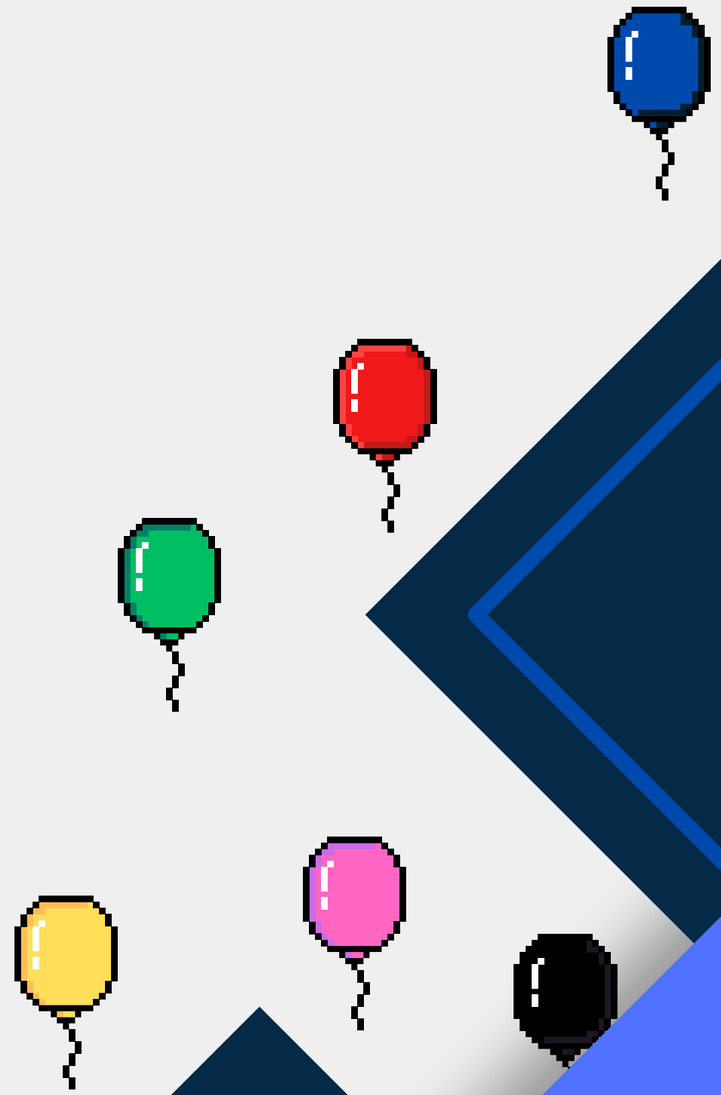
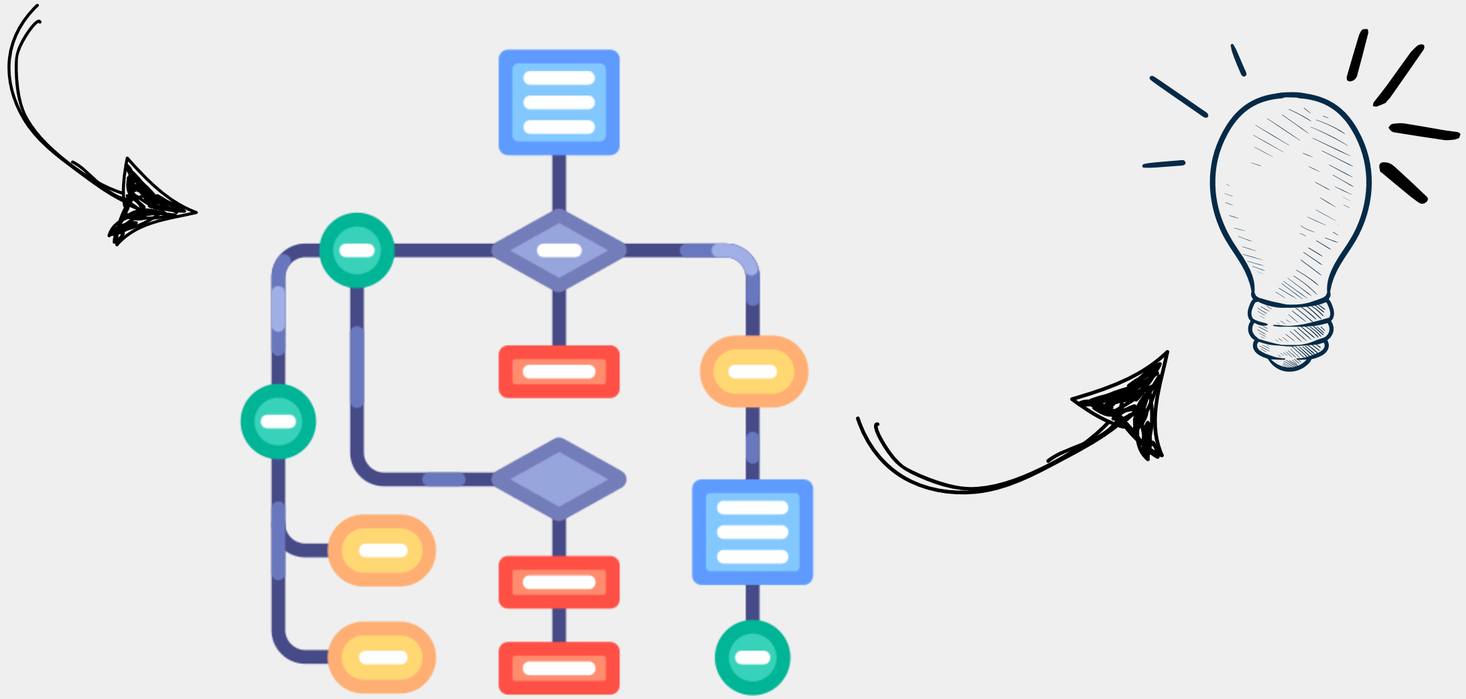
1º WORKSHOP

PROGRAMAÇÃO COMPETITIVA



O QUE É PROGRAMAÇÃO COMPETITIVA ?

- Resolver problemas desafiadores de forma eficiente
- Design de algoritmo • Implementação de algoritmo

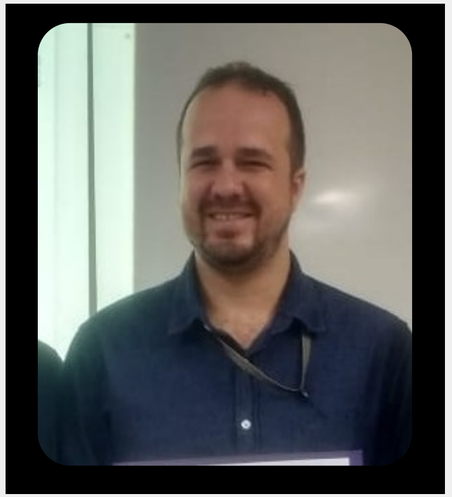


PONTO DE VISTAS DIFERENTES → **22/02/2022**

QUEM SOMOS?

Prof Lucas Farias

Prof Diego Pinheiro



Prof Jheymesson

QUEM SOMOS?

Victor Hugo

5º Período

Julia Vilela

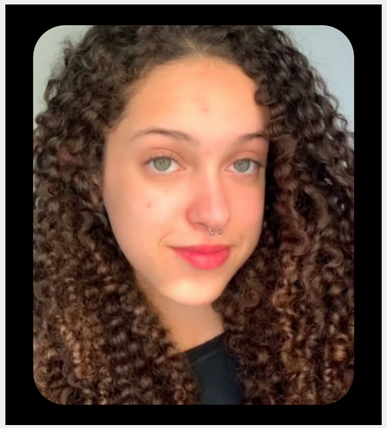
5º Período

Maria Luiza

5º Período

Isadora Xavier

7º Período



Isabela Medeiros

5º Período

Eduardo Braga

5º Período

Henrique Franca

5º Período

O QUE FAZEMOS ?

Nos reunimos duas vezes na semana

MARATONISTAS

Wrath	Vasya and Strings	Ogros	Free Set	Ascend Sequence	Obtain the String	Present
AC	AC	AC	AC	AC	AC	AC
AC	AC	AC	AC	AC	WA	TLE
AC	AC	AC	AC	AC	TLE	
AC	AC	AC	AC	AC		
AC	AC	AC	AC	AC	AC	WA

Resolvemos e comentamos questões de uma planilha disponibilizada pela USP/UFMG

MARATONISTAS

Penalty	A	B	C	D	E	F
	1 / 1	4 / 17	2 / 8	0 / 2	0 / 1	2 / 3
799	3:26:03	4:22:30 (-8)	0:42:38			2:07:50
232			1:54:29 (-1)	(-2)		1:17:57 (-1)
71		1:11:49				
84		1:24:57			(-1)	

AC → Accepted

TLE → Time Limited Exceded

WA → Wrong Answer

Realizamos contests entre nossas equipes, como treinamento e simulação para as competições

ALGUNS JUÍZES ONLINES QUE UTILIZAMOS

CODEFORCES

UVa Online Judge

NEPS ACADEMY

 Virtual Judge

Sphere online judge

 KATTIS

beecrowd

HackerRank

COMO SERIA UM PROBLEMA?

Existem *N* casas sendo vendidas e cada casa tem um preço *C* e Pedro tem um saldo *K* de dinheiro. Queremos saber qual o <u>máximo</u> de casas que Pedro consegue comprar com este saldo *K*:

(1 <= N <= 100)
(1 <= K <= 1000)
(1 <= Ci <= 10000)

EX: *N = 10* *K = 400*

	1	2	3	4	5	6	7	8	9	10
	250	100	350	90	500	80	97	350	180	400

EVENTOS QUE PARTICIPAMOS

A *Maratona de Programação* é um evento da SBC que nasceu das competições regionais classificatórias para as etapas mundiais da competição de programação, o *International Collegiate Programming Contest*, onde o objetivo é competir através da resolução do maior número de problemas em menos tempo.

- **Primeira Fase (Regional)**
- **Segunda Fase (Final Brasileira)**
- **Terceira Fase (Final Latino-Americana)**
- **Quarta Fase (Final Mundial)**

EVENTOS QUE PARTICIPAMOS

Fase Regional 2023

Final Brasileira 2023

Final Brasileira 2023

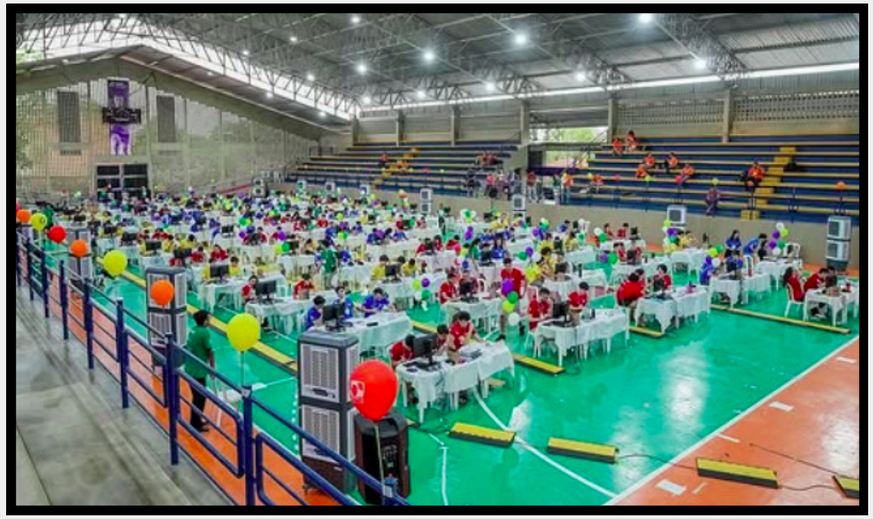
Fase Regional 2024

Fase Regional 2024

Fase Regional 2024

Final Nacional 2024

Final Nacional 2024



Brazilian ICPC Summer School

UNICAMP

O objetivo do evento é trabalhar habilidades de programação avançada, incluíndo tópicos como teoria dos grafos, geometria computacional, strings, programação dinâmica, árvores, entre outros

Summer School 2024

Summer School 2025

EVENTOS QUE ORGANIZAMOS

A Day in the Life

Momento em que o grupo traz questões para resolver durante o evento.

Maratona de Programação

Maratona para resolução de questões

Oficina de Programação

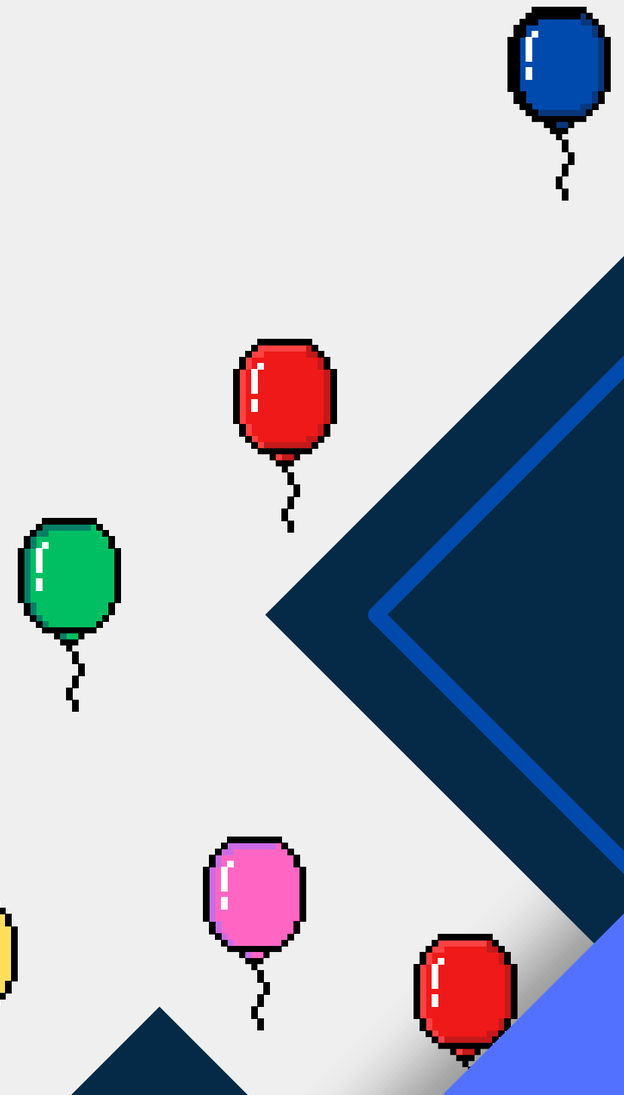
Discussão de problemas de programação competitiva e métodos de resolução

Workshop de Programação

Curso de Programação Competitiva de 2 semanas

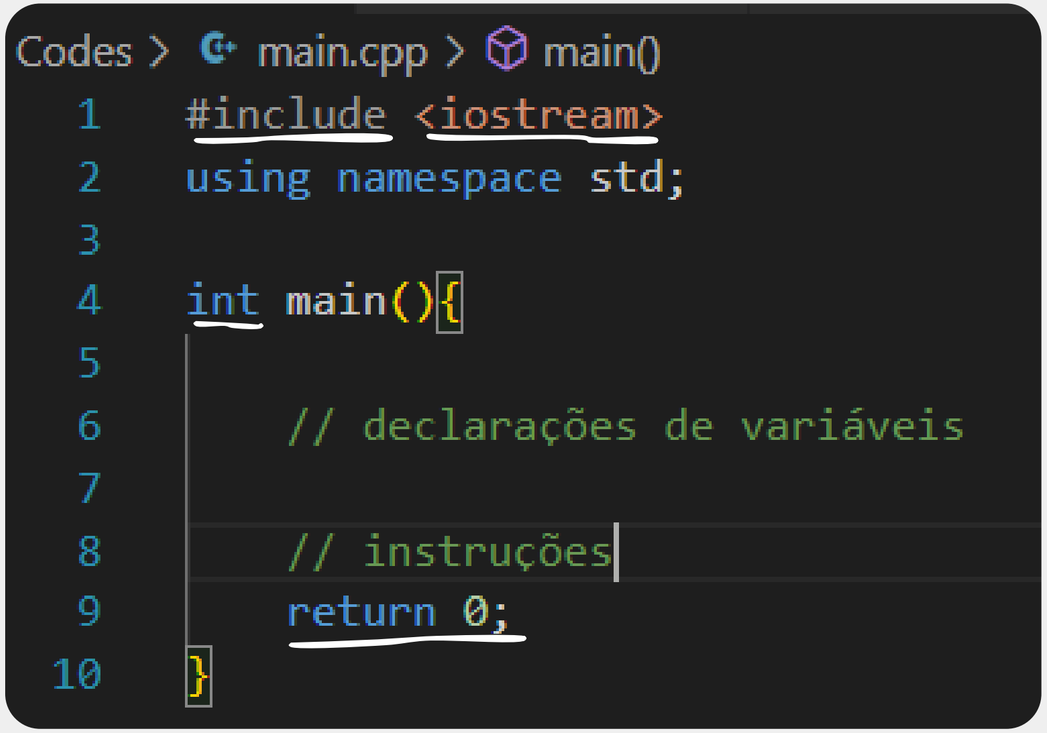

SBC
Sociedade Brasileira de Computação
Maratona de Programação
C3

LINGUAGEM



QUE LINGUAGEM VAMOS USAR?

ESTRUTURA BÁSICA DE UM CÓDIGO



#include <bits/stdc++.h>

ESTRUTURA BÁSICA DE UM CÓDIGO

des > main.py > ...

```python
import math

num = int(input())

raiz = math.sqrt(num)

print(f"A raiz de {num} é {raiz}")

```

Python

```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main(){

    int num;
    cin >> num;
    float raiz = sqrt(num);

    cout << "A raiz quadrada de " << num << " eh " << raiz << endl;

    return 0;
}
```

C++

COMO COMPILAR E RODAR UM CÓDIGO?

g++ -g nome-arq.cpp -o nome-exec

./nome-exec

TIPOS PRIMITIVOS

Type	Size (Bytes)	Range
char	1	-127 to 127
int	4	-2.147.483.648 to 2.147.483.647
unsigned int	4	0 to 4.294.967.295
short int	2	-32.768 to 32.767
unsigned short int	2	0 to 65.535
long long int	4	-9.223.372.036.854.775.808 to 9.223.372.036.854.775.807
unsigned long int	4	0 to 18.446.744.073.709.551.615
float	4	+/- 3.4e+/-38 (~7 digits)
double	8	+/-1.7e+/-308(~15 digits)
bool	1	True or False

COMO DECLARAR E LER UMA VARIÁVEL?

Type nome-variavel ;

cin >> nome-variavel

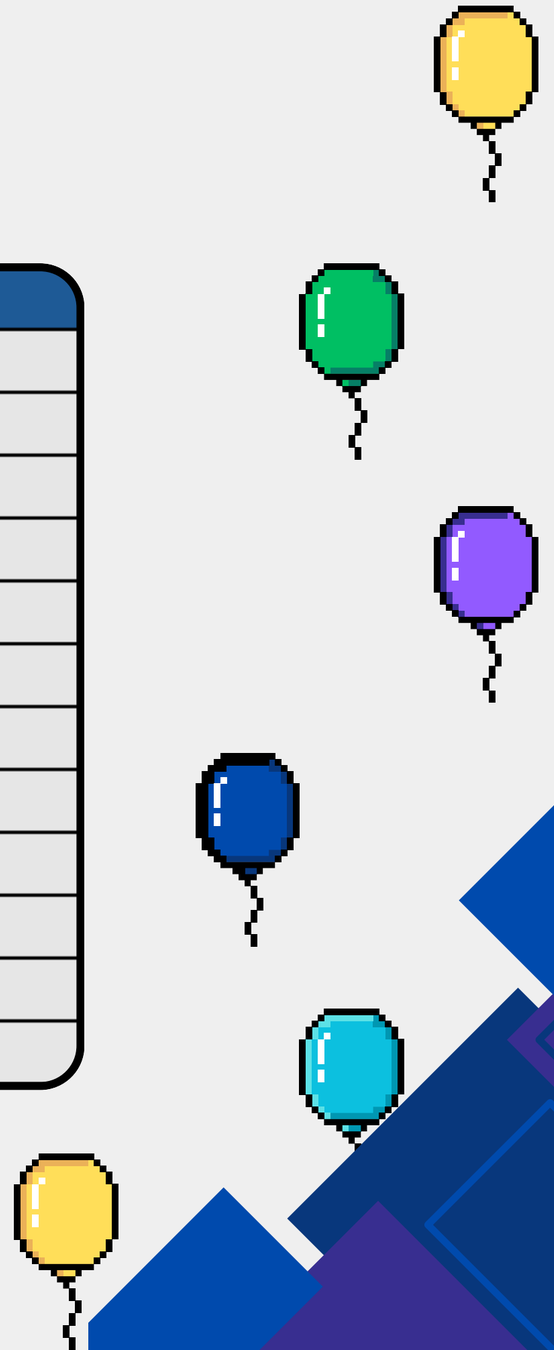
LÓGICA MATEMÁTICA E FUNDAMENTOS

Isabela Medeiros

OPERADORES ARITMÉTICOS

Operador	Descrição	Exemplo
+	Adição	a + b
-	Subtração	a - b
*	Multiplicação	a * b
/	Divisão	a / b
%	Módulo (resto da divisão)	a % b
++	Incremento	a++
--	Decremento	b--
+=	Atribuição por soma	a += b (a = a + b)
-=	Atribuição por subtração	a -= b (a = a - b)
*=	Atribuição por multiplicação	a *= b (a = a * b)
/=	Atribuição por divisão	a /= b (a = a / b)
%=	Atribuição por módulo	a %= b (a = a % b)

questão para praticar: *Produto Simples*



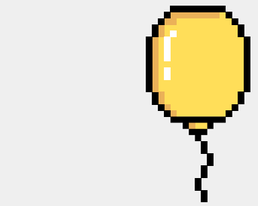
PRIORIDADE DAS OPERAÇÕES

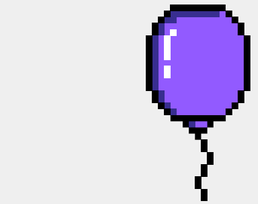
Prioridade	Operador
1	()
2	(*), (/), (%)
3	(+), (-)

CONDICIONAIS

O if - else if - else é usado quando há múltiplas condições a serem verificadas. O programa testa cada condição na ordem em que aparecem e executa o primeiro bloco verdadeiro. Se nenhuma condição for atendida, o else é executado.

```
if (condição1) {
    // Executa este bloco de comandos se condição1 for verdadeira
} else if (condição2) {
    // Executa este bloco de comandos se condição1 for falsa,
    // mas condição2 for verdadeira
} else {
    // Executa este bloco de comandos se todas as condições anteriores forem falsas
}
```






OPERADORES RELACIONAIS

Símbolo	Nome do operador	Exemplo	Significado
>	Maior que	x > y	x é maior que y?
>=	Maior ou igual	x >= y	x é maior ou igual a y?
<	Menor que	x < y	x é menor que y?
<=	Menor ou igual	x <= y	x é menor ou igual a y?
=	Igualdade	x == y	x é igual a y?
!=	Diferente de	x != y	x é diferente de y?

OPERADORES LÓGICOS

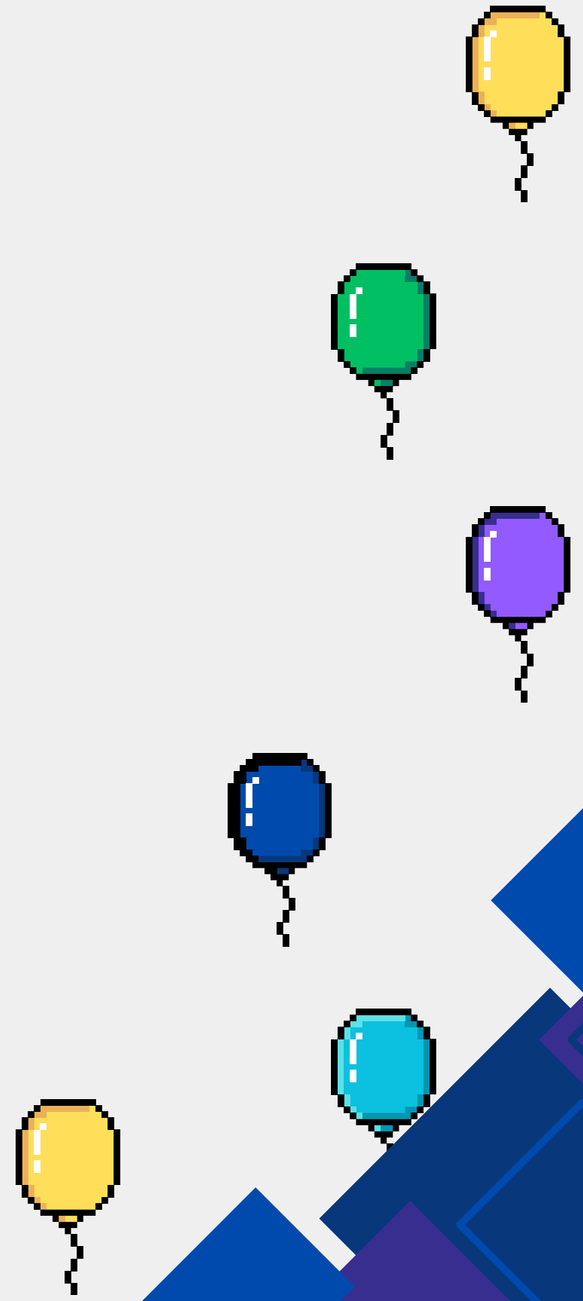
Operadores lógicos	Descrição	resultado
&&	E lógico	Retorna true se ambas as condições forem verdadeiras
||	OU lógico	Retorna true se pelo menos uma das condições for verdadeira
!	NÃO lógico	Inverte o valor lógico

EXEMPLOS BÁSICOS

-verificar se um número é par ou ímpar

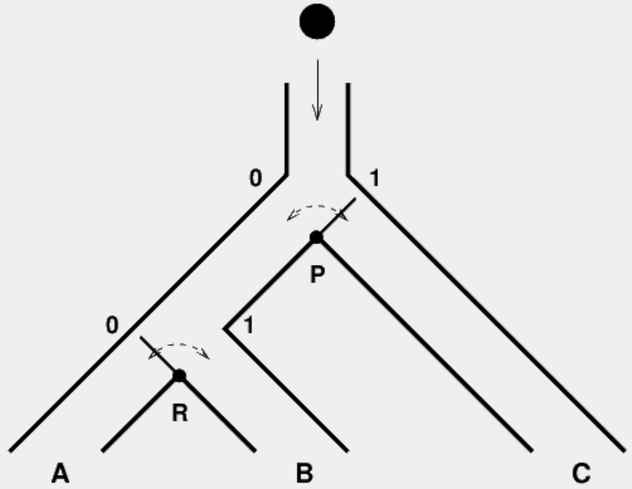
-verificar se um número é positivo ou negativo

-verificar qual número é maior



QUESTÃO PRÁTICA : FLÍPER

Flíper é um tipo de jogo onde uma bolinha de metal cai por um labirinto de caminhos até chegar na parte de baixo do labirinto. A quantidade de pontos que o jogador ganha depende do caminho que a bolinha seguir. O jogador pode controlar o percurso da bolinha mudando a posição de algumas portinhas do labirinto. Cada portinha pode estar na posição 0, que significa virada para a esquerda, ou na posição 1 que quer dizer virada para a direita. Considere o flíper da figura abaixo, que tem duas portinhas. A portinha PP está na posição 1 e a portinha RR, na posição 0. Desse jeito, a bolinha vai cair pelo caminho B.



Você deve escrever um programa que, dadas as posições das portinhas PP e RR, neste flíper da figura, diga por qual dos três caminhos, A, B ou C, a bolinha vai cair!

flíper

LEITURA DE ENTRADAS ATÉ EOF

Permite processar múltiplas entradas de forma contínua, é usado em problemas de programação competitiva onde a quantidade de entradas não é fixa.

LEITURA DE ENTRADAS COM QUANTIDADE FIXA

Útil quando o número de entradas é conhecido, pois o programa lê um número fixo de entradas, definido pelo usuário.

QUESTÕES PRÁTICAS

Back to High School Psysics

relembrando fórmula do movimento retilíneo uniforme:

$$S = v \times t$$

Desafio do Maior Número

Divisibility Problem