

# Aula 01



Rodrigo G. Araújo

24/07/2019

# Programação Competitiva

Resolver problemas de Ciência da Computação conhecidos, o mais rápido possível.

Steven & Felix Halim (2010)

# Programação Competitiva

- Todos os problemas contem soluções existentes
- A velocidade caracteriza a competição
- Formar profissionais capazes de produzir softwares de qualidade
- Trabalho em equipe

# Programação Competitiva

Competitive programming combines two topics: (1) the design of algorithms and (2) the implementation of algorithms

Antti Laaksonen (2018)

# Programação Competitiva

- Resolução de problemas e pensamento matemático
- Combinação de técnicas conhecidas
- Novas interpretações de técnicas
- Habilidade em programação

## ACM ICPC

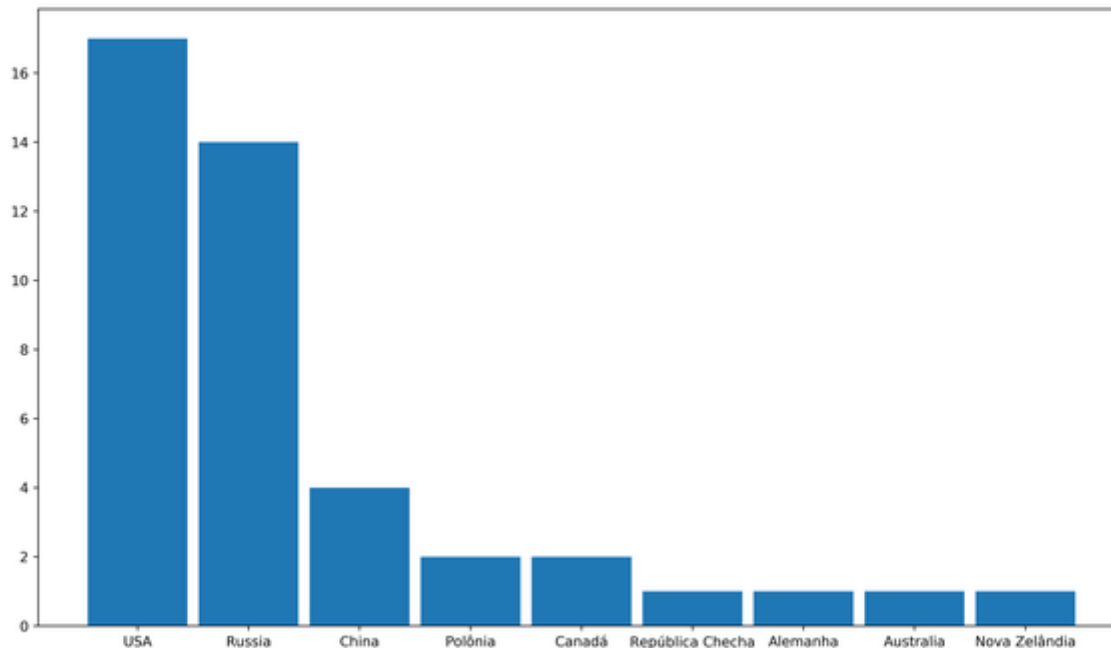
- Sub-Regional (14/09 - Centro Universitário IESB)
- Regional (07/11 - Campina Grande)
- Mundial (TDB)
- Equipes de três alunos, um reserva e um coach
- 8 a 14 problemas em 5 horas

# Regras da Maratona

## Critérios de Vitória

- Maior número de problemas
- Menor tempo total na submissão de soluções
- O tempo de submissão é o tempo transcorrido até o momento na competição
- Cada submissão incorreta antes de acertar gera uma penalidade de 20 minutos
- O tempo total é a soma dos tempos de submissão acrescido das penalidades
- Cada problema tem uma cor, e é entregue um balão ao acertar

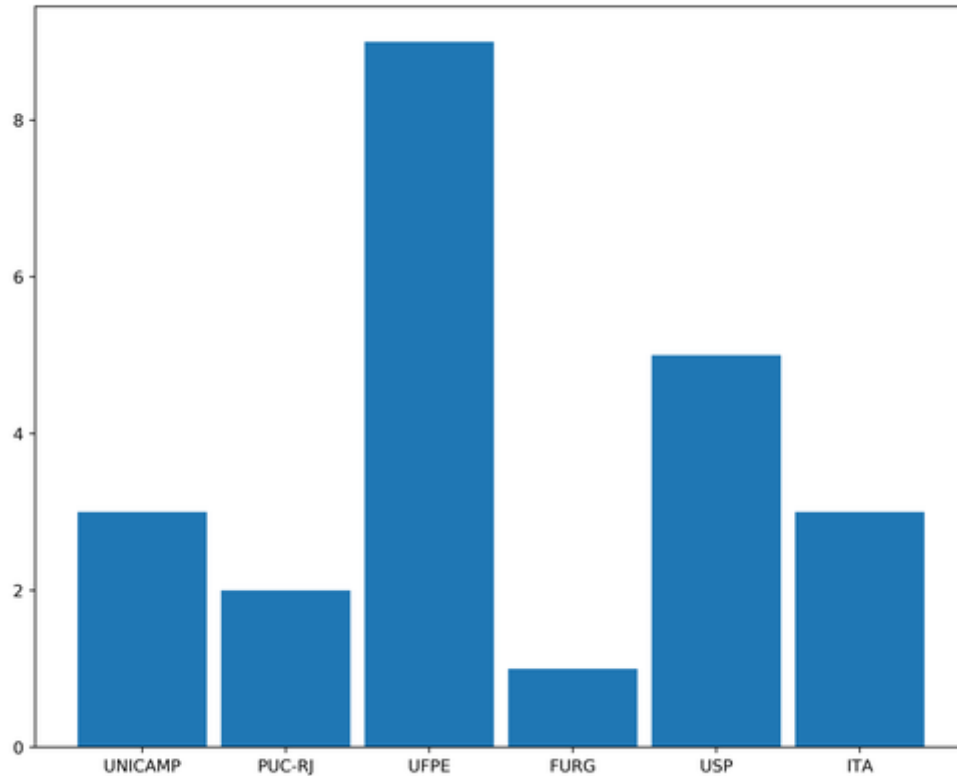
# Campeões Mundiais



Brasil conseguiu 13º lugar no mundial em 2005 com a USP



# Campeões Brasileiros



Brasília (UNB) conseguiu o 2º lugar em 1996

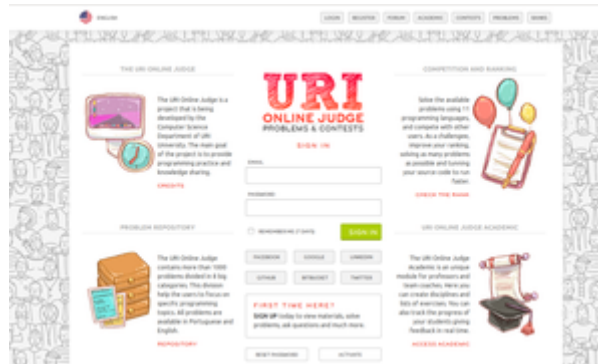
# IESB nas Maratonas

- 2016 - 11º Lugar - 3 Balões - Programação Orientada a Cebola
- 2017 - 7º Lugar - 5 Balões - Programação Orientada a Cebola
- 2018 - 9º Lugar - 3 Balões - DevLabs

UNB CIC e UNB FGA disputam a primeira posição todos os anos.

# Juizes Online

## URI - Iniciantes



## Codeforces - Intermediário/Avançado



# Juizes Online

Cada solução submetida ao juiz retornará um feedback sobre a solução

As linguagens aceitas na Maratona de Programação SBC são: C, C++, Java e Python

Código	Erro	Descrição	Código	Erro	Descrição
WA	<i>Wrong Answer</i>	Uma ou mais saídas geradas estão incorretas. O juiz não informa as entradas que geraram o erro nem a resposta correta para tais entradas	RE	<i>Runtime Error</i>	O programa trava durante a execução, geralmente por conta de falhas de segmentação, divisão por zero, etc
PE	<i>Presentation Error</i>	As saídas do programa estão corretas, mas a apresentação (formatação, espaçamento, etc) está diferente do que foi especificado	TLE	<i>Time Limit Exceeded</i>	Os programas devem gerar as saídas válidas dentro de um limite de tempo especificado. Caso o programa exceda este tempo, esta será a resposta do juiz
CE	<i>Compilation Error</i>	O programa não compila corretamente. Em geral, os juízes listam os parâmetros de compilação utilizados na correção	MLE	<i>Memory Limit Exceeded</i>	O programa requer mais memória em sua execução do que o juiz permite

Imagens obtidas de <https://github.com/edsomjr/TEP/>

# Entrada e Saída

A programação competitiva requer que os problemas sejam solucionados lendo arquivos de entrada específicos e escrevendo a saída em um arquivo da forma solicitada. Os arquivos em sua maioria são obtidos pela entrada e saída padrão.

Os arquivos de entrada são divididos em quatro tipos:

1. Um único caso do problema
2. N casos do problema, com o valor de N informado na primeira linha
3. N casos do problema, acabando com um valor informado
4. M casos do problema, até que acabe o arquivo (EOF)

# Entrada e Saída

Categoria 1: Um único caso do problema

Arquivo de Entrada

X Y

```
x, y = map(int, input().split())  
print(x+y)
```

# Entrada e Saída

Categoria 2: Varios casos informados no inicio.

Arquivo de Entrada

QTD  
X Y

```
qtd = int(input())  
for _ in range(qtd):  
    x, y = map(int, input().split())  
    print(x+y)
```

# Entrada e Saída

Categoria 2: Varios casos acabando com um valor.

Arquivo de Entrada

X Y

X Y

X Y

-1 -1

```
while True:
    x, y = map(int, input().split())
    if x == -1 and y == -1:
        break
    print(x+y)
```



# Entrada e Saída

Categoria 2: Varios casos até o fim de arquivo.

Arquivo de Entrada

X Y

X Y

X Y

X Y

```
while True:
    try:
        x, y = map(int, input().split())
        print(x+y)
    except EOFError:
        break
```

# Exercicios

UVA 100 - The  $3n + 1$  problem

# Exercicios

## UVA 100 - The $3n + 1$ problem

```
def circle(n):
    qtd = 0
    while True:
        qtd += 1
        if n == 1:
            break
        n = (3*n + 1) if n%2 != 0 else n//2
    return qtd

while True:
    try:
        x, y = map(int, input().split())
        ans, a, b = 0, min([x, y]), max([x, y])
        while a <= b:
            ans = max(ans, circle(a))
            a += 1
        print(x, y, ans)
    except EOFError:
        break
```