.

Sample Input

```
AaBbCc
```

Sample Output

```
true
```

Sample Input

```
TWO plus TWO equals FIVE
```

Sample Output

```
true
```

Sample Input

```
Go! Go! Go! Go! Go!
```

Sample Output

```
true
```

Sample Input

```
The quick brown fox jumps over the lazy dog
```

Sample Output

```
false
```

Consider These Edge Cases

- Does your program properly handle a string with numbers only? (e.g. 0 upper = 0 lower = true)

Learn More
Loop through each character in the string and determine if the character is uppercase or lowercase.

Programming languages vary in how they compare characters. Some languages can compare a letter to other letters (e.g. if “A” <= letter <= “Z”) while other languages require that you convert the letter to a numerical value (e.g. if 100 <= value && value <= 125). How does your language compare characters?

Intermediate Problems

Problem 6: Closest Integers

150 points
Filename: prob06 *(e.g. prob06.js, prob06.c, prob06.cpp, prob06.java, prob06.py2, prob06.py3)*

Description
Given a single line of input that contains N integers (where N will always be 1 to 100) each separated by a single space, output the two smallest integers that are closest together, separated by a single space. The smallest number should be output first. Each integer value will be between -100 and 100.

Sample Input

```
5 4 3 2 1
```

Sample Output

```
1 2
```

Sample Input

```
-1 2 5 10 -3 -5
```

Sample Output

```
-5 -3
```

Sample Input

```
6 8 1 3 12 15
```

Sample Output

```
1 3
```

Consider These Edge Cases

- Does your program properly handle a single integer? (e.g. output only that integer)
- Does your program properly handle two integers? (e.g. output both integers, smallest first)
- Does your program properly handle a list of integers if they are all the same value? (e.g. output the same two integers)

Learn More
This problem builds on the first few integer problems. You must allow for duplicate values.

Problem 7: Decode Vanity Phone Numbers

150 points
Filename: prob07 *(e.g. prob07.js, prob07.c, prob07.cpp, prob07.java, prob07.py2, prob07.py3)*

Description
Input contains a single phone number always formatted with the number 1, followed by a dash, followed by a combination of 10 alphanumeric characters and any number of dashes.

Sample Input

```
1-800-555-1212
```

Sample Output

```
1-800-555-1212
```

Sample Input

```
1-866-4-Coders
```

Sample Output

```
1-866-4-26337
```

Sample Input

```
1-866-HACKERS
```

Sample Output

```
1-866-4225377
```

Sample Input

```
1-888-CODE-FTW
```

Sample Output

```
1-888-2633-389
```

Consider These Edge Cases

- Does your program properly handle a dash between each number or letter?
- Does your program properly handle multiple consecutive dashes? How about trailing dashes?

Learn More
Consider using a key/value map or array to store the dialpad letters and their associated numbers.

Problem 8: Sam's Secrets

150 points
Filename: prob08 *(e.g. prob08.js, prob08.c, prob08.cpp, prob08.java, prob08.py2, prob08.py3)*

Description
Sam builds forts with super secret security. To enter Sam's forts, guests must type a secret passphrase. The passphrase must contain 3 separate strings, and each string must begin or end with the letter "s".

Given an input that contains a secret passphrase, your program should output the string "welcome" if the passphrase is valid or output the string "denied" if the passphrase is invalid.

Sample Input
```
Sam invites guests
```
Sample Output
```
welcome
```

Sample Input
```
Sam requests cookies
```
Sample Output
```
welcome
```

Sample Input
```
Sam sent me
```
Sample Output
```
denied
```

Sample Input
```
Sam requests sugar cookies
```
Sample Output
```
denied
```

Consider These Edge Cases
- Does your program properly handle a passphrase with single character strings?
- Does your program properly handle a passphrase with fewer or more than 3 strings?

Learn More
Rather than looping through each character in each word, check to see if your programming language will allow you to read the first and last character of each word. Most languages have a shortcut for this.

Problem 9: Code Words

150 points
Filename: prob09 *(e.g. prob09.js, prob09.c, prob09.cpp, prob09.java, prob09.py2, prob09.py3)*

Description
Sam sends encoded messages. Each message begins with a list of code words and the corresponding decoded words. You need to decode the message by replacing the code word with the decoded word.

You will receive an input with two lines. The first line is a string containing up to 50 word pairs, with each word separated by a single space. The second line is a second string containing an encoded message, with each word separated by a space. Decode the message and output the decoded message as a string. If the code word does not appear in the word pairs, discard the word. If the code word appears multiple times in the word pairs, only use the first code word and decoded word pair.

Assume the input will include at least one word pair and an encoded message with at least one word.

Sample Input

```
potato world purple hello
purple potato
```

Sample Output

```
hello world
```

Sample Input

```
your the funny quick white brown cat fox hides jumps under over your that
big lazy chair dog
your funny white cat hides under your big chair
```

Sample Output

```
the quick brown fox jumps over the lazy dog
```

Consider These Edge Cases

- Does your program properly handle a code word that does not appear in the word pairs?

Learn More
Consider using a key/value map or array approach as suggested in Problem 7. Map entire words instead of single characters.

Advanced Problems

Problem 10: Calculator

200 points
Filename: prob10 *(e.g. prob10.js, prob10.c, prob10.cpp, prob10.java, prob10.py2, prob10.py3)*

Description
Parse the input string that contains one or more positive integers, each separated by a single plus sign (+) or minus sign (-), and ending with a single equals sign (=). Perform addition and subtraction each time you encounter a plus or minus sign and return the result when you encounter the equals sign.

The input will always begin with a positive integer and end with an equal sign. The input will only contain positive integer(s), optional non-sequential plus sign(s) and/or minus sign(s), and end with a single equals sign.

Sample Input

```
1+2-3+4-5+6=
```

Sample Output

```
5
```

Sample Input

```
1000-5000=
```

Sample Output

```
-4000
```

Sample Input

```
10+20-30+40-50+60=
```

Sample Output

```
50
```

Sample Input

```
500=
```

Sample Output

```
500
```

Consider These Edge Cases

- Does your program properly handle a single integer followed by an equals sign?
- Does your program properly handle negative results?

Learn More
Consider tokenizing the string and analyzing each token. Can you think of any ways to split the integers, plus signs, and minus signs, and equals sign into separate tokens without analyzing each individual character?

Problem 11: Page Ranges

200 points
Filename: prob11 *(e.g. prob11.js, prob11.c, prob11.cpp, prob11.java, prob11.py2, prob11.py3)*

Description
We have received a random list of page numbers that we need to review. We would like to review the pages in order and we would like to avoid reviewing any pages twice. We would also like to make our final list appear as short as possible, by listing page ranges instead of individual pages whenever possible.

Given a single line of input that contains N positive integers representing page numbers (where N will always be 1 to 100) each separated by a single space, remove any duplicate page numbers and return the list of page numbers in ascending order, separated by a comma and a space. Furthermore, condense two or more consecutive page numbers into a page range where the first page number and last page number in the range are separated by a dash (e.g. “2,3,4” would condense to “2-4”).

Sample Input

```
9 2 10 10 1 9 7 9 9 7 9 4 10 1 9 9 9 6 9 4 9 8
```

Sample Output

```
1, 2, 4, 6-10
```

Sample Input

```
7 18 4 10 11 13 3 18 1 8 6
```

Sample Output

```
1, 3-4, 6-8, 10-11, 13, 18
```

Consider These Edge Cases

- Does your program properly handle a single integer?
- Does your program properly handle a list of integers if they are all the same value?

Learn More
This problem builds on the Problem 3. What conditions would you need to check for as you loop through your sorted list of unique page numbers to identify consecutive pages?

Problem 12: Ransom Note

200 points
Filename: prob12 *(e.g. prob12.js, prob12.c, prob12.cpp, prob12.java, prob12.py2, prob12.py3)*

Description

Given two input strings, determine if the first string from a magazine cover contains all of the letters you will need to create the ransom note in the second string. This is case sensitive! Ignore spaces. Do NOT ignore numbers or punctuation.

Sample Input

```
Trouble attracting talent? MAYBE IT'S YOU.
Meet US at nOon.
```

Sample Output

```
true
```

Sample Input

```
Trouble attracting talent? MAYBE IT'S YOU.
Meet us at noon.
```

Sample Output

```
false
```

Sample Input

```
HOW THE BIG DOGS EAT
GO TO THE EAST
```

Sample Output

```
false
```

Consider These Edge Cases

- Does your program properly handle a ransom note with more spaces than the magazine cover?

Learn More

Most programming languages offer a simple way to add and remove items from a list or array.

Prepared by Jason Klein
Magazine Text from covers of Biz417 issues published in 2019 https://www.biz417.com/
Image generated by Ransomizer http://www.ransomizer.com/

Problem 13: Encode Messages

200 points
Filename: prob13 *(e.g. prob13.js, prob13.c, prob13.cpp, prob13.java, prob13.py2, prob13.py3)*

Description
Given a string containing between 1 and 100 characters, shift each alphabetical letter (from A to Z) by 13 characters. Do not modify any numbers or punctuation in the input string.

The solution is case sensitive! The shifted letters you output should be the same case as the original letters in the input.

Sample Input

```
What is put on a table, cut, but never eaten?
```

Sample Output

```
Jung vf chg ba n gnoyr, phg, ohg arire rngra?
```

Sample Input

```
A pack of cards.
```

Sample Output

```
N cnpx bs pneqf.
```

Sample Input

```
N cnpx bs pneqf.
```

Sample Output

```
A pack of cards.
```

Sample Input

```
1234567890
```

Sample Output

```
1234567890
```

Sample Input

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
```

Sample Output

```
NOPQRSTUVWXYZABCDEFGHIJKLM
```

Sample Input

```
abcdefghijklmnopqrstuvwxyz
```

Sample Output

```
nopqrstuvwxyzabcdefghijklm
```

Learn More
How many letters are in the alphabet? Most programming languages allow you to convert an alphabetic character to a numeric value. Could you increase or decrease the numeric value by 13, then convert the numeric value back to an alphabetic character?

Prepared by Jason Klein

Index