Técnicas de Programação Avançada

Trabalho 1 – Aclimatação com Juízes Online de Programação

Matheus da Silva Garcias

1 de Agosto de 2020

Conteúdo

1 Introdução **1**
1.1 Introdução ao documento 1
1.2 O que é a plataforma Kattis 1
1.3 Quem corrige os códigos enviados 1

2 Problemas **1**
2.1 Os problemas escolhidos 1
2.2 Soluções 2
2.2.1 A New Alphabet 2
2.2.2 Broken Swords 3
2.2.3 Encoded Message 4
2.2.4 Nasty Hacks 5
2.2.5 Odd Gnome 6
2.2.6 Speed Limit 6
2.2.7 Symmetric Order 7
2.2.8 Vacuumba 8
2.2.9 Trik 9
2.2.10 Apaxiaaaaaaaaaaans! 10

1 Introdução

1.1 Introdução ao documento

Este documento é rum relatório de problemas resolvidos na plataforma Kattis. Através desse documento será apresentada uma breve introdução, logo após serão apresentados os códigos e comentários sobre suas soluções.

1.2 O que é a plataforma Kattis

A plataforma Kattis é um website que permite que estudantes enviem soluções de problemas de computação variados, em nível de dificuldade e assunto, propostos pelo mesmo.

1.3 Quem corrige os códigos enviados

A verificação se dá a partir da entrada e saída de dados do programa. Cada problema define um modelo de entrada de dados e um modelo de saída. O programador deve interpretar a saída e entrada de dados de acordo com os formatos definidos pelo problema.

Após o envio do programa para o juiz virtual, a é executa uma verificação baseado no saída de dados do seu programa com os resultados reais. Caso todos as saídas do seu programam sejam iguais ao resultados dos juiz, seu programa é aprovado.

2 Problemas

2.1 Os problemas escolhidos

Os problemas escolhidos para resolução e aprensentação desse trabalho foram:

- A New Alphabet

  Resultado Kattis

- Broken Swords

  Resultado Kattis

- Encoded Message

  Resultado Kattis

- Nasty Hacks

  Resultado Kattis

- Odd Gnome

  Resultado Kattis

- Speed Limit

  Resultado Kattis

- Symmetric Order

  Resultado Kattis

- Vacuumba

  Resultado Kattis

- Trik

  Resultado Kattis

- Apaxiaaaaaaaaaaaans!

  Resultado Kattis

2.2 Soluções

2.2.1 A New Alphabet

A resolução deste problema é simples, basta trocar cada caractere da string fornecida pelos caracteres equivalentes fornecidos pela tabela do problema.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/anewalphabet
Resultado: https://open.kattis.com/submissions/5468063
"""

from sys import stdin, stdout

input = stdin.readline
print = stdout.write

symbols = ['@', '8', '(', '|)', '3', '#', '6', '[-]', '|',
 ↪  '_|', '|<', '1', r'[]\/[]', '[]\[]', '0', '|D', '(,)',
 ↪  '|Z', '$', '\'][\'', '|_|', r'\/', r'\/\/', '}{', '`/',
 ↪  '2']

t = input().lower()

for l in t:
    if ord(l) >= 97 and ord(l) <= 122:
```

```
            print(symbols[ord(l) - 97])
        else:
            print(l)

    print('\n')
```

Basicamente todos os caracteres lidos são transformados para minúscula, casa haja algum maiúsculo. Uma tabela para conversão dos caracteres é pré-criada para uso posterior no código.

Na parte do código a seguir, como apenas caracteres alfabéticos serão convertidos, é vericado se são caracteres a partir de seus números na tabela ASCII, já que são sequenciais. Caso seja alfabético, é subtraído 97 de seu valor ASCII, porque 97 é o valor de "a", a primeira letra do alfabeto. Assim podemos usar essa subtração para obter o indíce da lista de símbolos.

```
            for l in t:
                if ord(l) >= 97 and ord(l) <= 122:
                    print(symbols[ord(l) - 97])
                else:
                    print(l)
```

Caso o caractere não esteja dentro dos valores da tabela ASCII de caracteres alfabéticos minúsculos, então ele é enviado para a saída padrão sem nenhum processamento.

2.2.2 Broken Swords

Esse problema foi relativamente simples. Foi utilizado algumas operações bitwise, ou seja operações em níveis de bits.

Como a parte de cima da espada e a parte de baixo são as mesmas, então utiliza-se uma variável para guardar ambos.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/brokenswords
Resultado: https://open.kattis.com/submissions/5467655
"""

from sys import stdin, stdout

input = stdin.readline
print = stdout.write
```

```
tb, lr = (0, 0)
q = 0

n = int(input())

for _ in range(n):
    s = int(input(), 2)
    tb += (((s >> 2) & 1) ^ 1) + ((s >> 3) ^ 1)
    lr += (((s & 0b0010) >> 1) ^ 1) + ((s & 1) ^ 1)

while(tb >= 2 and lr >= 2):
    q += 1
    tb -= 2
    lr -= 2

print("%d %d %d\n" % (q, tb, lr))
```

Nesta linha de código abaixo, na parte antes do "+", é executado uma operação de "SHIFT"de dois 2 bits, e então uma operação de "AND"com 1 para descartar o último bit, e, por último, uma operação de "XOR"para negar o bit, visto que 0 quer dizer que esse pedaço da espada não está quebrada. A parte depois do "+"é muito parecida com a anterior, porém não é feito a operação de "AND"com 1, pois é o último bit (são 4 bits), e todos os bits depois dele são 0.

```
tb += (((s >> 2) & 1) ^ 1) + ((s >> 3) ^ 1)
```

A linha de código abaixo é basicamente a mesma coisa da anterior, porém ajustando alguns bits para conseguir extrair os bits corretos, os dois últimos bits.

```
lr += (((s & 0b0010) >> 1) ^ 1) + ((s & 1) ^ 1)
```

2.2.3 Encoded Message

Este é um problema simples, quando a matriz é girada, a primeira linha se torna a última coluna, a segunda linha a penúltima coluna e o resto segue a mesma lógica. Resolvi esse problema na mesma linha ao invés de criar uma matriz com listas.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
```

```
Problema: https://open.kattis.com/problems/encodedmessage
Resultado: https://open.kattis.com/submissions/5464102
"""

from sys import stdin, stdout
from math import sqrt

input = stdin.readline
print = stdout.write

n = int(input())

for _ in range(n):
    m = input()
    s = int(sqrt(len(m)))
    for j in range(s, 0, -1):
        for i in range(s):
            print(m[i*s+j-1])
    print("\n")
```

2.2.4 Nasty Hacks

Provavelmente este foi o problema mais simples de todos. Basta escrever a lógica descrita no problema no formato da lógica da linguagem de programação.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/nastyhacks
Resultado: https://open.kattis.com/submissions/5463684
"""

from sys import stdin, stdout

input = stdin.readline
print = stdout.write

n = int(input())

for _ in range(n):
    r, e, c = [int(x) for x in input().strip().split(' ')]
```

```
    print("does not matter\n" if r + c == e else "do not
    ↪  advertise\n" if r + c > e else "advertise\n")
```

2.2.5 Odd Gnome

Este problema também é um dos simples, como descrito no problema, cada gnomo, que não seja o rei, está em ordem crescente em relação ao seu ID, logo bastar iterar sob os gnomos e subtrair o gnomo atual com o próximo gnomo. Caso o número seja diferente de 1, então quer dizer aquele é o gnomo rei.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/oddgnome
Resultado: https://open.kattis.com/submissions/5464256
"""

from sys import stdin, stdout
from math import sqrt

input = stdin.readline
print = stdout.write

n = int(input())

for _ in range(n):
    g = [int(x) for x in input().strip().split(' ')][1:]
    for i in range(1, len(g)):
        if g[i] - g[i-1] != 1:
            print(str(i+1)+'\n')
            break
```

2.2.6 Speed Limit

Este problema basta somar as milhas percorridas com a multiplicação das milhas por hora pela subtração do tempo atual com o tempo anterior. Caso não seja feita a subtração, o programa contará os tempos anteriores percorridos, que leverá a um resultado errôneo.

Segue o código completo

```
#!/usr/bin/env python3
"""
```

```
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/speedlimit
Resultado: https://open.kattis.com/submissions/5463805
"""

from sys import stdin, stdout

input = stdin.readline
print = stdout.write

n = int(input())

while(n > -1):
    miles = 0
    last_t = 0
    for _ in range(n):
        s, t = [int(x) for x in input().strip().split(' ')]
        tmp = t
        t -= last_t
        miles += t * s
        last_t = tmp
    print("%d miles\n" % miles)

    n = int(input())
```

2.2.7 Symmetric Order

Neste problema, como a entrada de dados já está ordenada por tamanho da string, ela começa pelas palavras pequenas. Então basta ter uma lista para salvar a parte de cima e a parte de baixo. Enquanto se lê as linhas, basta criar uma virável que servirá de chaveadora para dizer se entrará no topo ou embaixo.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/symmetricorder
Resultado: https://open.kattis.com/submissions/5464202
"""

from sys import stdin, stdout
from math import sqrt
```

```
input = stdin.readline
print = stdout.write

n = int(input())
j = 1

while(n != 0):
    print("SET %d\n" % j)
    j += 1
    top = []
    bottom = []
    switch = True
    for _ in range(n):
        if switch:
            top.append(input())
        else:
            bottom.append(input())
        switch = not switch
    for string in top:
        print(string)
    for i in range(len(bottom)-1, -1, -1):
        print(bottom[i])

    n = int(input())
```

2.2.8 Vacuumba

Neste problema já temos a posição inicial do robô, basta ir adicionando a sua posição os valores de entrada. Como é ângulo + quantidade andado, temos que fazer contas com seno e consseno.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/vacuumba
Resultado: https://open.kattis.com/submissions/5467802
"""

from sys import stdin, stdout
from math import cos, sin, radians
```

```
input = stdin.readline
print = stdout.write

n = int(input())

for _ in range(n):
    m = int(input())
    x, y, a = (float(0), float(0), float(90))
    for _ in range(m):
        na, d = [float(x) for x in input().strip().split('
        ↪  ')]
        a += na
        x += cos(radians(a)) * d
        y += sin(radians(a)) * d
    print('%.6f %.6f\n' % (x, y))
```

2.2.9 Trik

Este problema é bem simples, os recipientes são representados pela posição da lista, a cada dado lido, apenas troca-se de posição os itens de acordo com a especificação do problema.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/trik
Resultado: https://open.kattis.com/submissions/5892564
"""

from sys import stdin, stdout

input = stdin.readline
print = stdout.write

state = [1, 0, 0]

line = input()

for l in line:
    if l == 'A':
        state[0], state[1] = state[1], state[0]
    elif l == 'B':
```

```
            state[1], state[2] = state[2], state[1]
        elif l == 'C':
            state[0], state[2] = state[2], state[0]

for i in range(3):
    if state[i]:
        print(str(i+1)+'\n')
        break
```

2.2.10 Apaxiaaaaaaaaaaans!

Para este problema basta salvar qual foi a última letra imprimida, e nas letras seguintes a serem possivelmente imprimidas, verificar se é igual a letra salva, caso seja, não imprimir o valor, caso não seja, imprimir o valor.

Segue o código completo

```
#!/usr/bin/env python3
"""
Autor: Matheus da Silva Garcias
Matrícula: 20171BSI0456
Problema: https://open.kattis.com/problems/apaxiaaans
Resultado: https://open.kattis.com/submissions/5892569
"""

from sys import stdin, stdout

input = stdin.readline
print = stdout.write

n = input()

if len(n) == 1:
    print(n + '\n')
else:
    last_letter = ''
    for l in n:
        if(l != last_letter):
            print(l)
            last_letter = l
```