

Pensamento Computacional



Cruzeiro do Sul Virtual
Educação a distância

Material Teórico



Reconhecimento de Padrões

Responsável pelo Conteúdo:

Prof. Me. Hugo Batista Fernandes

Revisão Textual:

Prof.^a Dr.^a Selma Aparecida Cesarin

UNIDADE

Reconhecimento de Padrões



- Introdução;
- Reconhecimento de Padrões na Prática:
Criando um Programa no Scratch;
- Reconhecendo Padrões e Criando Soluções.



OBJETIVO DE APRENDIZADO

- Explorar os conceitos de Reconhecimento de Padrões, utilizando como abordagem para sua promoção a Programação de Computadores por meio da Plataforma *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

O Reconhecimento de Padrões é a capacidade de notar semelhanças ou diferenças comuns em dados que nos ajudarão a fazer previsões ou nos levarão por meio de um caminho mais curto a uma solução; frequentemente, é a base para resolver problemas e projetar algoritmos.

Também pode ser definido como a classificação de dados com base no conhecimento já obtido ou em informações estatísticas extraídas de padrões e/ou sua representação.

O Reconhecimento de Padrões é, ainda, essencial para muitas Áreas da Ciência da Computação, incluindo análise de *Big Data*, identificação biométrica, segurança e Inteligência Artificial (IA).

Um problema de computação comum é reconhecer padrões em sequências numéricas. Por exemplo, tente identificar o “X” nas sequências a seguir:

1. 3, 6, 9, 12, 15, X;
2. 1, 3 ,5, 7, 9, 11, X;
3. 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, X.

A estratégia para resolver essa questão está em identificar os padrões em que os números são exibidos.

Para a sequência “a”, note que os números estão em ordem crescente, sempre somando 3. Desse modo, temos $3 + 3 = 6$, $6+3 = 9$, $9+3 = 12$ etc. Assim, X é igual **18** ($15+3$).

A sequência “b” segue o mesmo padrão, porém somando 2. Assim, X é igual **13** ($11+2$).

Por fim, na sequência “c”, o padrão que os números apresentam é mais complexo. Primeiramente soma-se 0 + 1. A partir desse primeiro procedimento, soma-se o resultado com o número anterior. Assim, temos:

0, 1, **1**

1, 1, **2**

1, 2, **3**

2, 3, **5**

3, 5, **8**

5, 8, **13**

Logo, X é igual a **55** ($34+21$).



Reconhecimento de Padrões também é muito utilizado em aplicação de Inteligência Artificial. Saiba mais em: <http://bit.ly/2E42eJN>

A habilidade de Reconhecimento de Padrões é, sobretudo, uma maneira que nos ajuda a resolver rapidamente novos problemas com base em soluções e experiências anteriores.

Reconhecimento de Padrões na Prática: Criando um Programa no Scratch

Para o nosso estudo prático, iremos criar um Programa em que o personagem se movimentará na tela em quatro direções: para cima, para baixo, esquerda e direita.

Ao se movimentar, nosso ator mudará de cor, deixando um rastro colorido por onde passar.

Primeiro, devemos configurar o evento responsável pelo Código que fará nosso ator se movimentar. Primeiro, iremos codificar a movimentação para o lado direito.

Para isso, localize, na aba Eventos, o comando “quando a tecla for pressionada” e o arraste para a área de codificação.



Figura 1

Em seguida, clique na seta de configuração do comando e selecione “seta para direita”, como se pode observar na Figura 2.

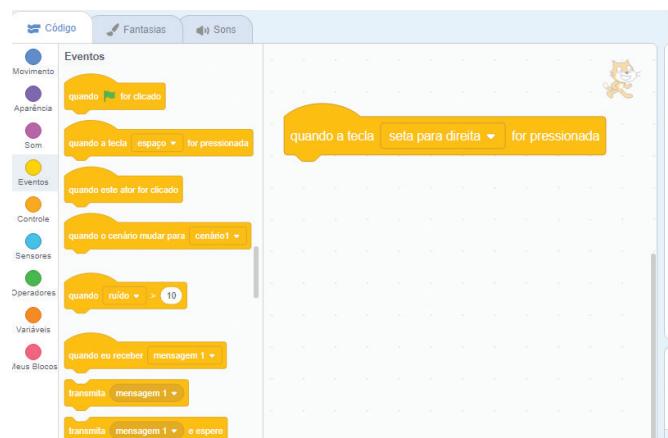


Figura 2 – Arrastando um comando quando a tecla for pressionada

Agora, iremos arrastar os comandos responsáveis por movimentar o ator, para isso, localize a aba Movimento e arraste os comandos “aponte para a direção” e “mova passos”, como se pode ver na Figura 3.

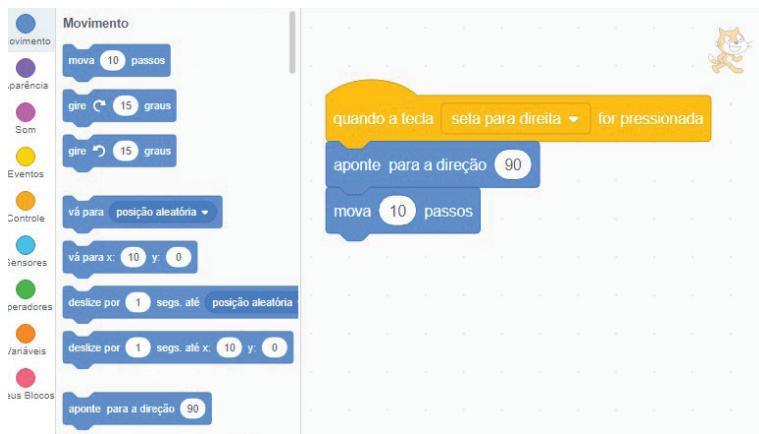


Figura 3 – Arrastando comandos de movimento

Nessa configuração, o comando “aponte para a direção”, quando executado, direciona o ator para “90” graus, ou seja, ele direciona o ator para a direita.

Em seguida, o comando “mova passos” movimenta o ator “10” passos.



Como sugestão, para introdução de **movimentação de atores**, acesse os tutoriais disponibilizados na Plataforma *Scratch*, por meio do *link* e procure o tutorial “Use as setas do teclado”: <http://bit.ly/2LshPZG>

Agora, iremos arrastar os comandos responsáveis por movimentar o ator para a esquerda. Primeiro localize, na aba Eventos, o comando “quando a tecla for pressionada” e o arraste para a área de codificação.

Em seguida, clique na seta de configuração do comando e selecione “seta para **esquerda**”. Na sequência, localize a aba Movimento e arraste os comandos “aponte para a direção” e “mova passos”, como se vê na Figura 4.



Figura 4 – Arrastando comandos de movimento

Nessa configuração, o comando “aponte para a direção”, quando executado, direciona o ator para “-90” graus, ou seja, ele direciona o ator para a esquerda. Em seguida, o comando “mova passos” movimenta o ator “10” passos.

Iremos, agora, arrastar os comandos responsáveis por movimentar o ator para cima. Primeiro localize, na aba Eventos, o comando “quando a tecla for pressionada” e o arraste para a área de codificação.

Em seguida, clique na seta de configuração do comando e selecione “seta para cima”. Na sequência, localize a aba Movimento e arraste os comandos “aponte para a direção” e “mova passos”, como se observa na Figura 5.

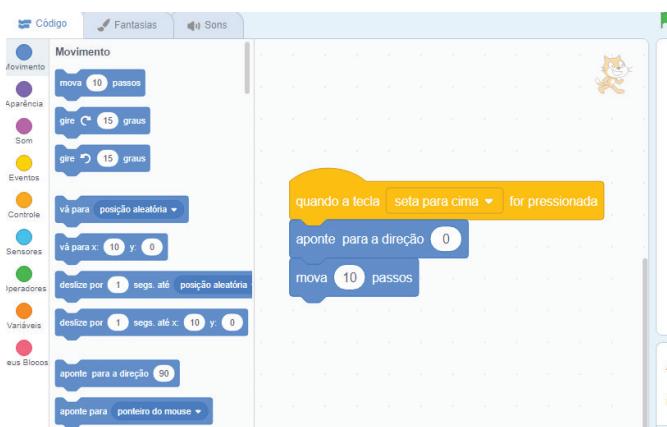


Figura 5 – Arrastando comandos de movimento

Nessa configuração, o comando “aponte para a direção”, quando executado, direciona o ator para “0” graus, ou seja, ele direciona o ator para cima. Em seguida, o comando “mova passos” movimenta o ator “10” passos.

Por fim, iremos arrastar os comandos responsáveis por movimentar o ator para baixo. Primeiro localize, na aba Eventos, o comando “quando a tecla for pressionada” e o arraste para a área de codificação. Em seguida, clique na seta de configuração do comando e selecione “seta para baixo”. Na sequência, localize a aba Movimento e arraste os comandos “aponte para a direção” e “mova passos”, como se vê na Figura 6.



Figura 6 – Arrastando comandos de movimento

Nessa configuração, o comando “aponte para a direção”, quando executado, direciona o ator para “180” graus, ou seja, ele direciona o ator para baixo. Em seguida, o comando “mova passos” movimenta o ator “10” passos.

Com essa sequência de configurações, nosso ator já é capaz de se movimentar pela tela.

Agora iremos arrastar comandos que darão uma sessão de movimento ao ator, alterando sua cor.



Como sugestão, para introdução de movimentação de atores, acesse os tutoriais disponibilizados na Plataforma *Scratch*, por meio do link: <http://bit.ly/2H3bvDr> e procure o tutorial “Anime um ator”.

Para isso, localize na aba Aparência os comandos “próxima fantasia” e “mude ao efeito cor”. Arraste esses comandos na sequência dos comandos responsáveis pela configuração da movimentação de nosso ator para a direita, como se observa na Figura 7.

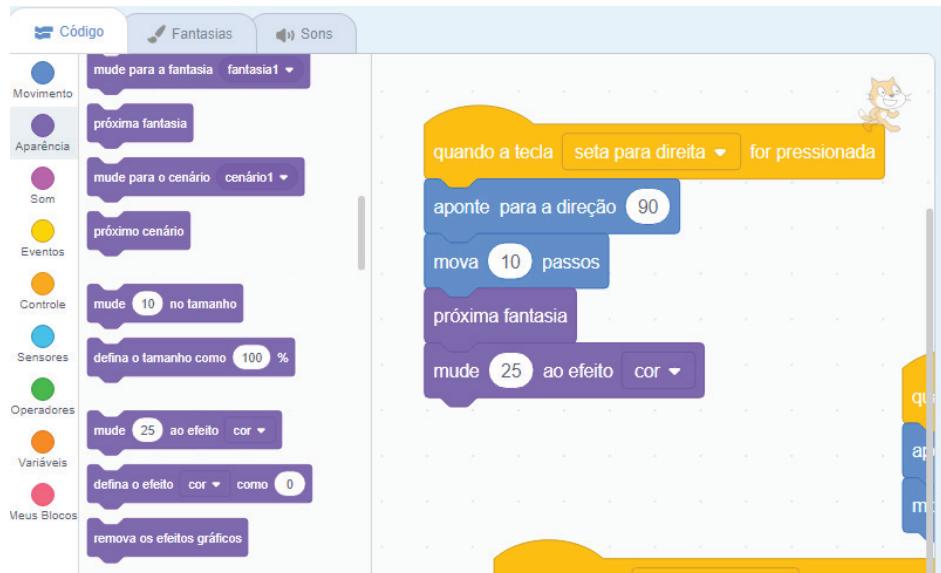


Figura 7 – Arrastando comandos de aparência

Nessa configuração, o comando “próxima fantasia” é responsável por alterar a imagem de nosso ator; já o comando “mude ao efeito cor” altera a cor do ator. Arraste essa mesma sequência para todas as configurações de movimento de nosso ator. Seu código deve ficar como se vê na Figura 8.

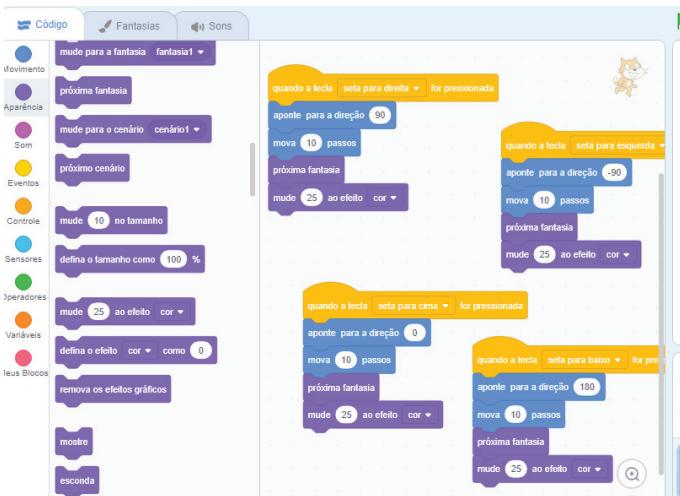


Figura 8 – Códigos de movimento do ator

Para finalizarmos o Programa, devemos criar a configuração responsável por deixar um rastro colorido no caminho de nosso ator. Para isso, iremos utilizar comandos da aba Caneta.

Note que, por padrão, essa aba não está disponível. Assim, devemos adicionar essa aba. Desse modo, na área de seleção de abas, no canto inferior à esquerda, clique no botão Adicionar uma extensão:

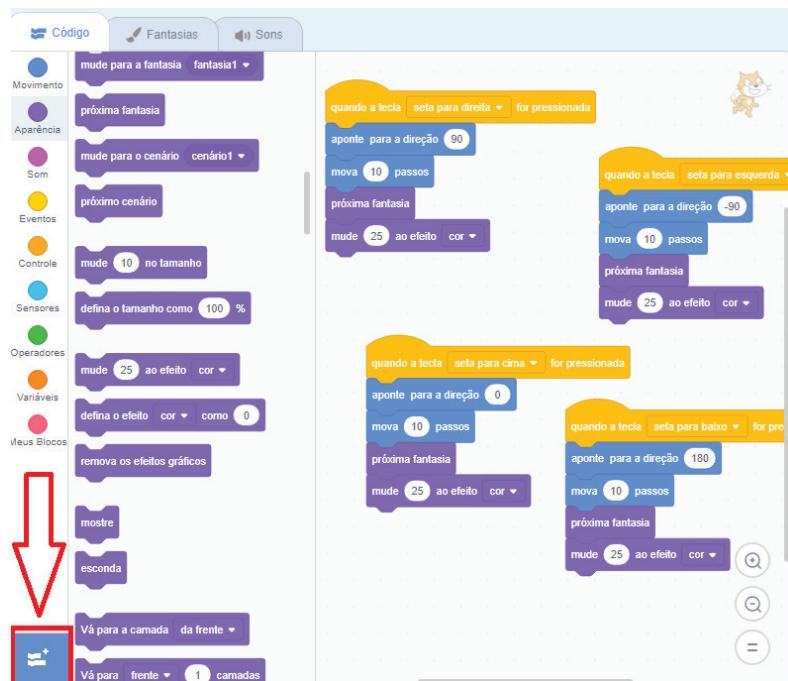


Figura 9 – Adicionando uma extensão

Na tela seguinte, selecione a extensão Caneta:

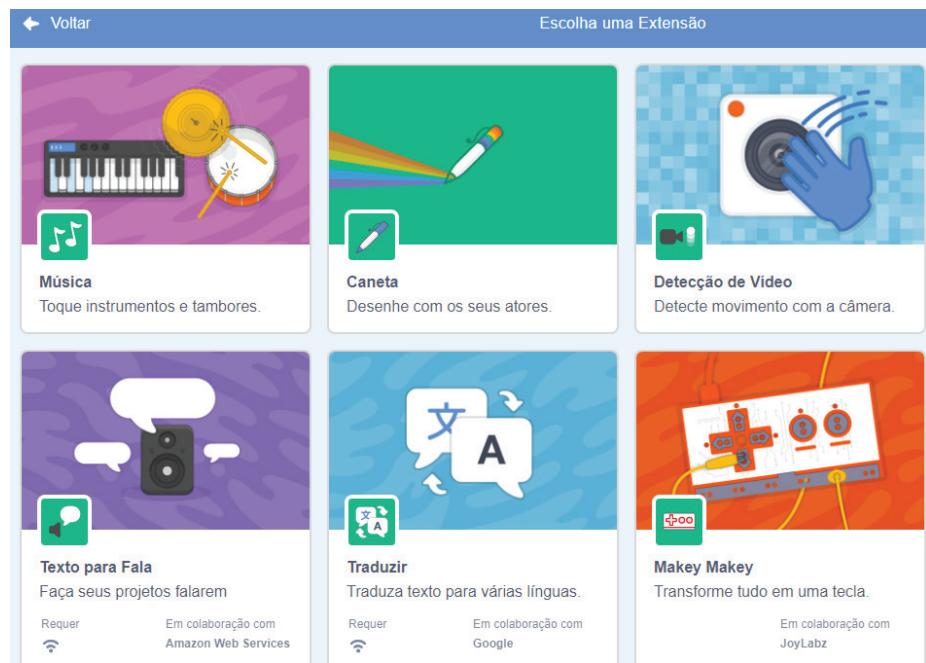


Figura 10 – Extensão Caneta

Esse procedimento faz com que a aba Caneta seja adicionada ao *Scratch*. Agora, na aba Caneta, localize e arraste o comando “Carimbe” para a sequência dos comandos responsáveis pela configuração da movimentação de nosso ator para a direita, como se vê na Figura 11:



Figura 11 – Comando carimbe

Arraste essa mesma sequência para todas as configurações de movimento de nosso ator. Seu código deve ficar como o da Figura 12:

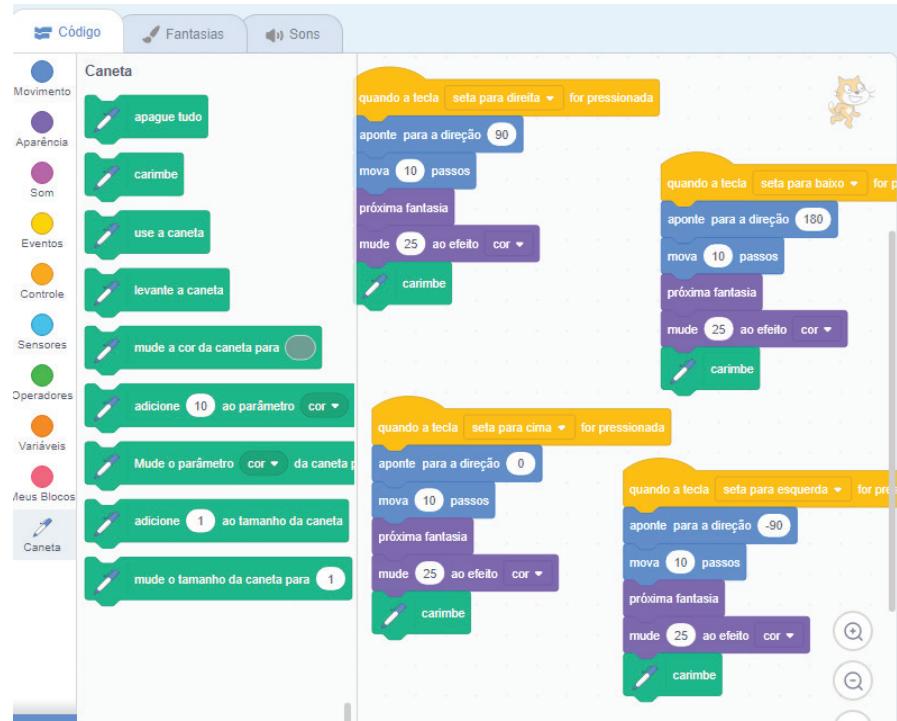


Figura 12 – Comandos para carimbar a imagem do ator na tela

Para finalizar, arraste um Bloco de eventos “quando for clicado”, na sequência na aba Caneta, localize e arraste um comando “apagar tudo”; por fim, arraste dois comandos da aba Movimento: “aponte para a direção” e “vá para X e Y”. Podemos ver essa configuração na Figura 13:

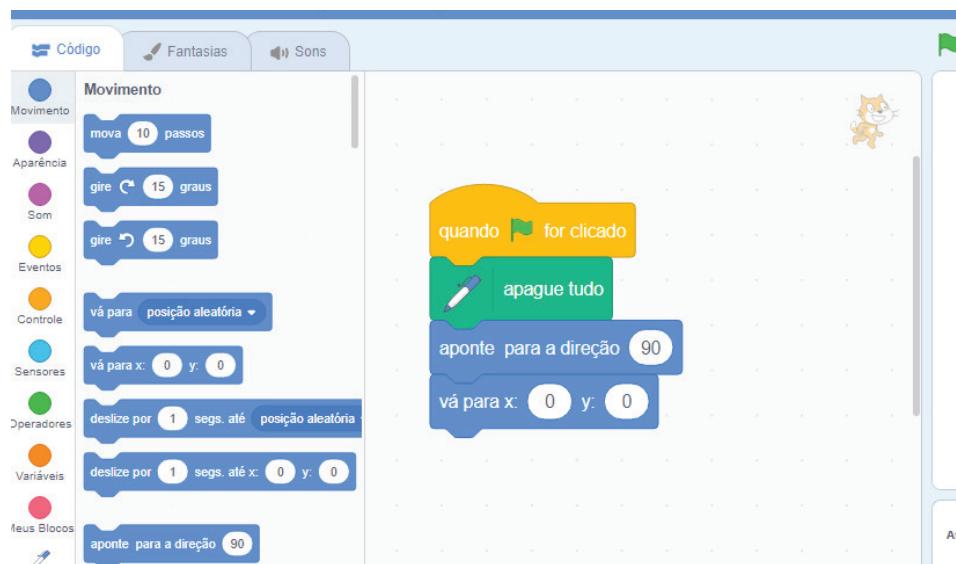


Figura 13 – Limpando a tela

Essa configuração irá garantir que, ao iniciar nosso Programa, a área do ator será limpa e o ator iniciará na posição central.

Testando nosso programa, podemos ver o resultado:

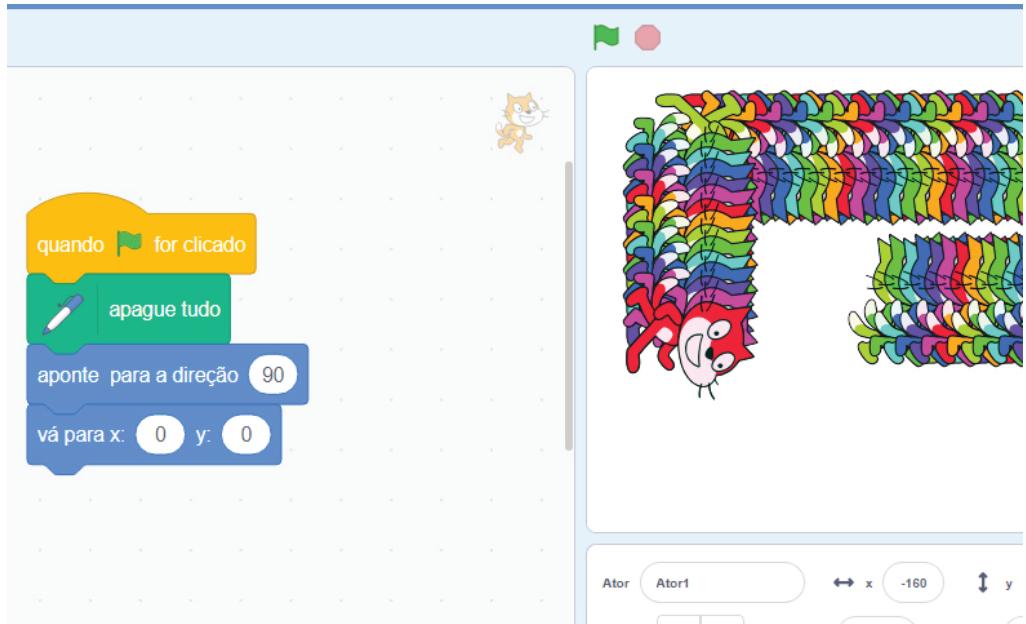


Figura 14 – Testando o programa



Exemplos de Utilização da Caneta. Acesse: <http://bit.ly/2Jg0pwC>

Reconhecendo Padrões e Criando Soluções

Relembrando as definições de Reconhecimento de Padrões, temos que ele nos ajuda a resolver rapidamente novos problemas, com base em soluções e em experiências anteriores.

Em nosso Código, existem alguns padrões ou comandos que se repetem. Se conseguirmos identificá-los, poderemos criar soluções para as tarefas de formas mais fáceis (nesse caso, tarefas dentro do próprio Programa), utilizando um menor esforço, como se fosse um caminho mais rápido para chegar à solução.

Analizando o código, podemos identificar que, logo depois do comando de identificação da tecla pressionada, todo o restante de código é semelhante para todos as configurações de movimento do ator.

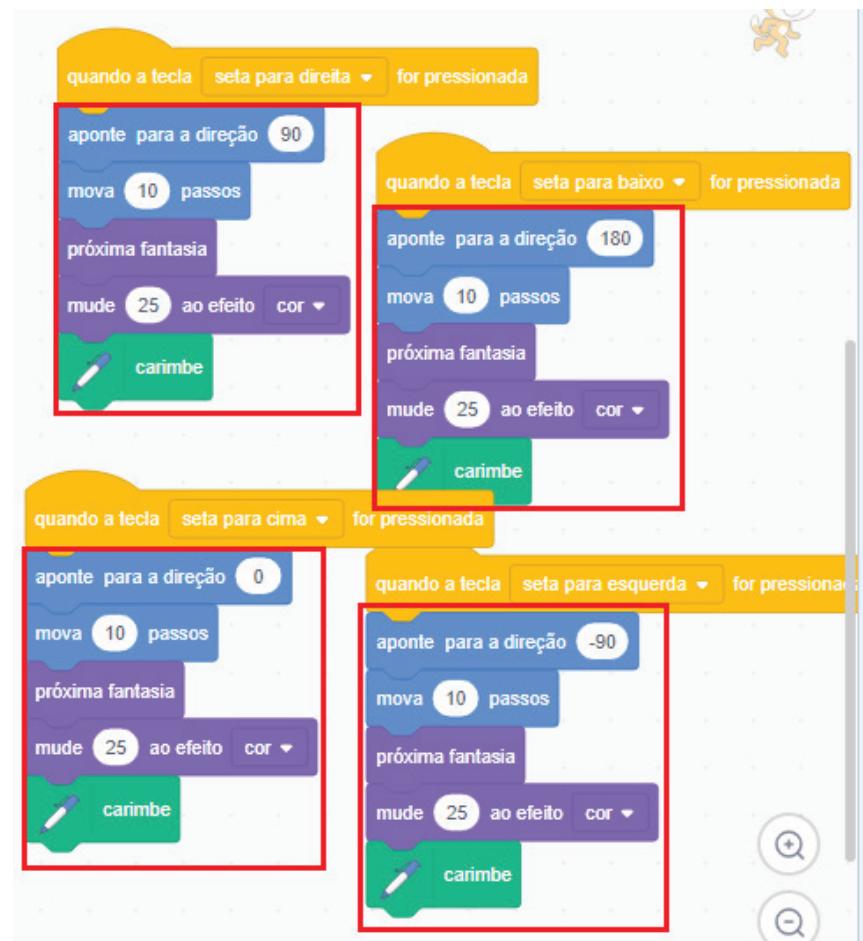


Figura 15 – Código de nosso programa

Encontramos um padrão. Desse modo, conhecendo esse padrão, poderemos criar uma solução mais simples para a tarefa de movimentação do ator. Com isso, iremos, agora, criar apenas uma codificação que servirá para qualquer um dos quatro movimentos possíveis de nosso ator.

Nesse intuito, iremos criar um Bloco especial. Será um Bloco de comando criado por nós. Para isso, localize a aba Meus Blocos e clique no botão Criar um Bloco.

Para o nome do Bloco, digite “movimento”; em seguida, selecione a opção “adicionar uma entrada”, digite nesse campo “direção” e, por fim, clique no botão ok.



Figura 16 – Criando um Bloco

Arraste agora a sequência de Código responsável pelo movimento do ator para direita, assim como podemos ver na Figura 17:



Figura 17 – Configurando o Bloco movimento

Agora arraste o Bloco “Direção” para dentro do campo do comando “aponte para a direção”:



Figura 18 – Alterando a variável

Na área de Código, localize o comando “quando a tecla **seta para direita** for pressionada” e arraste para a sequência o comando Movimento, como podemos ver na Figura 19.

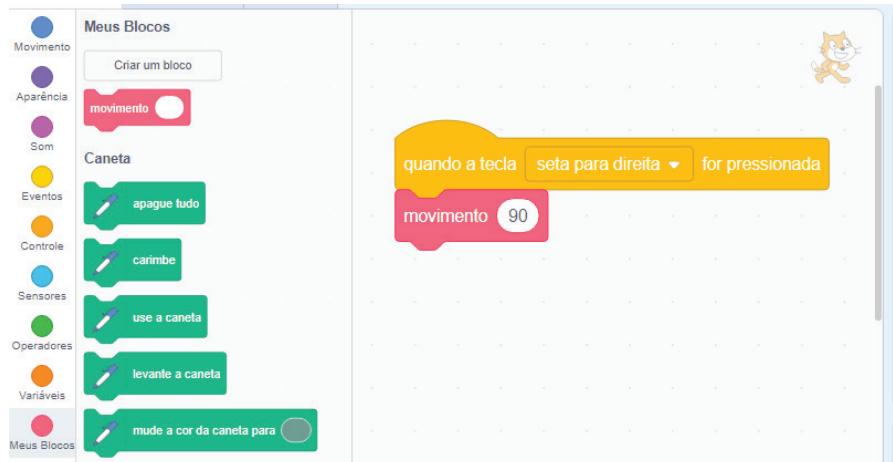


Figura 19 – Configurando o movimento do ator

Porque “90”?

O valor 90 foi adicionado pois é esse o valor que iremos utilizar no comando “aponte para a direção”.

Faça esse procedimento em todos os comandos de “quando a tecla for pressionada”.

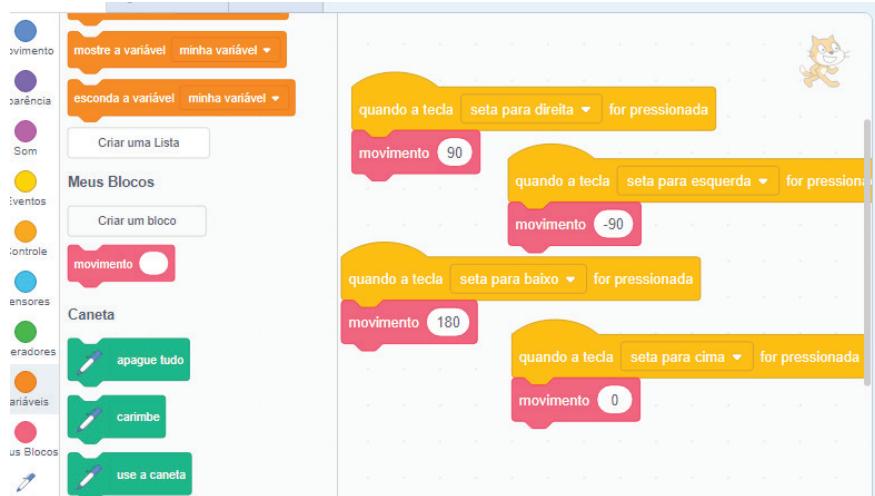


Figura 20 – Configurando o movimento do ator

Comparado a quantidade de Códigos que tínhamos, nosso programa ficou mais claro e intuitivo, isso graças à identificação de padrões que nos possibilitou visualizar um trecho de Código que se repetia.

Criamos um Código mais limpo e, caso seja necessária uma alteração referente ao movimento do ator, só será necessário alterar um trecho de Código, e não quatro trechos, como na versão anterior.

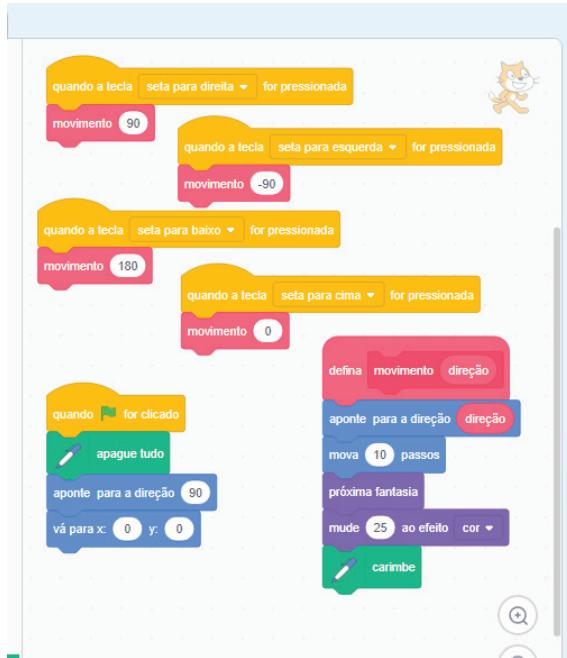


Figura 21 – Programa finalizado

Em conclusão, exploramos o conceito de Reconhecimento de Padrões e, por meio dessa habilidade, conseguimos criar um Programa que soluciona uma tarefa de forma mais simples, utilizando um número menor de recursos e comandos.

Material Complementar

Indicações para saber mais sobre os assuntos abordados nesta Unidade:



Sites

Pensamento Computacional: Uma competência para o futuro

<http://bit.ly/2vljz68>



Vídeos

Scratch – Caneta

<https://youtu.be/qBsY2R9lFzs>

Scratch – Blocos

<https://youtu.be/fU3U5xfA1oU>



Leitura

Entendendo o Pensamento Computacional

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

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>. 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