Hour of Code 2021

Wednesday December 15th, 2021
Bethlehem CSHS

TABLE OF CONTENTS

Written By:
Charlie He, Nikhilesh Radosevich

Problem 1: Hello World!

Everything starts off with a greeting. Why don't we greet the world with a print() statement?

Task: Create a program that prints hello world

Example Run #1:

```
Hello World
```

Part 2:

Now ask for the user's name and print Hello followed by the user's name.

Example Run #1:

```
What is your name?
Bob
Hello Bob
```

PROBLEM 1 SOLUTION:

```
print("Hello World")
```

PROBLEM 1 PART 2 SOLUTION:

```
print("What is your name?")
name = input()
print("Hello " + name)
```

Problem 2: Square That

Big Al wants to know what the square of different numbers are, help him out by writing some code.

Task: Create a program that prints the square of an inputted number:

Example Run #1:

```
What number should be squared?
 6
36
```

Example Run #2:

```
What number should be squared?
 15
225
```

PROBLEM 2 SOLUTION:

```
print("What number should be squared?")
num = int(input())
print(num**2)
```

Problem 3: Even or Odd

Gordon, Nikhil and Charlie are playing monopoly. The game is taking too long so they decide to roll the dice to determine the winner.

Task:
Create a program that takes in two numbers and determines if they are odd or even.

1. If both numbers are even, print *Gordon wins*.
2. If both numbers are odd, print *Charlie wins*.
3. If one number is odd, and the other is even, print *Nikhil wins*.

Examples
Example Run #1:

```
Number 1:
 2
Number 2:
 6
Gordon wins
```

Example Run #2:

```
Number 1:
 3
Number 2:
 1
Charlie wins
```

Example Run #3

```
Number 1:
 2
Number 2:
 5
Nikhil wins
```

SOLUTION:

```
print("Number 1:")
n1 = int(input())
print("Number 2:")
n2 = int(input())

if n1 % 2 == 0:
  n1even = True
else:
  n1even = False
if n2 % 2 == 0:
  n2even = True
else:
  n2even = False

if n1even and n2even:
  print("Gordon wins")
elif not n1even and not n2even:
  print("Charlie wins")
else:
  print("Nikhil wins")

#Alternate Solution:

print("Number 1:")
n1 = int(input())
print("Number 2:")
n2 = int(input())

if (n1 * n2) % 2 == 1:
  print("Charlie wins")
elif (n1 + n2) % 2 == 0:
  print("Gordon wins")
else:
  print("Nikhil wins")
```

Problem 4: Roman Numerals

Background Information:

Symbol	I	V	X	L	C	D	M
Value	1	5	10	50	100	500	1000

A number's Roman numeral value is written by representing each power of ten (thousands, hundreds, tens, and ones) from left to right in that order. Generally, the number's representation is the concatenation of letters, where each letter's value, when added together, sums to the original number. The exception occurs when a symbol with a smaller value precedes a symbol with a larger value; in this case, you subtract the smaller value from the larger value. For example XC is the proper way to represent 90 instead of LXXXX. In fact, no representation of input for this problem will have any one symbol repeated four or more times. You may only use the following subtraction representations:

IV IX XL XC CD and CM
for 4 9 40 90 400 and 900

Programming Problem:
Input: A positive integer N where 1 ≤ N ≤ 3999
Output: The roman numeral representation, in all capital letters.

Example 1:

```
1001
MI
```

Example 2:

```
3005
MMMV
```

Example 3:

```
51
LI
```

Problem originally adopted from Siena Coding Comp 2021.

PROBLEM 4 SOLUTION:

```
number = int(input())
fullN = number
roman = ""
while number >= 1000:
  roman = roman+"M"
  number = number-1000
if number >= 900:
  roman = roman +"CM"
  number = number-900
while number >= 500:
  roman = roman+"D"
  number = number-500
if number >= 400:
  roman = roman +"CD"
  number = number-400
while number >= 100:
  roman = roman+"C"
  number = number-100
if number >= 90:
  roman = roman +"XC"
  number = number-90
while number >= 50:
  roman = roman+"L"
  number = number-50
if number >= 40:
  roman = roman +"XL"
  number = number-40
while number >= 10:
  roman = roman+"X"
  number = number-10
if number >= 9:
  roman = roman +"IX"
  number = number-9
while number >= 5:
  roman = roman+"V"
  number = number-5
if number >= 4:
  roman = roman +"IV"
  number = number-4
while number >= 1:
  roman = roman+"I"
  number = number-1
print(roman)
```

Problem 5: Christmas Tree!

Let's get festive! Take in the amount of tree layers, and then create a beautiful Christmas tree!

Task:
INPUT: Amount of layers (integer)
OUTPUT: Symmetric pyramid of stars with one start at the top, then incrementing by two each line with required spacing. The height of the trunk is half of the amount of layers, rounded up.

Examples
Example Run #1:

```
How many layers? 2
 *
***
 |
```

Example Run #2:

```
How many layers?
4
   *
  ***
 *****
*******
   |
   |
```

Example Run #3

```
How many layers?
7
      *
     ***
    *****
   *******
  *********
 ***********
*************
      |
      |
      |
      |
```

PROBLEM 5 SOLUTION:

```
layers = int(input("How many layers? "))
  for layer in range(0, layers):
    print(" " * (layers - 1 - layer) + "*" * (1 + 2 * layer))
  for trunk in range(0, 1 + int(layer / 2)):
    print(" " * (layers - 1) + "|")
```


Problem 6: Calculus?

Why don't we get some math applications with this question? Let's try to get the area under any linear graph, between two x constraints.

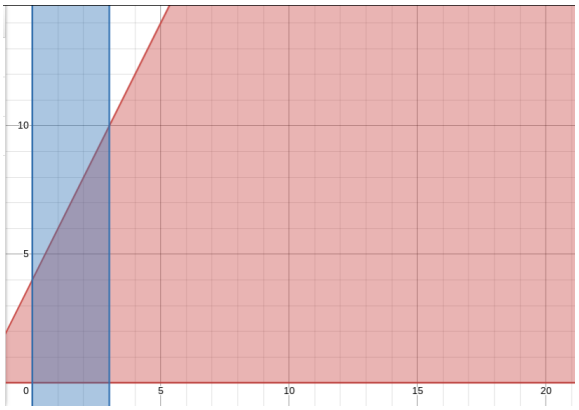
Several notes:

- The area is constrained by the x-axis (graph x = 0), the lower and upper x bounds (graphs y = Xlower and y = Xupper), and the linear graph (graph y = (slope)x + (y-intercept))
- If the area exists below the x-axis (graph x= 0), then the area is negative (subtracted from the total area, aka integral)
- If the upper x bound is less than the lower x bound, then the integral is negative compared to if the bounds were in reverse (an integral with bounds a to b is equal to the negative integral with bounds b to a)
- The solution is finding the definite integral of each linear graph.

HINT: Try using the trapezoid area formula! The only prerequisite for this problem is geometry; no calculus knowledge needed.

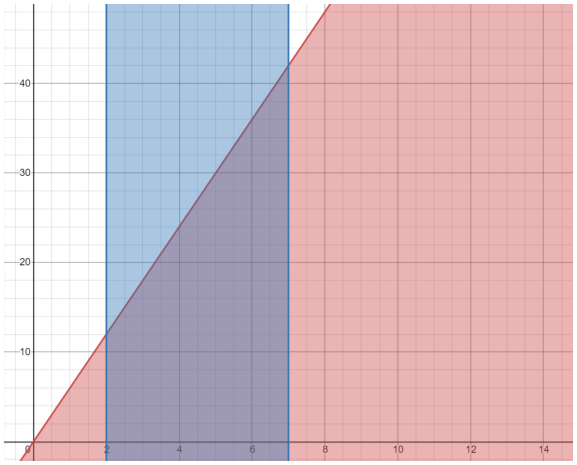
Task:
INPUT: Slope (integer/float), y-intercept (integer/float), lower x-constraint, upper x-constraint
OUTPUT: Printed string in the format "The integral of y = (Slope)x + (y-intercept) from (lower x-constraint) to (upper x-constraint) is (integral)."

Examples



Example Run #1:
The trapezoid with the area under y = 2x + 4 between the bounds x = 0 and x = 3 is highlighted in the intersection.

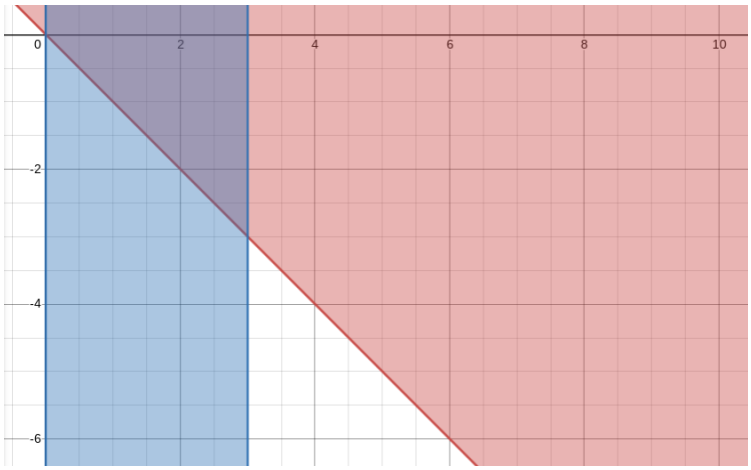
```
What is the slope of the line? 2
What is the y-intercept of the line? 4
y = 2x + 4
What is the lower x-constraint? 0
What is the upper x-constraint? 3
The integral of y = 2x + 4 from 0 to 3 is 21.
```



Example Run #2: *When the bounds are reversed, the area is the negative of the original area.*

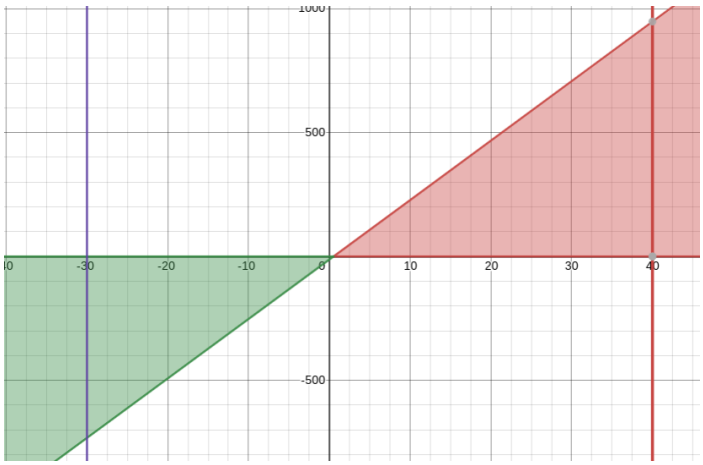
```
What is the slope of the line? 6
What is the y-intercept of the line? 0
y = 6x + 0
What is the lower x-constraint? 7
What is the upper x-constraint? 2
```

```
The integral of y = 6x + 0 from 7 to 2 is -135.0.
```



Example Run #3:
Notice how when the graph dips below the x-axis, the area is negative.

```
What is the slope of the line? -1
What is the y-intercept of the line? 0
y = -1x + 0
What is the lower x-constraint? 0
What is the upper x-constraint? 3
The integral of y = -1x + 0 from 0 to 3 is -4.5.
```



Example Run #4: *Notice how the area below is subtracted from the area above.*

```
What is the slope of the line? 24
What is the y-intercept of the line? -12
```

```
y = 24x + -12
What is the lower x-constraint? -30
What is the upper x-constraint? 40
The integral of y = 24x + -12 from -30 to 40 is 7560.0.
```

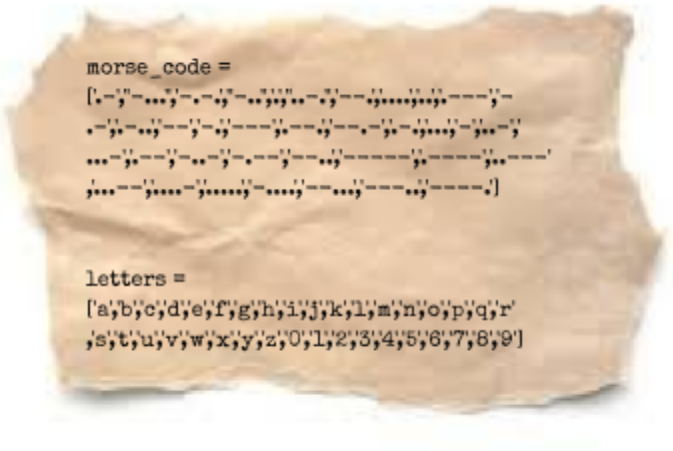
SOLUTION:

```
slope = int(input("What is the slope of the line? "))
yInt = int(input("What is the y-intercept of the line? "))
equation = "y = " + str(slope) + "x + " + str(yInt)
print(equation)
xL = int(input("What is the lower x-constraint? "))
xU = int(input("What is the upper x-constraint? "))
integral = (1 / 2) * ((slope * xL + yInt) + (slope * xU + yInt)) * (xU -
xL)
print("The integral of", equation, "from", str(xL), "to", str(xU), "is",
str(integral) + ".")
```

Problem 7: Morse Code Translator

… ___ … _._.__ You and a few other agents have been sent back to the early 20th century, and you are undercover as an employee of communications. Being inexperienced, not all of them know morse code, and they need to learn it quickly to avoid suspicion -- or why don't we cheat a bit? With your computer science knowledge and a device that converts morse code and thought words to a string and a string to an electrical impulse that automatically makes you tap the button for transmission, you can solve this issue with an automatic solution. Quick! We only have an hour before our shift starts!

Other than basic necessities and your false identity articles, you have only a scrap of paper that has the morse code conversions; your technology, with oversight on morse code translation, can translate visual text to typed text:



Task:
INPUT: String input for each mode - "encode" or "decode", string input in english for mode one or morse code for mode two.
OUTPUT: String output for morse code mode one or english code for mode two. You should remove any non-letter characters other than spaces. All output should be in lowercase.
The space between symbols (dots and dashes) of the same letter is 1 time unit. The space between letters is 3 time units. The space between words is 7 time units.

Resource:
In order to save you time, we are providing the following constants to you which you can copy and paste into your code.

```
MORSE_CODE =
['.-',"-...",'-.-.',"-..",'.',"..-.",'--.','....','..','.---','-.-','.-..',
'--','-.','---','.--.','--.-','.-.','...','-','..-','...-','.--','-..-','-.
--','--..','-----','.----','..---','...--','....-','.....','-....','--...',
'---..','----.']
LETTERS =
['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s
','t','u','v','w','x','y','z','0','1','2','3','4','5','6','7','8','9']
```

Examples
Example Run #1:

```
What would you like to do? (encode/decode): encode
Enter your message: one if by land, two if by sea

--- -. .   .. ..-.   -... -.--   .-.. .- -. -..   - .-- ---   .. ...-
-... -.--   ... . .-
```

Example Run #2:

```
What would you like to do? (encode/decode): decode
Enter your message: --- -. .   .. ..-.   -... -.--   .-.. .- -. -..   - .--
---   .. ..-.   -... -.--   ... . .-

one if by land two if by sea
```

Example Run #3

```
What would you like to do? (encode/decode): encode
Enter your message: Remember to bring this decoder with us if we head back
in time again!

. -- . -- -... . .-.   - ---   -... .-. .. -. --.   - .... .. ...   -.. .
-.-. --- -.. . .-.   .-- .. - ....   ..- ...   .. ..-.   .-- .   .... . .-
-..   -... .- -.-. -.-   .. -.   - .. -- .   .- --. .- .. -.
```

PROBLEM 7 SOLUTION:

```
MORSE_CODE =
['.-',"-...",'-.-.',"-..",'.',"..-.",'--.','....','..','.---','-.-','.-..',
'--','-.','---','.--.','--.-','.-.','...','-','..-','...-','.--','-..-','-.
--','--..','-----','.----','..---','...--','....-','.....','-....','--...',
'---..','----.']
letters =
['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q','r','s
','t','u','v','w','x','y','z','0','1','2','3','4','5','6','7','8','9']

action = input("What would you like to do? (encode/decode)")
message = input("Enter your message: ")
output = ""
if action == "encode":
    for character in message:
        if character in letters:
            output += morse_code[letters.index(character)] + " "
        elif character == " ":
            output += "  "
    print("\n"+output)
elif action == "decode":
    morse_list = message.split(" ")
    for character in morse_list:
        if character == "":
            output += " "
        else:
            output += letters[morse_code.index(character)]
    print("\n"+output)
```