

O JOGO DO CIRCUITO

INTRODUÇÃO

O que é o jogo do circuito?

Bem-vindo ao guia do Jogo do Circuito! 😊 Este é um jogo divertido e educativo que ajudará você a entender o que é um algoritmo e como ele se diferencia de um programa.

Objetivo do jogo

O principal objetivo do Jogo do Circuito é apreender conceitos básicos de pensamento computacional de forma prática e divertida. Você poderá apreender:

1. **O que é um algoritmo:** Passo a passo para resolver um problema.
2. **Diferença entre algoritmo e programa:** Algoritmo é a receita, o programa é o bolo pronto!
3. **Como montar e seguir circuitos:** Colocando a mão na massa para entender na prática.

PENSAMENTO COMPUTACIONAL

Pensamento computacional é uma abordagem para resolver problemas que envolve dividir problemas complexos em partes menores e mais gerenciáveis, identificar padrões, abstrair detalhes irrelevantes e criar algoritmos para a solução dos problemas. Essa abordagem é essencial não só para a programação de computadores, mas também para resolver problemas em várias áreas do conhecimento.

Componentes principais do pensamento computacional:

1. **Decomposição:** Dividir um problema grande em partes menores e mais fáceis de resolver.
2. **Reconhecimento de Padrões:** Identificar semelhanças ou padrões dentro dos problemas.
3. **Abstração:** Focar nos detalhes importantes e ignorar os irrelevantes para simplificar o problema.
4. **Algoritmos:** Desenvolver um conjunto de instruções passo a passo para resolver cada parte do problema.

O pensamento computacional é habilidade técnica, que pode ser aplicada em diversas situações do dia a dia ajudando a ter raciocínio lógico e eficiente, facilitando a tomada de decisões e na resolução de problemas complexos.

Neste jogo você poderá explorar o componente de **Algoritmos** e o componente de **Reconhecimento de Padrões**.

O JOGO

Aqui está o mapa do jogo e o seu objetivo que é pegar o item no mapa sem cair na água ou subir nas árvores.



Note que no mapa estão descritos o objetivo principal “Pegar o item” e as restrições mencionadas e existe um espaço para colocar uma sequência de instruções. Na parte inferior você encontrará os comandos disponíveis, **escolha** um destes personagens e posicione-o em qualquer lugar do mapa:



O personagem escolhido deverá caminhar até o item a ser coletado e depois escolha um item e posicione-o no mapa:



Personagem e item posicionados é hora do desafio principal: construir uma sequência de instruções que leve seu personagem a coletar o item.

Desafio
Pegar o item.
Obs: Não sei nadar, nem subo em árvores.

Instruções

Mapa

Repita	Ande para →	Ande para ↑
Ande para ↓	Ande para ←	Pegar

Josépon Peixoto Filho
Pedro Henrique Coqueiro Braga

Neste exemplo há 3 possíveis caminhos de solução. No primeiro momento não importa qual é o melhor caminho, mas escolher um deles. Nós normalmente batemos o olho no mapa e intuímos o caminho construindo uma imagem mental dele, mas o nosso personagem não sabe o que tem na casa ao lado, então, precisamos orientá-lo, passo a passo, por isso as instruções.

Se a primeira instrução for **Ande para** nosso personagem terá um problema: cairá na água e não sabe nadar. Tirei a linha

1. Ande para
2. Ande para
3. Ande para
4. Ande para
5. Ande para
6. Ande para
7. Ande para
8. Ande para
9. Ande para
10. Ande para
11. Ande para
12. Pegar

Faça as instruções à parte do tabuleiro, uma para cada solução, assim o tabuleiro poderá ser usado muitas vezes;

Desta forma você executará as diversas soluções reaproveitando o mapa.

Lembre-se de memorizar as posições iniciais do personagem e do item!

Desafio
Pegar o item.
Obs: Não sei nadar,
nem subo em árvores.

Instruções

The map shows a character in a green field with a red apple. There are several black dots representing trees and blue squares representing water. A large orange arrow points from the text area to the list of commands on the left.

```

1 - Ande para ←
2 - Ande para ↑
3 - Ande para ↑
4 - Ande para ↑
5 - Ande para →
6 - Ande para →
7 - Ande para →
8 - Ande para →
9 - Ande para →
10 - Ande para →
11 - Ande para ↓
12 - Pegar
  
```

Repita	Ande para →	Ande para ↑
Ande para ↓	Ande para ←	Pegar

Jogos Pequenos Fofos
Pedro Henrique Coqueiro Braga

Teste cada solução: movimente o personagem, passo a passo;
Verifique se consegue resolver o caminho 2 e 3 seguindo a mesma lógica;

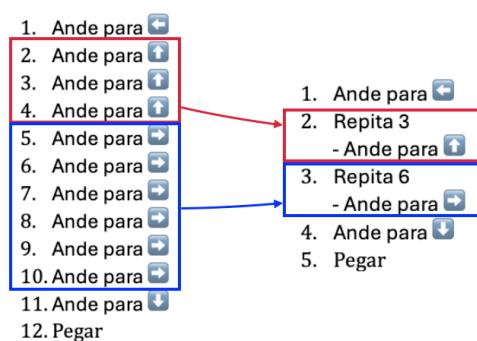
Observe que há muitos comandos repetidos. Para deixar as instruções mais simples use o comando **Repita**.

Para esse comando você precisa indicar quantas vezes um conjunto de instruções deve ser repetido.

Observe o uso do comando repita:

1. Ande para ←
2. Repita 3
 - Ande para ↑
3. Repita 6
 - Ande para →
4. Ande para ↓
5. Pegar

O último conjunto de instruções é equivalente ao anterior, portanto é importante saber que **ambas as soluções estão corretas**, a diferença é que essa segunda solução é mais **curta**.



Coverta sua solução do caminho **2** e do **3** usando o comando **Repita** quando considerar necessário.

Variações

Variação 1: É possível variar este jogo, permitindo colocar mais de um item no tabuleiro, exigindo que a bola seja pega antes da maçã, por exemplo.



Variação 2: Com dois personagens, cada um tem que pegar um item, os dois se movem uma casa por vez, ao mesmo tempo, mas não podem ocupar a mesma casa.



Outras variações: Use sua imaginação e crie variações, se quiser pode incluir mais comandos como por exemplo **Parar** que significa que o personagem não se move. Também pode retirar a restrição de não saber nadar e incluir o comando **Nadar**. A ideia é se divertir enquanto aprende que para se construir um algoritmo é preciso saber quais os possíveis comandos, quais as restrições e qual o contexto do problema (o mapa)

ALGORITMO VS PROGRAMA

Algoritmo e programa são conceitos fundamentais na computação, mas muitas vezes são confundidos.

Um **algoritmo** é uma **sequência** de passos bem **definidos e finitos** que **resolve** um problema ou realiza uma tarefa específica. Pense em um algoritmo como uma receita de bolo. A receita descreve cada etapa que você precisa seguir para fazer o bolo, desde reunir os ingredientes até o tempo de cozimento.

Características dos algoritmos:

1. **Seqüencial:** Executado em uma ordem específica;
2. **Clareza:** Não há ambiguidade em cada;
3. **Finitude:** Há um número finito de passos;
4. **Eficácia:** Resolver o problema – eficiência .

Exemplo de um algoritmo simples:

1. Adicione 2 xícaras de farinha.
2. Misture 1 xícara de açúcar.
3. Adicione 2 ovos.
4. Misture bem todos os ingredientes.
5. Coloque a mistura em uma forma e asse por 30 minutos.

Programa, por outro lado, é a implementação de um ou mais algoritmos em uma linguagem de programação que um computador pode executar. Enquanto o algoritmo é a receita, o programa é o bolo pronto, é a tradução dos passos descritos no algoritmo em um código que a máquina pode entender e executar.

Características dos programas:

1. **Codificado:** Escrito em uma linguagem de programação como Swift, Python, Java, C++, etc.
2. **Executável:** Pode ser rodado por um computador para realizar a tarefa especificada.
3. **Interativo:** Pode receber entradas e fornecer saídas.

Exemplo de um programa simples em Python que segue o algoritmo de somar dois números:

```
# Algoritmo: somar dois números
def somar(n1, n2):
    return n1 + n2

# Programa: implementação do algoritmo em Python
numero1 = 5
numero2 = 3
resultado = somar(numero1, numero2)
print("A soma é:", resultado)
```

Em resumo, enquanto um algoritmo é uma série de passos para resolver um problema, um programa é a implementação desse algoritmo em uma linguagem que um computador pode entender e executar.

No contexto do nosso jogo, a sequência de instruções escritas no papel é o nosso algoritmo, mas se quiséssemos que o computador resolvesse o problema teríamos que usar uma linguagem de programação e escrever esse algoritmo nessa linguagem.

Você já viu aquelas competições que robôs precisam sair de um labirinto e quem resolver primeiro ganha? Um exemplo pode ser visto no YouTube acessando o link: <https://youtu.be/ZMQbHMgK2rw?si=no4Muj0JAb9iAGR9>

Bem, é parecido com nosso problema, não é? O robô é o nosso personagem e a saída o nosso item. A diferença é que aqui aprendemos como resolver problemas específicos, mas poderíamos pensar em um algoritmo que resolve qualquer situação do nosso jogo, mas para isso precisaríamos de um conjunto mais de comandos, além daqueles já apresentados.

FONTE

Este tipo de atividade é inspirado em atividade propostas por outros pesquisadores, por isso relaciono aqui algumas fontes que descrevem atividades como esta apresentada:

Code.org – Lição 3: Mapas divertidos

<https://studio.code.org/s/coursea-2022/lessons/3#activity-630427>

Stem Family – Human Algorithm Desing

<https://www.stem.family/2018/11/26/human-algorithm-design/>