Frankfurt University of Applied Sciences Programming Day Contest 2022

Benedikt Zundel and Contributors

Decembter 8th 2022

Contents

1 Introduction **3**

2 Problem Overview **4**

3 Solutions **5**
3.1 A: Coffee Cup Combo 5
3.2 B: Estimating the Area of a Circle 5
3.3 C: Fun House 6
3.4 D: Espresso! 7
3.5 E: Pea Soup and Pancakes! 7
3.6 F: Math Trade 7
3.7 G: Love Polygon 8
3.8 H: Santa Klas 8
3.9 I: Triangle Ornaments 8
3.10 J: Sacred Texts 8
3.11 K: Spelling Bee 9
3.12 L: Fifty Shades of Pink 9

1 Introduction

This document serves as a writeup for the Frankfurt University of Applied Sciences Programming Day Contest hosted by Prof. Dr. Logofatu on December 8^{th} 2022. It contains an outline of the problems presented during the contest with information about difficulty[1] for revision and demonstrates a possible way of solving them (in Python).

A repository containing all the solutions contained in this document and the document itself can be found at https://github.com/bezunyl/frauas-programming-day-2022. Contributing is highly encouraged, instructions to doing so can be found in the repository's readme file.

[1] The difficulty of the problems may change [1]. Given level corresponds to the difficulty viewed on 15^{th} of December 2022.

2 Problem Overview

According to Kattis' Terms of Service [2] it is prohibited to copy any content displayed on their site, due to which I cannot display the problems and their descriptions in full within this document. However, the below table contains the name and difficulty to each questions of the contest and a link under which it can be found on the official Kattis webpage.

Tag	Name	Difficulty	Link
A	Coffee Cup Combo	1.5 Easy	https://open.kattis.com/problems/coffeecupcombo
B	Estimating the Area of a Circle	1.5 Easy	https://open.kattis.com/problems/estimatingtheareaofacircle
C	Fun House	1.9 Easy	https://open.kattis.com/problems/funhouse
D	Espresso!	2.1 Easy	https://open.kattis.com/problems/espresso
E	Pea Soup and Pancakes	2.2 Easy	https://open.kattis.com/problems/peasoup
F	Math Trade	3.9 Medium	https://open.kattis.com/problems/mathtrade
G	Love Polygon	n.a.	n.a.
H	Santa Klas	2.8 Medium	https://open.kattis.com/problems/santaklas
I	Triangle Ornaments	n.a.	n.a.
J	Sacred Texts	3.9 Medium	https://open.kattis.com/problems/sacredtexts
K	Spelling Bee	2.1 Easy	https://open.kattis.com/problems/spellingbee
L	Fifty Shades of Pink	2.1 Easy	https://open.kattis.com/problems/fiftyshades

3 Solutions

3.1 A: Coffee Cup Combo

In this problem we help a student figure out how many lectures she can stay awake for, based on the premise that she can only stay awake if she drinks one coffee per lecture. We are given the amount of lectures that the student attends on a single day and information for each lecture about whether or not the lecture hall contains a coffee machine to refill her two cups.

This leaves us with a fairly straightforward problem: We have to count how many lectures she can stay awake for by tracking the amount of full coffee cups she has. To do so, we view all possible situations. Either there is a coffee machine in the lecture hall, or there isn't (case 1 and 0, respectively). In case 1, we do not worry if she currently has any cups of coffee left, as she can fill both cups before the lecture, drink on during the lecture and finally fill both cups up again after the lecture. Therefore, we just increment the count of lecutres she can stay awake for and reset her remaining full cups to two. In case 0 there are two branches we have to consider: Either she has at least one cup of coffee left (case 0_y) or both her cups are empty (case 0_n). In case 0_y we simply increment the count of lectures she does not sleep in and decrement the variable keeping track of how many cups of coffee she has left. Case 0_n is simple, as we do nothing.

```
n = input()
ls = input()

cups = 0
count = 0

for i in ls:
    if i == "1":
        cups = 2
        count += 1

    if i == "0":
        if cups > 0:
            count += 1
            cups -= 1

print(count)
```

Listing 1: "Coffee Cup Combo Solution"

The above is an implementation of what was explained prior. First we receive the two lines of input as store them in variables, n being the amount of total lectures and ls being the presence of a coffee machine in the lecture hall, respective to its index. Two more variables are initialized to keep track of the amount of full coffee cups the student has (initialized to 0) and the amount of lectures she's awake in (the output), respectively. Following those two initializations is the beginning of a for loop which iterates over ls to read the presence of a coffee machine in each lectures hall. The dicision tree explained above is then implemented with if-statements. Finally, the output is printed to the terminal and the program terminates.

3.2 B: Estimating the Area of a Circle

Problem B is also quite straightforward, as it is essentially pure mathematics that are applied.

```
import math

def realArea(r):
    return math.pi * r ** 2

def approxArea(r, pm, pin):
    return (pin / pm) * ((r * 2) ** 2)

def compute(r, m, c):
    print("%f %f"%(realArea(float(r)), approxArea(float(r), int(m), int(c))))

while True:
    l = input()

    if l == "0 0 0":
        break

    s = l.split()
    compute(s[0], s[1], s[2])
```

Listing 2: "Estimating the Area of a Circle" Solution

3.3 C: Fun House

```
houseid = 1

def getCoordsOfChar(tiles, char):
    for i, ts in enumerate(tiles):
        if char in ts:
            return (i, ts.index(char))

def startVelocity(tiles, ci):
    if ci[0] == 0:
        return (1, 0)
    elif ci[0] == len(tiles) - 1:
        return (-1, 0)
    elif ci[1] == 0:
        return (0, 1)
    elif ci[1] == len(tiles[0]) - 1:
        return (0, -1)

def addTup(a, b):
    return(a[0] + b[0], a[1] + b[1])

def reverseTup(a):
    return (a[1], a[0])

def inverseTup(a):
    return (a[0] * -1, a[1] * -1)

def directionChange(tiles, pos, v):
    if tiles[pos[0]][pos[1]] == "\\":
        return reverseTup(v)

    if tiles[pos[0]][pos[1]] == "/":
        return inverseTup(reverseTup(v))

    else:
        return v

def printMap(tiles):
    for ts in tiles:
        print("".join(ts))

while True:
    l = input()

    if (l == "0 0"):
        break

    s = l.split()

    w = int(s[0])
    h = int(s[1])

    tiles = []

    for i in range(h):
        ts = input()
        tiles.append([t for t in ts])

    entry = getCoordsOfChar(tiles, "*")
    v = startVelocity(tiles, entry)
    pos = addTup(entry, v)

    while True:
        if tiles[pos[0]][pos[1]] == "x":
            tiles[pos[0]][pos[1]] = "&"
            break

        v = directionChange(tiles, pos, v)
        pos = addTup(pos, v)

    print("HOUSE %d"%(houseid))
    printMap(tiles)

    houseid += 1
```

Listing 3: "Fun House" Solution

3.4 D: Espresso!

```
def waterreq(order):
    if len(order) == 1:
        return int(order)
    else:
        return int(order[0]) + 1

l1 = input().split()
n = int(l1[0])
s = int(l1[1])

tank = s
count = 0

for i in range(n):
    l = input()

    if (tank - waterreq(l) < 0):
        tank = s - waterreq(l)
        count += 1
    else:
        tank -= waterreq(l)

print(count)
```

Listing 4: "Espresso!" Solution

3.5 E: Pea Soup and Pancakes!

```
n = int(input())
cond = False

for i in range(n):
    c = int(input())
    name = input()
    food = []

    for j in range(c):
        f = input()
        food.append(f)

    if ("pea soup" in food) and ("pancakes" in food):
        print(name)
        cond = True
        break

if cond == False:
    print("Anywhere is fine I guess")
```

Listing 5: "Pea Soup and Pancakes!" Solution

3.6 F: Math Trade

```
class Part:
    def __init__(self, name, has, wants):
        self.name = name
        self.has = has
        self.wants = wants

people = []

n = int(input())

for i in range(n):
    s = input().split()

    x = Part(s[0], s[1], s[2])

    people.append(x)

maxc = 0

for p in people:
    c = 0
    target = p
```

```
    targets = []
    while True:
        inter = list(filter(lambda x: x.has == target.wants and x != target, people))

        if len(inter) == 0:
            break
        else:
            if inter[0] in targets:
                if c > maxc:
                    maxc = c
                break

            targets.append(inter[0])

            c += 1
            target = inter[0]

if maxc > 0:
    print(maxc)
else:
    print("No trades possible")
```

Listing 6: "Math Trade" Solution

3.7 G: Love Polygon

Missing content

3.8 H: Santa Klas

```
import math

s = input().split()

h = int(s[0])
a = int(s[1])

if a >= 0 and a <= 180:
    print("safe")
else:
    angle = a

    if a > 180 and a < 270:
        angle = 270 + (270 - angle)

    angle = angle - 270

    print(math.floor(h / (math.cos(math.radians(angle)))))
```

Listing 7: "Santa Klas" Solution

3.9 I: Triangle Ornaments

Missing content

3.10 J: Sacred Texts

```
from sys import stdin

def calcRuneValue(rune):
    res = 0
    for r in rune:
        res += ord(r) - 32

    return res

lines = stdin.readlines()

l1 = input().split()
rune = l1[0]
trans = l1[1]

letters = [chr(a) for a in range(ord("a"), ord("z") + 1)]

offset = letters.index(trans) - calcRuneValue(rune)
```

```
19
20 for i in range(8):
21     l = input()
22
23     if l == "":
24         break
25
26     res = ""
27
28     runes = l.split()
29
30     for r in runes:
31         if r == "0":
32             res += " "
33             continue
34         elif r == "<":
35             res += ","
36             continue
37         elif r == ">":
38             res += "."
39             continue
40
41         res += letters[(calcRuneValue(r) + offset) % len(letters)]
42
43     print(res)
```

Listing 8: "Sacred Texts" Solution

3.11 K: Spelling Bee

```
1  lets = input()
2  letters = [l for l in lets]
3  words = []
4
5  n = int(input())
6
7  for i in range(n):
8      word = input()
9      words.append(word)
10
11 for word in words:
12     if len(word) >= 4:
13         cond = True
14         for l in word:
15             if not l in letters:
16                 cond = False
17                 break
18         if not letters[0] in word:
19             cond = False
20
21         if cond:
22             print(word)
```

Listing 9: "Spelling Bee" Solution

3.12 L: Fifty Shades of Pink

```
1  n = int(input())
2
3  count = 0
4
5  for i in range(n):
6      col = input().lower()
7
8      if "pink" in col or "rose" in col:
9          count += 1
10
11 if count == 0:
12     print("I must watch Star Wars with my daughter")
13 else:
14     print(count)
```

Listing 10: "Fifty Shades of Pink" Solution

References

[1] Kattis *Ranklist, scores and difficulties*, https://open.kattis.com/help/ranklist.

[2] Kattis *Terms of Service*, https://open.kattis.com/help/tos