

# Pensamento Computacional



Cruzeiro do Sul Virtual  
Educação a distância



# Material Teórico



Abstração

**Responsável pelo Conteúdo:**

Prof. Me. Hugo Batista Fernandes

**Revisão Textual:**

Prof.<sup>a</sup> Dr.<sup>a</sup> Selma Aparecida Cesarin



# UNIDADE

## Abstração



- **Introdução;**
- **Promovendo Abstração na Prática: Jogo de Cartas Binário;**
- **Par ou Ímpar: Criando um Programa no *Scratch*.**



### OBJETIVOS DE APRENDIZADO

- Explorar e promover um dos conceitos mais importantes do Pensamento Computacional: a **abstração**, um conceito que ajuda a entender um problema e a planejar uma boa solução;
- Utilizar atividades desplugadas e também Programação de Computadores por meio do *Scratch*.





# Orientações de estudo

Para que o conteúdo desta Disciplina seja bem aproveitado e haja maior aplicabilidade na sua formação acadêmica e atuação profissional, siga algumas recomendações básicas:

Determine um horário fixo para estudar.

Mantenha o foco! Evite se distrair com as redes sociais.

Procure manter contato com seus colegas e tutores para trocar ideias! Isso amplia a aprendizagem.

Seja original! Nunca plágie trabalhos.

Aproveite as indicações de Material Complementar.

Conserve seu material e local de estudos sempre organizados.

Não se esqueça de se alimentar e de se manter hidratado.

## Assim:

- ✓ Organize seus estudos de maneira que passem a fazer parte da sua rotina. Por exemplo, você poderá determinar um dia e horário fixos como seu “momento do estudo”;
- ✓ Procure se alimentar e se hidratar quando for estudar; lembre-se de que uma alimentação saudável pode proporcionar melhor aproveitamento do estudo;
- ✓ No material de cada Unidade, há leituras indicadas e, entre elas, artigos científicos, livros, vídeos e sites para aprofundar os conhecimentos adquiridos ao longo da Unidade. Além disso, você também encontrará sugestões de conteúdo extra no item **Material Complementar**, que ampliarão sua interpretação e auxiliarão no pleno entendimento dos temas abordados;
- ✓ Após o contato com o conteúdo proposto, participe dos debates mediados em fóruns de discussão, pois irão auxiliar a verificar o quanto você absorveu de conhecimento, além de propiciar o contato com seus colegas e tutores, o que se apresenta como rico espaço de troca de ideias e de aprendizagem.

# Introdução

A **abstração** é sobre simplificar as coisas, permitindo-nos gerenciar a complexidade e tornar as coisas mais fáceis de fazer. A abstração nos permite criar uma ideia geral de qual é o problema e de como resolvê-lo.

O processo de abstração remove todos os detalhes específicos, e quaisquer padrões que não nos ajudem a resolver nosso problema. Isso nos ajuda a formar nossa ideia do problema que é conhecida como “modelo”.



Pensamento Abstrato. Acesse: <http://bit.ly/2XP1BuH>

Se estivéssemos aprendendo/ensinando sobre os países do mundo, por exemplo, utilizariíamos um Mapa Mundial, ou seja, um modelo.

Um Mapa Mundial que mostra os países é uma abstração. É reduzido aos “países do mundo”. Uma outra abstração seria um Mapa com apenas 7 Continentes nomeados.

Mas se estivéssemos andando em São Paulo, um Mapa Mundial não teria detalhes suficientes para nos ajudar. Precisaríamos de um Mapa local que indicasse as ruas, os parques, os bairros etc. Nesse contexto, há muito mais detalhes, porque seria o necessário à situação.

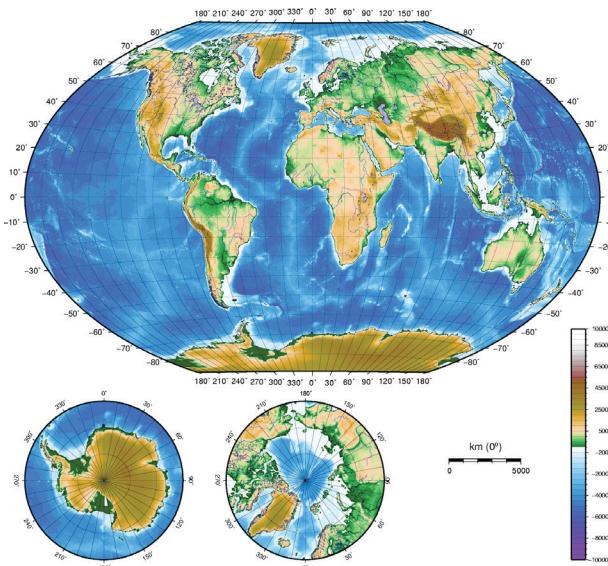


Figura 1 – Mapa Mundi

Fonte: Wikimedia Commons

Na Matemática, quando recorremos a letras e outros símbolos para representar números e quantidades, fórmulas e equações, estamos utilizando o poder da abstração.

Não é mais necessário depender de objetos do mundo real para resolver enigmas matemáticos; graças à abstração, pode-se generalizar procedimentos e aplicar a solução de um problema semelhante a diversas outras situações.

Em Computação, a maioria dos profissionais não precisará utilizar ou conhecer a fundo Álgebra e Estatística; porém a principal lição da Matemática é como abstrair a vasta complexidade do mundo.

Por exemplo, o tratamento de dados de diferentes representações numéricas como:

- Números inteiros;
- Números decimais;
- Números hexadecimais.

Qualquer dado (texto, imagens, vídeo etc.) pode ser expresso como uma lista gigante de números com uma combinação dos elementos acima.

Outro exemplo de abstração em Computação são os números binários. Nos computadores, os dados digitais, o armazenamento, o processamento e as comunicações são gravados e processados como números binários, numa sequência de 0 e 1.

Todos os dados que um computador processa precisam ser convertidos em binário. Os 0 e 1 em binário são usados para representar DESLIGADO ou LIGADO, respectivamente, ou seja, desligar ou ligar um sinal elétrico.

Números binários são importantes porque usá-los em vez do Sistema Decimal simplifica o *design* de Computadores e Tecnologias relacionadas.

A definição mais simples do Sistema Numérico Binário é um Sistema de Numeração que usa apenas dois dígitos – 0 e 1 – para representar números, em vez de usar os dígitos 1 a 9 mais 0 para representar números.



O que são números binários? Acesse: <http://bit.ly/2DyYAs>

## Promovendo Abstração na Prática: Jogo de Cartas Binário

Iremos agora explorar uma atividade descrita no livro **Computer Science Unplugged**, versão traduzida para Português<sup>1</sup>.

A atividade **Contando os Pontos – Números Binários** trabalha o conceito de abstração, utilizando e exigindo habilidades matemáticas de correlação, soma e ordenação.

<sup>1</sup> Disponível em: <https://classic.csunplugged.org/wp-content/uploads/2014/12/CSUnpluggedTeachers-portuguese-brazil-feb-2011.pdf>.

Nessa atividade, o aluno irá converter números decimais em números binários utilizando cinco cartões impressos, cada um contendo uma quantidade específica de pontos.

Esses cartões são apresentados na Figura 2.

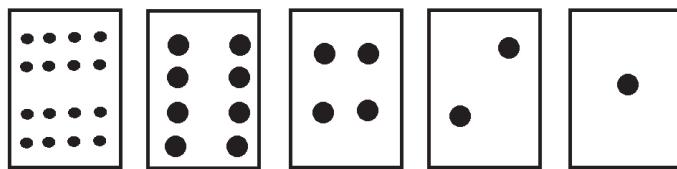


Figura 2 – Cartões da Atividade Contando os Pontos

Fonte: Adaptado de Bell; Witten e Fellows, 2015, p.11

No cartão 2, da direita para esquerda, temos a primeira carta com apenas um ponto. Essa carta representa o número DECIMAL 1, a carta seguinte o número decimal 2, e assim por diante até chegarmos à última carta contendo 16 pontos representando o número decimal 16.

Veja a Figura 3.

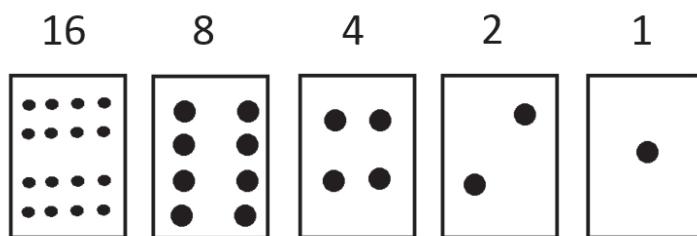


Figura 3 – Cartas representando os números decimais

Fonte: Acervo do Conteudista

As cartas contêm essas quantidades de pontos especificamente pois, na representação BINÁRIA, os números seguem uma sequência de aumento em potências de 2 (BASE 2).

Veja a Figura 4.

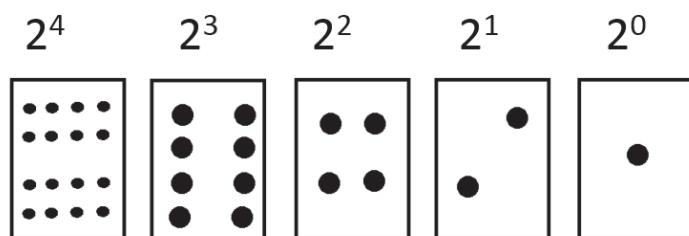


Figura 4 – Cartas representando as potências de 2

Fonte: Acervo do Conteudista

Nessa configuração, podemos representar números binários de 5 posições (5 cartas), porém, em contrapartida, podemos representar, levando em consideração o número 0,32 números decimais.

Em nossa atividade, cada carta representa um *BIT* e, como ocorre no computador, um *bit* LIGADO representa o número 1 e o *bit* DESLIGADO representa o número 0.

Iremos considerar as cartas “viradas para cima”, ou seja, exibindo os pontos, como um bit LIGADO. Cartas viradas para baixo, serão consideradas um bit DESLIGADO.

Vejamos um exemplo na Figura 5.

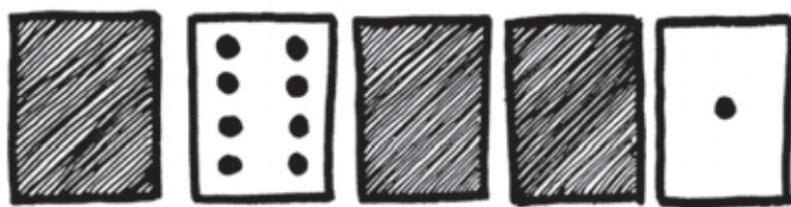


Figura 5 – Sequência de cartas

Fonte: Acervo do Conteudista

Seguindo a lógica de que cartas para cima representam o número 1 e cartas para baixo representam o número 0, temos a seguinte sequência de número BINÁRIO.

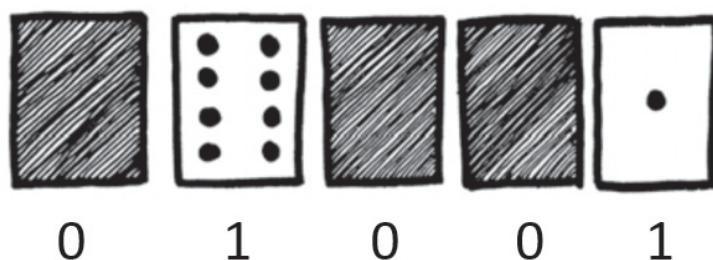


Figura 6

Fonte: Acervo do Conteudista

A configuração de cartas representa o número binário: **01001**. Para converter para número DECIMAL, basta somar a quantidade de pontos das cartas viradas. Desse modo, o número DECIMAL é 9 (8+1).

Vejamos o próximo exemplo de conversão de número binário para decimal.

Observe a configuração da Imagem 7.



Figura 7

Fonte: Acervo do Conteudista

Para converter essa sequência em número decimal, devemos primeiro descrever o número 0 para cartas viradas para baixo e o número 1 para cartas viradas para cima.

Desse modo, temos:

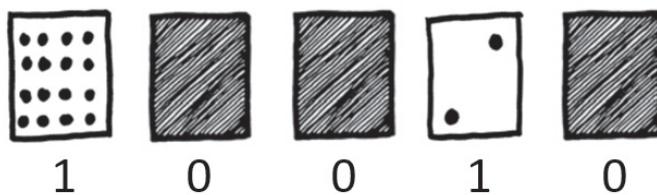


Figura 8

Fonte: Acervo do Conteudista

Agora, somamos a quantidade de pontos de cartas viradas para cima (carta com 16 pontos + carta com 2 pontos). Como resultado, a representação DECIMAL de 10010 é **18**.

## Convertendo Números Decimais em Números Binários

Seguindo a mesma linha de raciocínio de cartas viradas para cima e de cartas viradas para baixo, podemos converter um número decimal para binário.

Para converter um número decimal para binário, devemos partir com todas as cartas viradas para cima.

No exemplo a seguir, iremos converter o número 28 decimal para sua representação em binário.

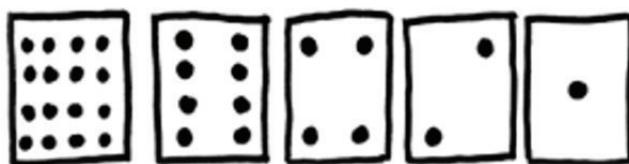


Figura 9

Fonte: Acervo do Conteudista

Esse tipo de conversão consiste em distribuir o valor decimal entre as cartas viradas, partindo da esquerda para direita.

Deve-se distribuir o valor que “cabe” em cada carta e a cada distribuição, subtrair o valor total pelo número de pontos da carta e seguir adiante com a distribuição, até chegar a 0.

Para cada carta que conseguimos distribuir o valor, representamos com o número 1; as cartas não utilizadas, com o número 0. Nunca se deve distribuir um valor a uma carta maior que o valor que se deseja distribuir.

Nesse cenário, temos o seguinte resultado:

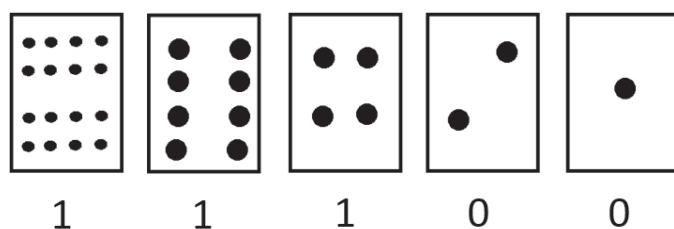


Figura 10  
 Fonte: Acervo do Conteudista

Começando da direita para esquerda, distribuímos o primeiro valor (28) na primeira carta (16 pontos). Efetuando a subtração do valor total (28) pelo número de pontos da carta (16), temos como resultado 12. Continuando a sequência, distribuímos o valor 12 na próxima carta (8 pontos), efetuamos novamente a subtração (12-8) e temos agora o valor 4, distribuímos o valor 4 na carta seguinte (4 pontos).

Para essas três primeiras cartas, representamos o número “1”, pois as utilizamos para distribuir o valor decimal; o restante das cartas, como não utilizadas, representam o número 0.



### Trocando ideias...

Pode-se fazer o *download* de imagens dos cartões da atividade: Contando os pontos diretamente do site *CS Unplugged*. Para isso, acesse o link: <http://bit.ly/2W6PSHy>.

# Par ou Ímpar: Criando um Programa no Scratch

Tomemos como exemplo a tarefa de identificar se um dado número é par ou ímpar.

Como identificar se um número, qualquer que seja seu tamanho, é par ou ímpar? Como apresentar essa regra a um computador?

Para responder e “ensinar” um computador a identificar se um número é par ou ímpar, podemos utilizar um tipo de abstração matemática chamada Aritmética Modular.

Para identificarmos se um número é par, basta dividi-lo por 2; se o resto da divisão resultar em 0, o número é par; caso contrário, o número é ímpar. Para fazer esse cálculo, podemos utilizar a expressão “ $A \bmod 2$ ”. Por exemplo, se quisermos verificar se o número 5 é par, faríamos: “ $5 \bmod 2$ ”, essa expressão resulta em “1”, ou seja, não é par; logo, é ímpar.

Apresentada essa regra, iremos desenvolver um programa de computador utilizando o *Scratch*.



Para acessar a plataforma *Scratch*, abra seu navegador web (em nossos exemplos, iremos utilizar o navegador *Google Chrome*) e acesse o seguinte link: <https://scratch.mit.edu/>

A Figura 11 apresenta a tela inicial da plataforma *Scratch*.

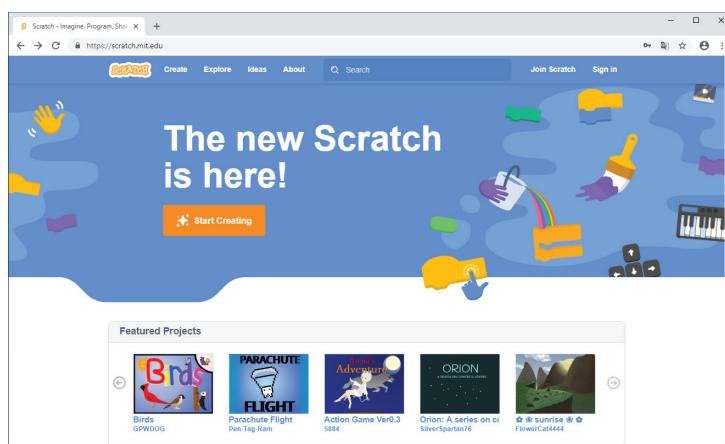


Figura 11 – Página Inicial

Para traduzir a Plataforma para Língua Portuguesa, desça até o final da página e selecione a opção correspondente ao nosso idioma, conforme apresentado na Figura 12.

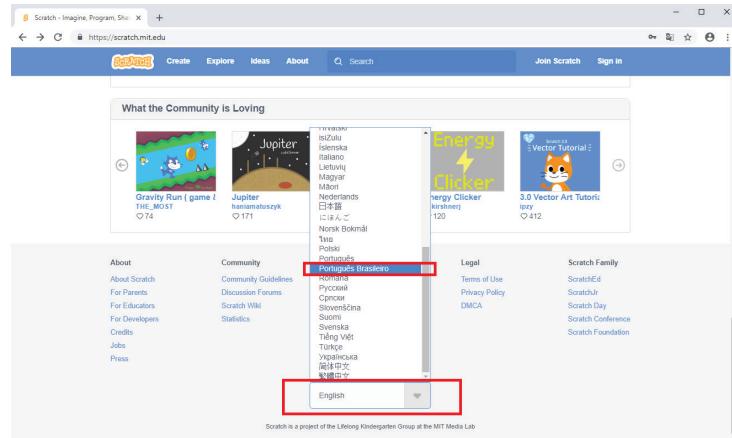


Figura 12 – Traduzindo a linguagem da plataforma

No topo da página inicial, estão posicionados os menus: Criar, Explorar, Ideias, Sobre, Buscar e, por fim, Inscrever-se e Entrar.

Na Tabela a seguir, são descritas as funcionalidades de cada menu.

Tabela 1 – Funcionalidades de cada Menu

Menu	Descrição
Criar	Por meio deste menu, a Plataforma é direcionada para área de criação de Programas do <i>Scratch</i> .
Explorar	Este menu dá acesso à página com Projetos compartilhados por usuários da comunidade <i>Scratch</i> .
Ideias	Esta página exibe uma série de tutoriais <i>Scratch</i> .
Sobre	Página sobre a Plataforma <i>Scratch</i> .
Buscar	Esse menu permite buscar Projetos compartilhados.
Inscrever-se	Por meio deste menu é possível criar um usuário de acesso à Plataforma.
Entrar	Esse menu permite logar na Plataforma.

Fonte: Elaborado pelo autor a partir dos dados obtidos no site *Scratch*

A Figura 13 destaca os menus superiores, bem como os Projetos compartilhados pela comunidade do *Scratch* na parte inferior da página.

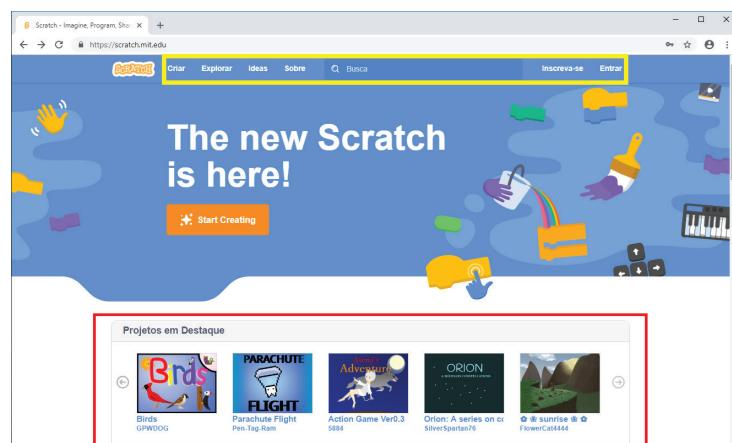


Figura 13 – Menus da página inicial



## Trocando ideias...

Para criarmos Programas no *Scratch*, não é necessário logar na Plataforma, porém é muito interessante que você crie uma conta e faça o *login*, pois dessa maneira todos os seus Projetos ficarão salvos e acessíveis e você também poderá compartilhá-los com outras pessoas.

Para criar Programas no *Scratch*, basta clicar no menu Criar.

Ao clicar no menu Criar, acessamos a página de criação do *Scratch*. Na primeira exibição dessa página, poderá ser exibido um tutorial. Para os nossos estudos, devemos **fechar** esse tutorial.

A Figura 14 destaca o tutorial.

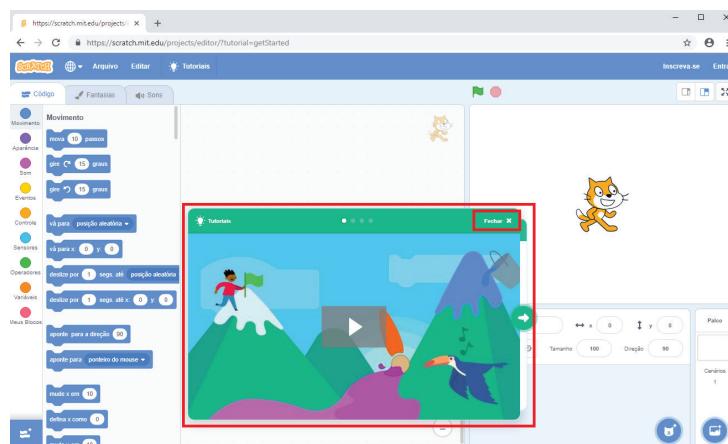


Figura 14 – Tutorial da página de criação

Antes de começarmos a criar nosso primeiro programa, iremos explorar a página de criação destacada na Figura 15.

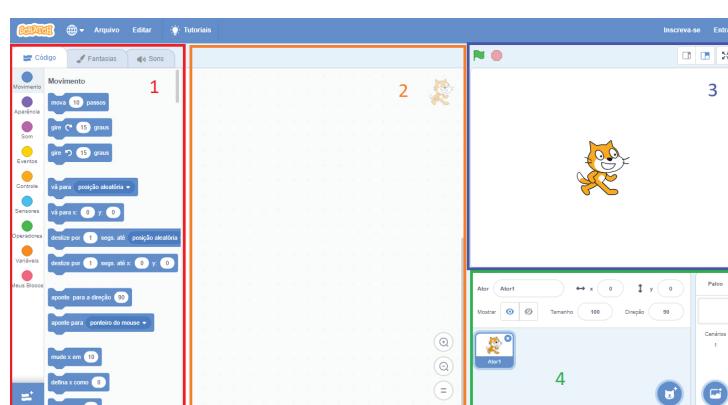


Figura 15 – Página de criação

Podemos observar as áreas identificadas a seguir.

## Área 1

---

Temos os Blocos de Comando do *Scratch* ordenados em categorias. Cada categoria possui um conjunto de blocos de comando.

O *Scratch* disponibiliza 8 categorias: Movimento, Aparência, Som, Eventos, Controle, Sensores, Operacionais e Variáveis.

A categoria Meus Blocos é utilizada quando o usuário cria seus próprios blocos. Ainda na Área 1 temos acesso às configurações de “Fantasias” que são as imagens que utilizamos para os nossos atores (área 3) no *Scratch* e também configurações de sons que podem ser reproduzidos.

## Área 2

---

Os Blocos de Comando são arrastados para essa área. A lógica de programação e nosso algoritmo e, desse modo, os programas são criados nessa área.

## Área 3

---

Os atores (personagens), as saídas de nossos Programas, são exibidos nessa área. Vemos o resultado de nossa programação nessa área.

## Área 4

---

No *Scratch*, o Código de Programação, ou seja, os Blocos de Comando, são vinculados ao ator ou ao palco (Plano de Fundo). Nessa área, é possível navegar entre os atores e/ou o palco para criar os Códigos de Programação além de dar acesso às configurações de tamanho e de posicionamento (eixos X e Y) do ator.

Em nosso programa, iremos criar um Código que, ao ser executado, irá solicitar um número, identificar se o número é par ou ímpar e, em seguida, exibir na tela a resposta, ou seja, se o número é par ou ímpar.

Primeiro, devemos configurar um evento que irá disparar a ação que iremos criar com os Códigos do *Scratch*.

Desse modo, na Área de Blocos de Comando, localize a categoria “Eventos”, clique com o botão esquerdo do mouse sobre o comando  e, sem soltar o botão do mouse, arraste para a área de Código, como podemos ver na Figura 16.

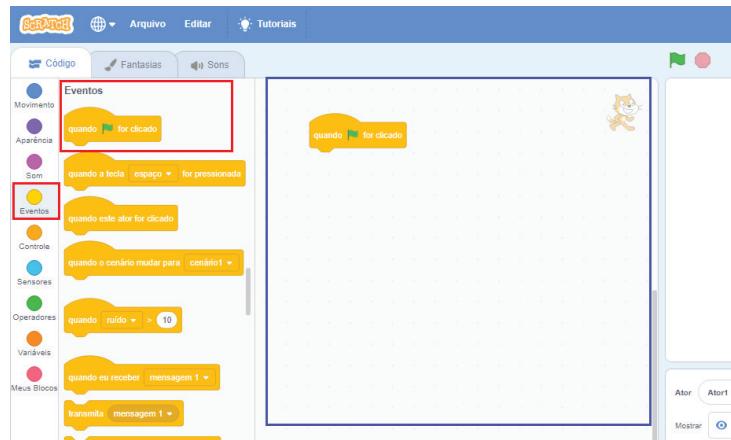


Figura 16 – Arrastando um bloco de eventos

Em seguida, localize a categoria “Sensores” e arraste o bloco (“Pergunte e espere”) para a Área de Código, “encaixe” logo abaixo do Bloco de eventos e altere o texto do Bloco para “Digite um número”, conforme podemos ver na Figura 17.

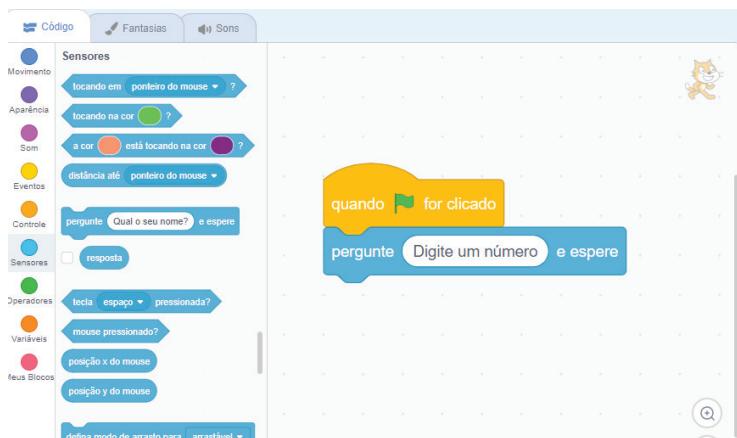


Figura 17 – Arrastando um bloco pergunte

Agora, localize a categoria “Controle”, arraste o Bloco representado abaixo (“se então” para a área de Código e “encaixe” logo abaixo do Bloco “pergunte”, conforme podemos ver na Figura 18.



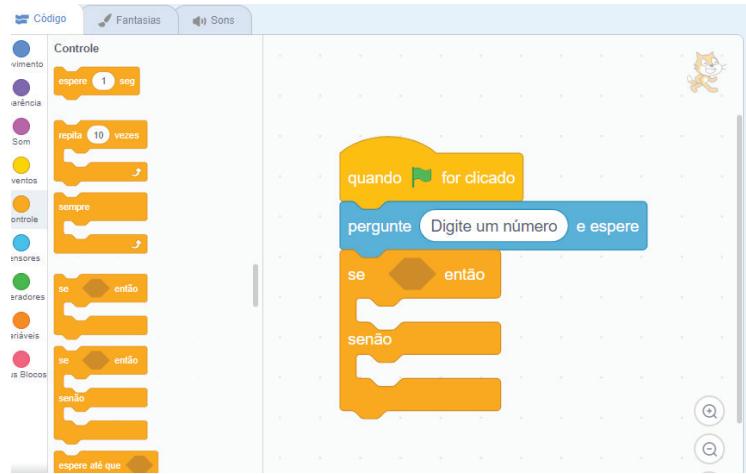


Figura 18 – Arrastando um Bloco condicional

Até o momento, nosso Programa já consegue solicitar um número, porém ainda não aplicamos a lógica matemática para identificar se um número é par ou ímpar.

Para isso, iremos utilizar um operador de igualdade e a função “resto de” que, no Scratch, calcula e retorna o resto da divisão de um número.

Comece arrastando e encaixando o comando  $\text{[número] } = 50$  para dentro do encaixe do bloco se então, conforme podemos ver na Figura 19.



Figura 19 – Arrastando um bloco de operador de igualdade

Em seguida, ainda na categoria Operadores, localize e arraste o comando **resto de por** para dentro do primeiro encaixe do comando de operação de igualdade, conforme podemos ver na Figura 20.



Figura 20 – Arrastando um Bloco resto de por

O passo seguinte será vincular o número digitado pelo usuário de nosso Programa à operação matemática. Para isso, na categoria Sensores, localize e arraste o comando **resposta** para o primeiro encaixe do comando **resto de por**, conforme podemos ver na Figura 21.



Figura 21 – Arrastando um Bloco resposta

Retomando a regra para identificar se algum número é par ou ímpar, devemos verificar o resto da divisão por 2: se for igual a zero, esse número é par; caso contrário, o número é ímpar. Para isso, precisamos, agora, digitar o número 2 no espaço em branco do comando **resto de por** e alterar o número 50 do operador de igualdade por 2.

Podemos ver essa configuração na Figura 22.



Figura 22 – Alterando valores de operadores

Nosso programa já consegue identificar se um número digitado pelo usuário é par ou ímpar; porém nosso Programa ainda não exibe uma mensagem de acordo com a resposta. Para isso, na categoria Aparência, localize e arraste um comando **diga** para dentro da lacuna do comando **se então**.

Veja a Figura 23: altere o texto do comando dia para “O número é par”.



Figura 23 – Arrastando um bloco diga

Por fim, iremos criar a programação que irá exibir uma mensagem caso o número digitado seja um número ímpar. Para isso, arraste um comando **diga** para dentro da lacuna do comando **se então**.

Veja a Figura 24.

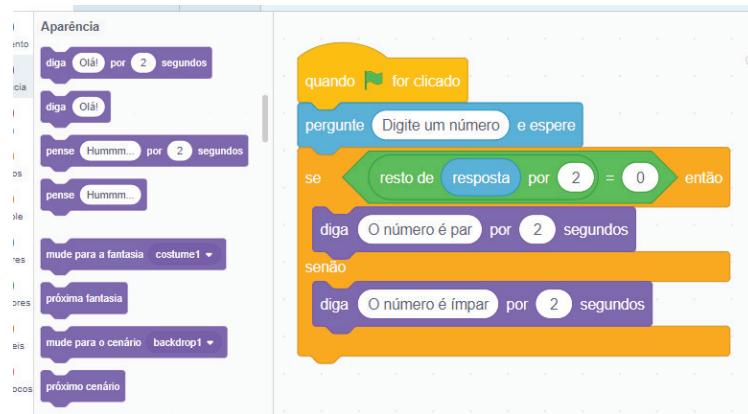


Figura 24 – Arrastando um bloco diga

Concluímos nosso Programa, agora devemos testá-lo. Para isso, clique no local indicado na Figura 25.

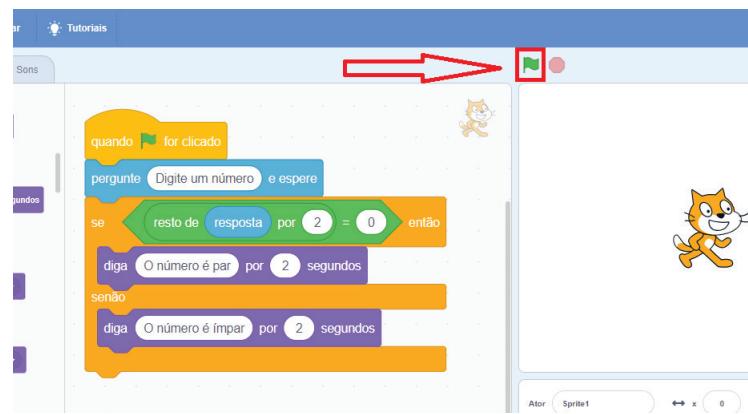


Figura 25 – Testando o programa

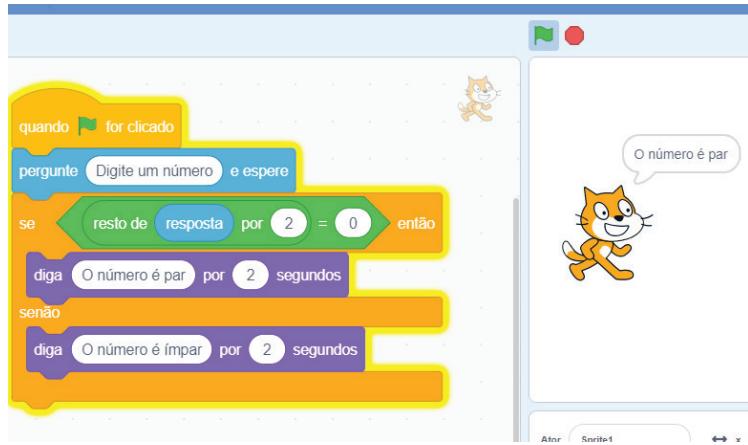


Figura 26 – Testando o programa



## Em Síntese

Nesta Unidade, exploramos o conceito de abstração utilizando atividades desplugadas e também criando um programa com o *Scratch*.

Como sugestão, para aprofundamento na utilização do *Scratch*, é relevante o estudo e o acesso aos tutoriais disponibilizados na Plataforma *Scratch*. Para isso, acesse o *link*:

<http://bit.ly/2GEPPhWZ>.

# Material Complementar

Indicações para saber mais sobre os assuntos abordados nesta Unidade:



Sites

Aritmética do Relógio

<http://bit.ly/2UZ0c84>



Vídeos

Introdução ao *Scratch* - Curso de Algoritmos #05 - Gustavo Guanabara

<https://youtu.be/GrPkuk1ezyo>

Lógica de Programação com *Scratch* – Condicionais e Mais Operadores

<https://youtu.be/YiPClvltJnc>



Leitura

O Software Binário e a Matemática (Parte I)

<http://bit.ly/2GB7XXP>

# Referências

BBC LEARNING, B. *What is computational thinking?* 2018. Disponível em: <<http://www.bbc.co.uk/education/guides/zp92mp3/revision>>. Acesso em: 1 fev. 2018.

BELL, T.; WITTEN, I. H.; FELLOWS, M. *Computer Science Unplugged: An enrichment and extension programme for primary-aged students.* 2015. Disponível em: <[https://classic.csunplugged.org/wp-content/uploads/2015/03/CSUnplugged\\_OS\\_2015\\_v3.1.pdf](https://classic.csunplugged.org/wp-content/uploads/2015/03/CSUnplugged_OS_2015_v3.1.pdf)>. Acesso em: 20 abr. 2019.

WING, J. M. *Computational thinking.* *Communications of the ACM*, v. 49, n. 3, p. 33-35, 2006. Disponível em: <<https://www.cs.cmu.edu/~./15110-s13/Wing06-ct.pdf>> . Acesso em: 20 abr. 2019.



**Cruzeiro do Sul**  
Educacional