Aula 10:
Upsolving do Warmup 3

***Disciplina*: Maratona de Programação 1**

Profs. Edmilson Marmo e Luiz Olmes

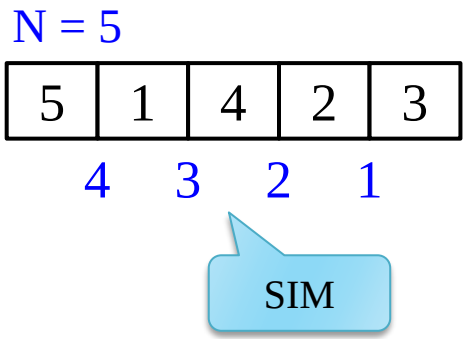
edmarmo@unifei.edu.br, olmes@unifei.edu.br

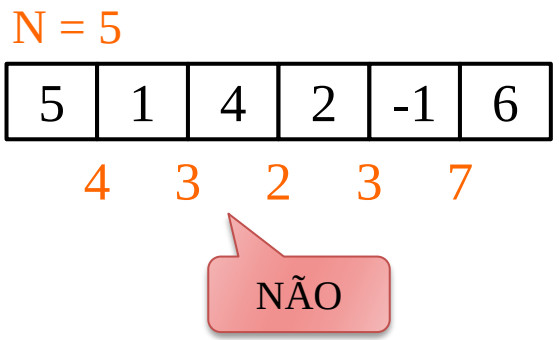
UNIFEI
UNIVERSIDADE FEDERAL DE ITAJUBÁ

Problema A: Jolly Jumpers

- Total de Submissões: 98
- Submissões Aceitas: 41 (42%)

- Dados N valores, basta verificar se o módulo das diferenças entre elementos sucessivos abrangem todos os valores entre 1 e (N – 1).





Problema B: Operadores Lógicos

- Total de Submissões: 94
- Submissões Aceitas: 24 (26%)

- Operadores lógicos:

A	B	NOT (A)	AND	OR	XOR	NAND	NOR	XNOR
0	0	1	0	0	0	1	1	1
0	1	1	0	1	1	1	0	0
1	0	0	0	1	1	1	0	0
1	1	0	1	1	0	0	0	1

Problema B: Operadores Lógicos

- NOT é um operador unário:
  - NOT 0011
- Os demais operadores são binários:
  - 1111 AND 0010
- Ler a entrada no formato de string facilita a manipulação dos operadores e operandos.
  - Usar string em C (vetor de char) torna-se mais simples do que empregar a classe `std::string` da linguagem C++.
- Pode-se usar condicionais para obter as respostas da tabela anterior.

Problema C: Não Acredito

- Total de Submissões: 84
- Submissões Aceitas: 44 (52%)

- Ao ler a entrada no formato de string, cada dígito pode ser convertido para a sua forma numérica com uma subtração do caractere `'0'`.

- Ao terminar de processar todos os dígitos, pode-se sobrescrever a string de entrada com o uso da função `sprintf`.

- Repete-se o processo, até que a string tenha tamanho 1.

Problema D: Classes de Equivalência

- Total de Submissões: 11
- Submissões Aceitas: 0 (0%)

- Para resolver o problema da quantidade de classes de equivalência de pares ordenados, uma das formas é aplicar a abordagem de conjuntos disjuntos.
  - Também conhecido como Union-find (*próximas aulas*).

- A ideia é contar quantas raízes únicas ("*parent*" no algoritmo) existem, que correspondem às classes de equivalência.

Problema E: O Crivo de Eratóstenes

- Total de Submissões: 111
- Submissões Aceitas: 11 (10%)

- O problema se traduz em uma contagem de quantos números que não são primos estão dentro do intervalo [A, B].

- Para cada intervalo da entrada, códigos que computem um laço para realizar a contagem inevitavelmente estouram o tempo limite do problema.

- Duas abordagens:
  - Computar o crivo de Eratóstenes antes de realizar a leitura dos valores de entrada.
  - Implementar um vetor de prefixo, para permitir computar a soma.

Problema E: O Crivo de Eratóstenes

```
1. // Inicialmente, assumimos que todos os números são primos
2. memset(ehPrimo, true, sizeof(ehPrimo));
3.
4. ehPrimo[0] = ehPrimo[1] = false; // 0 e 1 não são primos
5.
6. for(int i = 2; i * i <= MAX; i++)
7. {
8.     if(ehPrimo[i])
9.     {
10.        for(int j = i * i; j <= MAX; j += i)
11.        {
12.            ehPrimo[j] = false; // Marca os múltiplos de i como não-primos
13.        }
14.    }
15.}
```

Problema F: Fugindo do Chefe

- Total de Submissões: 23
- Submissões Aceitas: 0 (0%)

- Trata-se de um problema de caminho mínimo em grafo.
  - O chefe faz o percurso de casa até o escritório.
  - O funcionário faz o percurso de casa até o supermercado.
  - Os dois não podem se encontrar em nenhum momento.

- Ao encontrar o caminho mínimo o funcionário não pode passar por nenhum vértice que faça parte deste caminho.
  - Pode haver mais de um caminho mínimo. Neste caso, o funcionário não pode passar por nenhum deles.

Aula 10:
Upsolving do Warmup 3

***Disciplina*: Maratona de Programação 1**

Profs. Edmilson Marmo e Luiz Olmes

edmarmo@unifei.edu.br, olmes@unifei.edu.br

UNIFEI
UNIVERSIDADE FEDERAL DE ITAJUBÁ