November 2021
Westport, CT

William McGonagle
Nina Yuen
Greens Farms Academy

Fairfield Programming Association

Rules and Regulations 2022

Edition 1

Introduction

To Whom It May Concern,

Have you ever thought about creating a coding competition between schools? You may not have, but if you now are thinking this, I welcome you to continue reading. I am attempting to build a league of schools to host competitions testing merit in programming. It will be known as the Fairfield Programming Association (FPA). To create this organization, I only need a few things: support from your institution, two student delegates from your institution, and your help maintaining the cohesiveness of the FPA during its adolescent years. These first two years are likely going to be the most challenging for this new organization, and the FPA will be tested at its very foundation. That is why I need your help, and I have outlined what you can do below.

Firstly, your institution needs to support the FPA. This is not in any financial sense, but rather in a mutually beneficial partnership. The FPA's competitions would allow your students to create connections with students in other schools and it would give them the opportunity to thrive in a competitive environment- all while learning. All your institution would need to support the FPA is the commitment to hosting at least one competition each year. The logistics of these competitions are outlined in the below document.

You will need to elect two student delegates from your institution to help host the competitions. These delegates will be responsible for meeting in a committee with the other FPA delegates before each competition to design them so they are fair, challenging, and most importantly, fun. The logistics of the delegates and committee meetings will be outline in the below document.

Finally, your job will be to listen for any communications that the FPA sends out, and corral your students into teams for each competition. It is expected that every institution attends at least two other competitions, but that information, again, is outlined in the below document.

Best Regards,
William McGonagle

Table of Contents

Team Rules

Team Names

Teams can be referred to either as their school name or their school mascot. If the committee decides through a majority vote that a team mascot is not appropriate or is offensive, the team will only be referred to by their school name.

Team Coaches

Teams are required to have at least one coach. This individual must be over the age of 18 and employed by the institution that the team represents. The coach has legal responsibility of the students of their institution while attending the competition.

Coach Attendance

The coach, or a replacement coach, needs to attend every competition that the team attends. If a team does not have a coach present at the competition, the team is disqualified from that competition.

Replacement Coaches

Replacement coaches are allowed but greatly discouraged. A replacement coach must meet the same criteria as a regular coach.

Delegate Rules

Delegates are meant to represent schools, but they are not meant to represent the schools interests. Rather, delegates represent the Fairfield Programming Association and their actions should be in the best interest of the association.

Number of Delegates

Each school or organization is allowed to send two delegates at most. Alternatively, schools are allowed to send one delegate, and this delegate will have the power of two votes. If a school does not send any delegates, they will not have the ability to vote during the meeting.

Meeting Times

Meetings will be organized based on who will be able to attend. If there is at least a ⅔ majority of delegates that are able to attend, a meeting will be held. If less than ⅔ of the delegates are able to attend, meeting and voting will not be allowed.

Chief Delegate

At the start of every meeting, there will be a chief delegate that needs to be elected. For the initial meeting of every year, this will be the first delegate of Greens Farms Academy. This is done to prevent chaotic voting because the relations of the delegates may be uncertain at the beginning of the year.

Chief Delegate Responsibilities

The job of the chief delegate is to keep the meetings on track and stop the delegates from promoting inappropriate behavior. To do this, the chief delegate will track how long each of the delegates have spoken for and will also call votes whenever there needs to be one.

Chief Delegate Voting

The chief delegate is unable to vote in any of the decisions and they are not allowed to voice their opinions. They are there to act as a mitigator for chaos and a leader to keep the meetings on track.

Speaking Times

At the start of a meeting, delegates will be able to write their name on a docket. This docket will list the order of speaking for each of the delegates. Delegates are only allowed to speak for two minutes each, but are allowed to write their names back on the docket once their speaking has finished. The docket acts as a queue, so each new addition to the docket will occur at the bottom and the first speakers will be on the top.

Contest Rules

Contestants can use the python documentation but cannot use question and answer sites like stack overflow.

https://www.python.org/doc/

Problem Rules

Style Guide

Problem Language

must be written in python

should be accessible to someone who has done AP CS first semester

problems should have input/output

Easier problems are worth less points, hard problems are worth more

Problem solutions should be no more than 80 lines long

allow math library?

Delegates solutions should be nicely formatted and follow python style guidelines.
https://www.python.org/dev/peps/pep-0008/

The solution should have comments explaining what different aspects of the code do.

Solutions should only contain well known functions.
(Make a list of allowed functions)
https://docs.python.org/3/library/functions.html

Problems should have static input and output.
Problems cannot contain random generation.

Each problem should have a solution. you should include a sample solution in your problem submission

Each problem needs an intro, input and output formatting guide, short sample test suite, maybe history or a story if you would like.

Example Problems

Example 1 - Music Mixup

Intro

For synthesizers to work, they need to know what notes have certain values. Throughout the years, people have developed a string-notation system for these notes, and it is your job to calculate the note's integer value when given the note using the string-notation. To find the integer value of a note, you first need to understand how notes are stored.

```
C     =  0
C#/D- =  1
D     =  2
D#/E- =  3
E/F-  =  4
E#/F  =  5
F#/G  =  7
G#/A- =  8
A     =  9
A#/B- = 10
B     = 11
B#    = 12
```

Notes are stored using letters, accidental symbols, and numbers denoting the octave. The first character is always the letter, denoting the core value of the note. This letter can be anything A-G, but will always be uppercase. After the letter will be the accidental, which can be either a '#' or a '-'. This accidental is optional, and so not all string-notation notes will contain one. The final character of a string-notation note will always be the octave. An octave character can be any number ranging from 0-9.

In the case of this question, you do not have to think about the octave, only the note value inside of the octave. Your program should return the following values from above, but should do this in a programmatic way- no maps.

Input and Output Testing

The program should be able to take in the following values and output the following outputs.

Input	=	Output
A#3	=	10
A-3	=	8
B3	=	11
B#3	=	12
C2	=	0
D3	=	2
E5	=	4
F7	=	5
G2	=	7

Sample Solution

```
def GetNote(input):

    # Make Sure Note Exists
    if (len(input) < 2):
      return -1

    if (len(input) > 3):
      return -1

    # Dissect the String to Char
    noteChar = ord(input[0])

    # Find the Note Value. { A: 9, B: 11, C: 0, D: 2, E: 4, F: 5, G: 7}
    finalNotes = [ 9, 11, 0, 2, 4, 5, 7 ]
    finalNoteValue = finalNotes[noteChar - 65]

    # Check if note has accidental
    if (len(input) == 3):

        # Dissect the String
        accidentalChar = input[1]

        # If it has an accidental, apply accidental
        if (accidentalChar == '#'):
          finalNoteValue += 1

        if (accidentalChar == '-'):
          finalNoteValue -= 1

    # Return the Final Value
    return finalNoteValue
```

Example 2

Intro

Here's a question to see how well people know how to use loops and conditionals, "Can you write a program that prints all the numbers from 1 to 100?". But there is a catch, every multiple of three, print "Fizz", instead of the number. And for every multiple of five, print out "Buzz" instead of the number. Finally, if the number is a multiple of both three and five, print out "FizzBuzz". Each number should be separated by a newline, and each word should also be separated by a newline. An example of this can be shown in the figure below.

Input and Output Testing

Since there is no special input for this question, the only output that should be verified is below.

```
1
2
Fizz
4
Buzz
Fizz
7
8
Fizz
Buzz
11
Fizz
13
14
FizzBuzz
16
17
Fizz
19
Buzz
Fizz
22
23
Fizz
Buzz
26
Fizz
28
29
FizzBuzz
31
32
Fizz
34
Buzz
Fizz
37
38
Fizz
Buzz
41
Fizz
43
44
FizzBuzz
46
47
Fizz
49
Buzz
Fizz
52
53
Fizz
Buzz
56
Fizz
58
59
FizzBuzz
61
62
Fizz
64
Buzz
Fizz
67
68
Fizz
Buzz
71
Fizz
73
74
FizzBuzz
76
77
Fizz
79
Buzz
Fizz
82
83
Fizz
Buzz
86
Fizz
88
89
FizzBuzz
91
92
Fizz
94
Buzz
Fizz
97
98
Fizz
Buzz
```

Sample Solution

```
for fizzbuzz in range(100):

    if fizzbuzz % 3 == 0 and fizzbuzz % 5 == 0:
        print("fizzbuzz")
        continue

    elif fizzbuzz % 3 == 0:
        print("fizz")
        continue

    elif fizzbuzz % 5 == 0:
        print("buzz")
        continue

    print(fizzbuzz)
```