APRENDENDO MATEMÁTICA COM SCRATCH

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CURITIBA

PROGRAMA DE PÓS-GRADUAÇÃO EM
FORMAÇÃO CIENTÍFICA, EDUCACIONAL E TECNOLÓGICA

ADMILSON IARESK DA SILVA
MARCO AURÉLIO KALINKE

TRABALHANDO MATEMÁTICA COM O *SCRATCH*

PRODUTO EDUCACIONAL

Finalidade: O produto educacional visa disponibilizar ao professor sugestões para o desenvolvimento da lógica e da matemática básica, juntamente com o processo da programação, no ambiente de desenvolvimento de programas do so*ftware Scratch*. Sua aplicação tem por objetivo estimular a compreensão dos conceitos matemático, que faz parte do processo da construção de algoritmos.

CURITIBA
2020

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CURITIBA

PROGRAMA DE PÓS-GRADUAÇÃO EM
FORMAÇÃO CIENTÍFICA, EDUCACIONAL E TECNOLÓGICA

ADMILSON IARESK DA SILVA
MARCO AURÉLIO KALINKE

TRABALHANDO MATEMÁTICA COM O *SCRATCH*

PRODUTO EDUCACIONAL

Produto Educacional apresentado ao Programa de Pós-Graduação em Formação Científica, Educacional e Tecnológica (PPGFCET) da Universidade Tecnológica Federal do Paraná, como requisito para obtenção do grau de mestre em Ensino de Ciências e Matemática. Área de concentração: Ensino, Aprendizagem e Mediações.

Orientador: Prof. Dr. Marco Aurélio Kalinke

CURITIBA
2020

Ministério da Educação

Universidade Tecnológica Federal do Paraná
Campus Curitiba
Diretoria de Pesquisa e Pós-Graduação
Programa de Pós-Graduação em Formação Científica, Educacional e Tecnológica – PPGFCET

TERMO DE LICENCIAMENTO

Dados Internacionais de Catalogação na Publicação

Silva, Admilson Iaresk da
Trabalhando matemática com o *Scratch* [recurso eletrônico] / Admilson Iaresk da Silva, Marco Aurélio Kalinke. -- 2020.
1 arquivo eletrônico (63 f.) : PDF ; 17,5 MB.

Modo de acesso: World Wide Web.

1. Educação básica. 2. Scratch (Linguagem de programação de computador) - Programação. 3. Matemática - Estudo e ensino - Problemas, exercícios etc. 4. Lógica - Estudo e ensino - Problemas, exercícios etc. 5. Aprendizagem. 6. Prática de ensino. 7. Recursos eletrônicos de informação. I. Kalinke, Marco Aurélio. II. Título.

CDD: Ed. 23 -- 507.2

Biblioteca Central do Câmpus Curitiba - UTFPR
Bibliotecária: Luiza Aquemi Matsumoto CRB-9/794

APRESENTAÇÃO

Caros colegas,

Esse material apresenta sugestões para o trabalho com lógica e matemática básica, juntamente com o processo da programação, no ambiente de desenvolvimento de programas do s*oftware Scratch*. As atividades aqui compartilhadas são fruto da pesquisa, que buscou encontrar as possibilidades metodológicas para uso do *software Scratch* na educação básica.

Esse material visa disponibilizar ao professor informações sobre o nosso produto educacional, um *Ebook*. Optamos por apresentar uma breve descrição da história do *Scratch*, posteriormente suas ferramentas, e como elas estão relacionadas com o desenvolvimento dos algoritmos de programação, e sua representação no *Scratch*. Para o desenolvimento da lógica, apresentamos alguns exercícios que procurão primeiramente estimular a compreensão do desenvolvimento da lógica, que faz parte do processo da construção de algoritmos.

No desenvolvimento da lógica matemática, procuramos estimular o raciocínio matemático, também com alguns exercícios, porém estes fazendo a relação com as ferramentas do *software*. Deste modo, nas atividades propostas, somente algumas ferramentas serão utilizadas, permitindo de modo gradual o processo construtivo do algoritmico.

Informamos ainda, que nosso produto resulta da pesquisa de Mestrado Profissional pelo Programa de Pós-Graduação em Formação Científica, Educacional e Tecnológica da Universidade Tecnológica Federal do Paraná. Contemplando a Área de Concentração “Ensino, Aprendizagem e Mediações”, e a Linha de Pesquisa “Mediações por Tecnologias de Informação e Comunicação no Ensino de Ciências e Matemática”.

Consideramos que o processo da programação pode ser relacionado ao aprendizado da matemática, de modo construtivo, pois os conteúdos matemáticos que envolvem operações, expressões e lógicas, tem a possibilidade de serem representados pelos alunos no modelo de pequenos programas. Deste modo, este processo construtivo, realizado pelos alunos no

software Scratch, possibilita a compreensão do desenvolvimento de programas, juntamente com o aprendizado matemático.

Assim, com o produto aqui apresentado, visamos contribuir de modo diferenciado com os processos de ensino e aprendizagem da Matemática, agregando saberes com a apresentação da programação na prática educacional.

Atenciosamente,

Prof. Admilson Iaresk da Silva
Prof. Dr. Marco Aurélio Kalinke

GAZETINHA PROGRAMANDO COM SCRATCH

Curitiba, 2020

Scratch

Scratch – um software para o ensino da programação

Uma possibilidade de incorporar o ensino da programação nos processos educativos foi mostrada pelos pesquisadores Papert e Resnick. A programação determinou nossa escolha pelo *Software Scratch* como proposta de ferramenta educativa.

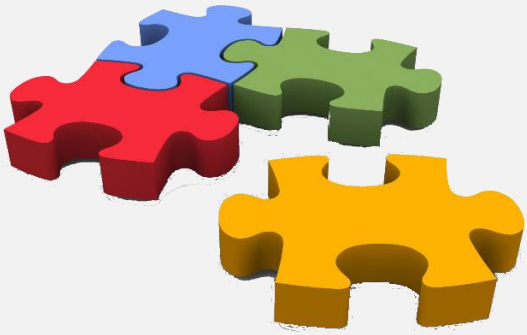
Com base nas descrições dos pesquisadores, a história do *Scratch* traz a perspectiva para compreender sua viabilidade, bem como a importância da programação nos processos de ensino e aprendizagem.

O *software Scratch* foi desenvolvido a partir de uma linguagem de programação específica, no caso o *Squeak.* Esta linguagem permitiu ao grupo *Lifelong Kindergartem Group* do laboratório do MIT, sob a liderança de Mitchel Resnick dar início ao desenvolvimento do *Scratch*.
Apresentado um *design* gráfico que teve origem no centro de computação pós escola da Intel Computer, disponibiliza uma interface gráfica visual e de mídias destinadas à programação. A construção da sintaxe de programação utilizada explora o conceito da orientação a objeto.

O nome *Scratch* vem da referência à técnica de manipulação de discos de vinil, para mixar as músicas, feita por *Disc-Jockey* no *Hip Hop* (denominada de *Scratching).* Ela está relacionada ao movimento com as mãos “para frente e para trás”. O *Scratch* possui a mesma ideia, pois mistura, diferentes estilos de mídia, como imagens, áudios, animações, fotos e músicas. Em 2003 foi iniciado seu desenvolvimento e foi lançado em 15 de maio de 2007, sendo apresentado como *software* de plataforma livre e com características específicas no *desig* gráfico, inerativo e intuitivo.

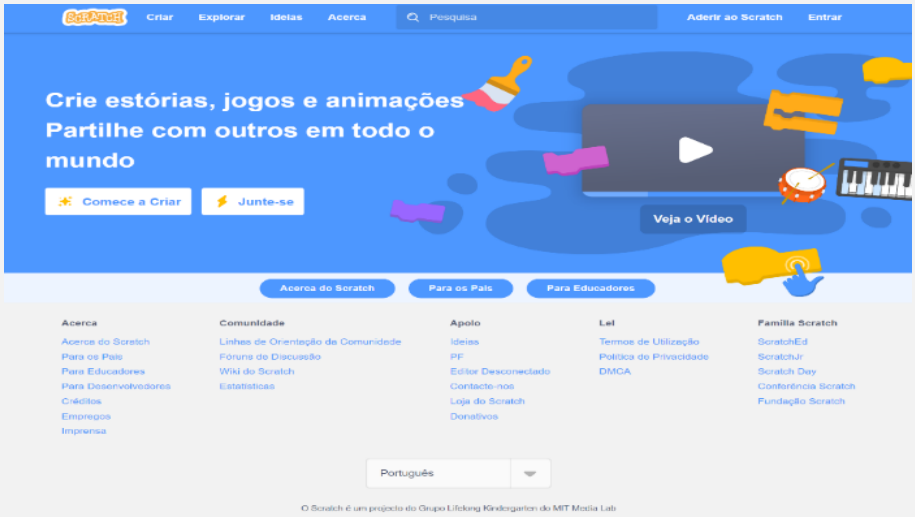
Seu início

O site *Scratch Web*, traduzido para mais 50 idiomas, apresenta alguns milhões de *downloads* do *Scratch*. Ele tem recursos de como receber e compartilhar projetos e de auxiliar os usuários. Recebe continuadamente projetos na forma de jogos, histórias interativas, animações gráficas, músicas e artes, abordando diferentes temas, assuntos e conteúdos. Existe também um modo de compartilhamento de projetos, que fazem parte da comunidade *online Scratch*. Seus membros trocam e dividem interesses na aprendizagem, possibilitando ao usuário aprender com os demais usuários. No campo denominado de "*auxiliar ao usuário*", recebe opiniões e sugestões, que tem a finalidade de desenvolver ações para o progresso do ambiente e da comunidade, com funções de tirar dúvidas e trocar experiências.



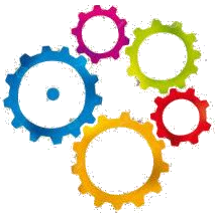
Suas versões 1.4, 2.0 e 3.0 estão disponíveis para os principais Sistemas Operacionais como *Windows*, *Linux*, *MAC OS X, MAC OS* 10.5. Existem dois tipos disponíveis no site oficial, a versão *online*, disponível a qualquer usuário com acesso à *internet,* e a versão *offline*, que apresenta um editor desconectado para a programação, denominado de *Scratch* de Secretaria, na versão 3.3. Nesta versão, os projetos desenvolvidos poderão ser carregados na versão *online* e compartilhados.

Há, também, outra versão, denominada de *Scratch Jr*. Este projeto busca atender às crianças com idade a partir de cinco anos, motivando-as a desenvolver seus próprios projetos, e beneficiando a educação pela programação A divulgação da versão *Scratch* 3.0, foi anunciada em janeiro de 2018, como uma versão pré-beta "*Preview*".



Em 1 de agosto de 2018 foi lançada uma versão *Scratch* 3.x e, em 2 de janeiro de 2019, a versão oficial do *Scratch* 3.0. Nesta versão foram apresentados novos recursos como Música, *Makey Makey*, *Text to Speech*, LEGO MINDSTORMS EV3, LEGO *Education WeDo* 2.0, BB3 e sensor de vídeo, que possibilitou a flexibilidade do uso a partir de celulares, tablet e computadores. Outro recurso foi a inclusão do conceito Arcade, que agregou funcionalidades para melhorar os jogos e animações.

Neste novo modelo, o editor de programação permite a adição de coleções de blocos, denominados como extensões. Estas extensões permitem a programação física, incluindo os *kits* de robótica, com o conceito do *bit* e LEGO, possibilitando que dispositivos reais possam ser ligados e controlados, criando a rotinas para pequenos robôs.



Vamos conhecer o *Software* de programação *Scratch* e sua linguagem

Ambiente de programação

Com interface gráfica intuitiva e interativa, o ambiente de programação do *Scratch* reúne recursos gráficos, animações, fotos, músicas e som, auxiliando no desenvolvimento de programas.

Os recursos são apresentados como ferramentas de auxílio e dão suporte a projetos, possibilitando a aprendizagem de modo construtivo. A manipulação de mídias, abrevia instruções e as estruturas algorítmicas são representadas por blocos, que permitem a reorganização do algoritmo, definindo a modelagem gráfica. Nesta reorganização os comandos computacionais da programação, *"entrada e saída de dados, tipos de dados, operadores matemáticos, lógicos e relacionais, variáveis de controle, estruturas de controle de decisão simples e composta, e laços de repetição finitos, além de arrays (vetores e matrizes)"*, comuns em qualquer outra linguagem, são facilitados.

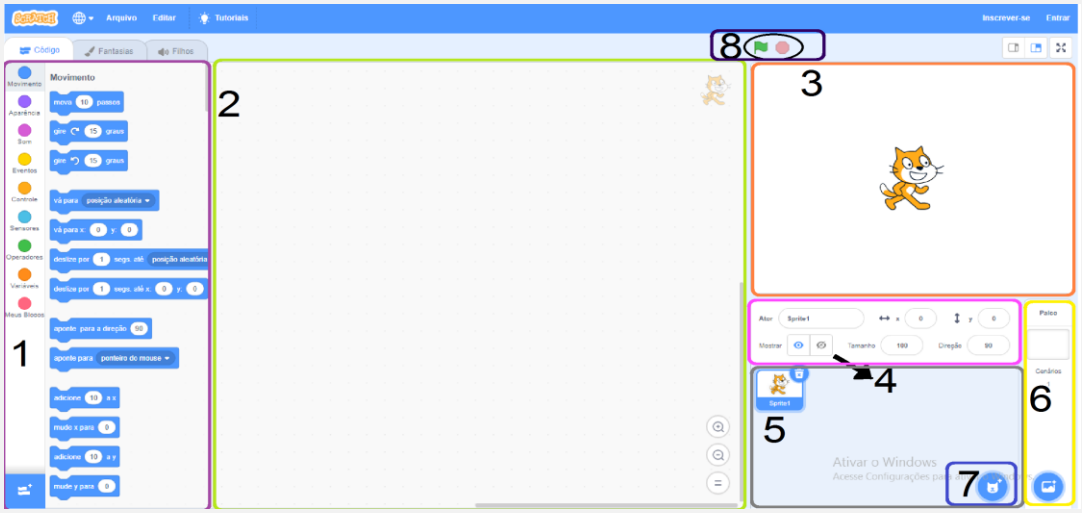
Como funciona a aba de "Código"

Formado por blocos interativos, o usuário tem a possibilidade de montar uma sequência de comandos, resultando em um projeto.

Este projeto pode na forma de animação, jogo, ou atividade, de acordo com a situação-problema proposta. Há oito possibilidades de montagem, no desenvolvimento do algoritmo, que estão representados do lado esquerdo da tela.

São estes os comandos:

- **Movimento**
- **Aparência**
- **Som**
- **Eventos**
- **Controle**
- **Sensores**
- **Operadores**
- **Variáveis**



Ao executar o *Scratch*, você verá a tela padrão, que dá a possibilidade de abrir a aba de "Código"

Áreas de trabalho: Códigos e suas funções		
1	*Armazém de Comando*	Os blocos de códigos são usados para controlar os **atores (*Sprites*)** e cenários. Todos os blocos estão representados por cores, que identificam as categorias de acordo com a paleta de blocos. Nesta paleta há 8 botões, que apresentam os comandos: Movimento, Aparência, Som, Eventos, Controle, Sensores, Operadores, Variáveis e Meus Blocos. Neste campo, também há o ícone "Adicionar extensões", que habilita novos comandos e outros blocos.
2	*Área de Recursos, ou codificação de Scripts*	Espaço definido para o processo de desenvolvimento do algorítmo. Neste espaço, são colocados os blocos de códigos, com suas devidas funções, basta arrastar os blocos da paleta de comando, para que os *Scripts* recebam suas ações.
3	*Simulador do Ecrã, ou palco*	Local para desenvolver os comandos, apresentando os resultados do processo de codificação desenvolvido na Área de recursos. Ao desenvolver um projeto, adiciona os *Sprites* com os códigos de programação. Deste modo, os *Sprites* aparecem no modo de estágio, podendo ser codificados para mover, reproduzir som, e outras possibilidades.
4	Lista de atores (*Sprites*)	Esta área, contém as configurações dos *Sprites* que participaram do projeto. Ao clicar neste campo, há a possibilidade de ativar qualquer novo *Sprite*, e seu comportamento.
5	Definição de atores	Esta área contém os *Sprites*, que podem ser trocados e reprogramados.
6	Palco	Nesta área, escolhe-se o modelo de palco.
7	Seleção de atores	Esta área contém a escolha dos novos *Sprites*.
8	Botões de iniciar e parar	Contém as funções de iniciar (*play*) e parar (pause) o programa.

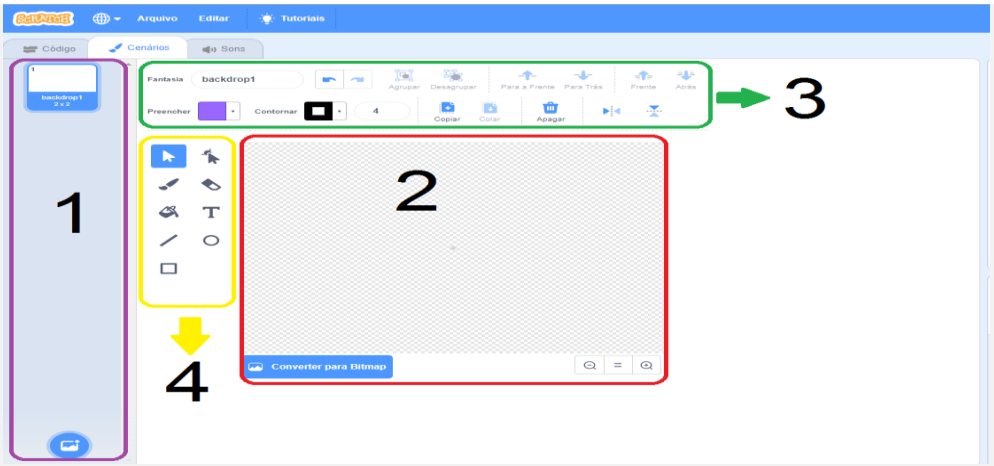
Como funciona a aba de "Fantasia"

Formado por ícones interativos, que representam o *Sprites*, o usuário tem a possibilidade de criar sua própria fantasia, e personagens, como também de escolher novos modelos.

Na aba Fantasias, pode-se modificar o(s) personagem(s) que estão sendo usado(s) no seu projeto. Essa aba pode ser encontrada no lado esquerdo da tela, ao lado da aba de Códigos.

São estes os comandos:

- **Área do Novo Traje**
- **Tela de pintura**
- **Painel de edição de fantasia**
- **Painel de pintura**



Ao executar o *Scratch*, você verá a tela padrão, que dá a possibilidade de abrir a aba de "Fantasia"

Criação de Fantasia		
1	Área do Novo Traje	Nesta área, há com escolher novos autores e fantasias, além de figurinos e adicioná-los à sua biblioteca. Existe a possibilidade de desenhar seus proprios *Sprites*, carregando uma imagem guardada na biblioteca de seu computador, seja ela uma foto ou desenho.
2	Tela de pintura	Nesta área, serão executadas as configurações e edição das fantasias. Há ferramentas a serem usadas para editar as imagens, adicionando linhas, formas, textos e cores. Em caso de erro na criação da fantasia, pode-se limpar o processo e começar de novo. Ao término, a fantasia pode ser adicionada à bilbioteca.
3	Painel de edição	Nesta área, a barra de ferramenta contém as objetos necessárias para a execução de um determinado comando.
4	Painel de pintura	Nesta área, é usada para criar ou editar trajes de fundos de palco.

Como funciona a aba de "Criação de Som"

Formada por blocos interativos, o usuário tem a possibilidade de montar uma fantasia, como também de escolher novos modelos.

Essa aba localizada ao lado da aba Fantasias, possibilita a manipulação dos efeitos sonoros do projeto. Estes efeitos podem ser adicionados e excluídos, como também permitem alterar o volume do áudio e modificá-lo.

São estes os comandos:

- Área de áudio
- Área de configuração
- Área de edição
- Painel de Controle de áudio

Ao executar o *Scratch*, você verá a tela padrão, que dá a possibilidade de abrir a aba de "Som"

Criação de Som		
1	Área de áudio	Nesta área, pode ser selecionado um áudio para o *Sprite*, ou para o palco. Pode-se montar uma biblioteca com áudios próprios. Basta gravar pelo microfone do computador. Assim, os sons do *Sprite* ou palco apareceram aqui e você tem a possibilidade de editá-los.
2	Área de configuração	Nesta área, pode-se configurar o áudio, como nomear ou mudar seu nome, além de cortar (editar) partes dele. Caso venha ocorrer algo errado, existe a possibilidade de desfazer ou refazer.
3	Área de edição	Nesta área, mostra o processo do áudio, com suas frequências, possibilitando ver o ponto exato a ser editado ou melhorado.
4	Painel de controle de áudio	Nesta área, encontram-se os botões de controle para reproduzir e modificar o áudio.

PROGRAMANDO NO SCRATCH

PROCESSO DE INICIALIZAÇÃO DA PROGRAMAÇÃO

Tudo começa com o algoritmo

Porque precisamos de algoritmos? Uma boa representação para a compreensão em relação à sua utilidade prática é a de uma receita de bolo. Seguindo-se a receita corretamente, com todos os ingredientes sendo aplicados de maneira correta, tudo dará certo, ao final. O mesmo se dá com o trabalho de programadores que escrevem o algoritmo – a receita de informática – no contexto de uma linguagem de programação específica, a fim de que o computador compreenda e execute corretamente os comandos.

Características

- O algoritmo é finito;
- Não pode estar aberto à dupla interpretação;
- Fabrica informações de saída para o mundo externo ao do ambiente do algoritmo;
- Ser efetivo no que diz respeito à execução em um tempo finito de todas as etapas existentes no algoritmo;

MODELO DE REPRESENTAÇÃO

Descrição Narrativa

Busca expressar a linguagem natural dos algoritmos, descrevendo a sequência lógica.

Exemplo 1: ***Chupar uma bala.***

- Pegar a bala;
- Retirar o papel;
- Colocar na boca a bala;
- Jogar o papel no lixo.

Exemplo 2: ***Troca de um pneu furado.***

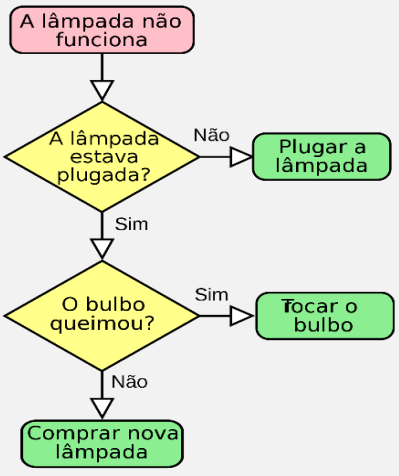
- Desparafusar as porcas;
- Suspender o carro;
- Retirar as porcas;
- Retirar o pneu furado;
- Colocar o pneu reserva;
- Apertar as porcas;
- Abaixar o carro;
- Dar o aperto final nas porcas.

Fluxograma Convencional

Modelo que os algoritmos são simbolizados.

A representação por formas geométricas difere e indica ações distintas de comandos.

Exemplo: **Trocando lâmpada**



OPERADORES DA PROGRAMAÇÃO

Na grande maioria dos problemas, é necessário que as variáveis tenham seus valores consultados ou alterados. Para isto, deve-se definir um conjunto de **OPERADORES**: Operador de atribuição, Operadores Aritméticos, Operadores Relacionais e Operadores Lógicos. Estes operadores são comandos que podem ser aplicados às expressões matemáticas, e executarem uma determinada ação ou operação, quando um programa está sendo desenvolvido.

Operador de atribuição

O operador de atribuição é o recurso algorítmico para **"Repassar"** um valor à variável.

NomeDaVariavel - Valor ou expressão detalhada.

Operadores Aritméticos: Os operadores aritméticos são conhecidos na programação como, operadores que possibilitam a realização de operações matemáticas (contas).

Símbolos	Descrição matemática
+	Adição
*	Multiplicação
-	Subtração ou inversor de sinal
/	Divisão
Quociente	Quociente da divisão de inteiros
Resto	Resto da divisão de inteiros
EXP (a,b)	Exponenciação de a elevado à b

Operadores Relacionais: São utilizados para relacionar variáveis ou expressões, resultando em valor lógico (Verdade e Falsidade).

Símbolos	Descrição matemática
=	Igual
<	Menor
>	Maior
<=	Menor igual
>=	Maior igual
<>	Diferente

Operadores Lógicos: São utilizados para avaliar expressões lógicas algorítmicas (Verdade e Falsidade).

Símbolos	Descrição matemática
E	E lógico ou conjunção
OU	OU lógico ou disjunção
NÃO	Negação

AS ESTRUTURAS PARA O USO DO *SCRATCH*

As estruturas de programação são definidas como "Funções" que realizam determinados cálculos e tarefas, descritas pelos algoritmos, e que possam ser escritos.

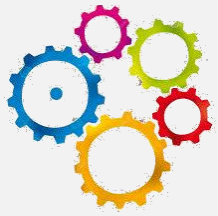
Bloco de entrada de dados

ESTRUTURA DE **ENTRADA** DE DADOS

Entrada

São os "Dados" de informações que fazem parte da inicialização do algoritmo, ou seja, uma ação do usuário, que passa a informação ao computador. O comando de entrada de dados, tem a função de solicitar dados no momento que é executado. Estes dados serão informados via teclado, ou de ação por mouse, sendo armazenados em uma variável ou em memória, para posterior processamento.

Bloco de entrada de dados



ESTRUTURA DE **PROCESSAMENTO** DE DADOS

Processamento

São os "Procedimentos" a serem executados, que se utilizam de uma sequência de informações para chegar a um resultado.

Bloco de saída de dados

ESTRUTURA DE **SAÍDA** DE DADOS

Saída

São os "Dados" que foram processados, ou seja é o comando de saída, que tem a função de mostrar ao usuário os resultados da operação. Estes dados serão mostrados no dispositivo de saída (monitor), como também podem ser impressos ou guardados no disco.

ESTRUTURA DE DECLARAÇÃO DE VARIÁVEIS

USAMOS VARIÁREIS PARA ARMAZENAR DADOS

Cada variável apresenta um tipo de dado, deste modo ao associar um dado a outro, teremos a informação. Assim, podemos manipular os dados, como também as informações, através de operações aritméticas e/ou lógicas.

Ao declarar (**criar**) uma variável, esta pode ser compreendida como um receptáculo (**caixas**) que guardam valores. Estes receptáculos devem ser representados por um "NOME", que receberá um "Valor", de um determinado tipo, sendo no modelo de textos e/ou números. Compreende-se que uma variável é a forma que se armazena um valor (**dado**), e que pode ser substituído ou trocado a qualquer momento. As variáveis facilitam a construção de expressões, porém no sentido matemático, representam símbolos que fazem parte de uma sequência de operações a serem feitas sobre determinados operandos, uma vez que visa a obtenção do resultado.

EXEMPLO: DECLARAÇÃO DE VARIÁVEIS

NO ALGORITMO

Algoritmo

```
Var
Nome: caracter;
Numero: inteiro;
Numerodois: real;
Verdade: lógico;
Falsidade: lógico;
```

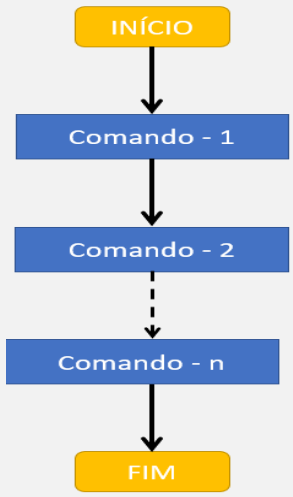
NO SCRATCH

No *Scratch* não há necessidade de fazer as declarações do tipo de variável, mas somente criá-las.

ESTRUTURA DE SINTAXE SEQUENCIAL

Os comandos serão executados de modo linear, ou seja, passo a passo, seguindo o texto escrito, de cima para baixo, sem nenhum desvio. Uma estrutura sequencial pode apresentar um ou vários comandos.

REPRESENTAÇÃO GRÁFICA DO FLUXOGRAMA DE COMANDO DA ESTRUTURA SEQUENCIAL



EXEMPLO: DE ESTRUTURA SEQUÊNCIAL DE COMANDOS

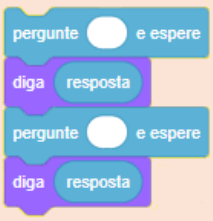
NO ALGORITMO

Algoritmo

```
Início
Escreva ('Digite seu nome');
Leia (nome);
Escreva ('Digite sua idade');
Leia (idade);
Escreva ('Seu nome é', nome);
Escreva ('e sua idade é', idade);
Fim.
```

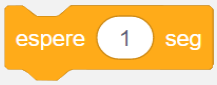
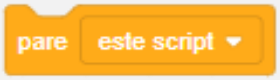
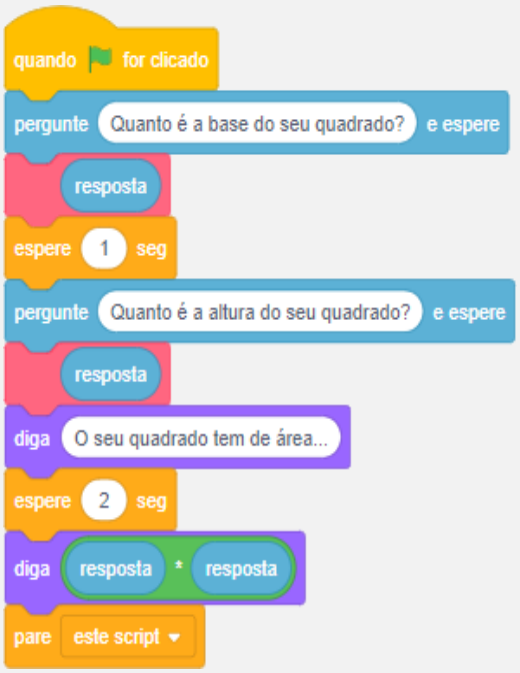
NO *SCRATCH*

Scratch



Construindo um exemplo no *Scratch*

Crie um algoritmo, em que o usuário deve digitar a base e altura do retângulo e imprimir a sua área.

Descrição: Resolução	Código final	Resposta
Clique na opção "Eventos" e mova o bloco.		
Clique na opção "Sensores" e mova o bloco para baixo do primeiro bloco.		
Clique na opção "Meus Blocos" e crie um bloco que adiciona uma entrada. Mova o bloco para baixo do segundo bloco.		
Clique na opção "Sensores" e mova o bloco para dentro do terceiro bloco.		
Clique na opção "Controle" e mova o bloco para baixo do terceiro bloco.		
Clique na opção "Sensores" e mova o bloco para baixo do quarto bloco.		
Clique na opção "Meus Blocos" e crie um bloco que adiciona uma entrada. Mova o bloco para baixo do quinto bloco.		
Clique na opção "Sensores" e mova o bloco para dentro do sexto bloco.		
Clique na opção "Aparência" e mova o bloco para baixo do sexto bloco.		
Clique na opção "Controle" e mova o bloco para baixo do sétimo bloco.		
Clique na opção "Aparência" e mova o bloco para baixo do oitavo bloco.		
Clique na opção "Operadores" e mova o bloco para dentro do nono bloco.		
Clique na opção "Sensores" e mova dois blocos para dentro dos operadores do nono bloco.		
Clique na opção "Controle" e mova o bloco para baixo do nono bloco.		
Execute o programa		

ESTRUTURA DE SINTAXE DE SELEÇÃO DE DECISÃO SIMPLES

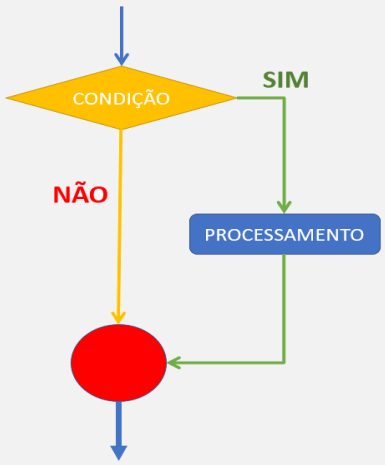
A estrutura de seleção é usada para tomar uma decisão, ou seja, procura desviar a execução do algoritmo de modo que a condição possa ser satisfeita com uma verdade ou uma falsidade. Na estrutura de decisão existem dois tipos: a "Simples" e a "Composta".

O *Scratch* apresenta os 2 (dois) tipos de instruções que envolvem os processos de seleção ou de decisão.

Estrutura de decisão Simples – utiliza o comando de sintaxe "se"

Trata-se da condição ou instrução de uma única seleção, pois seleciona ou ignora uma única ação (ou bloco de ações) caso seja "Verdadeira".

REPRESENTAÇÃO GRÁFICA DO FLUXOGRAMA DE COMANDO DA ESTRUTURA SEQUENCIAL



EXEMPLO: DE ESTRUTURA SELEÇÃO DE DECISÃO SIMPLES

NO ALGORITMO

Algoritmo

Se (condição)
Então
    Processamento
Fim-se

NO *SCRATCH*

Scratch

EXEMPLO: DA ESTRUTURA DE SELEÇÃO SIMPLES

UMA DECISÃO A SER TOMADA PELA VERDADE

Construa um algoritmo que verifique a nota do aluno. Se esta nota for maior que 59, expresse a mensagem que o aluno encontra-se "Aprovado".

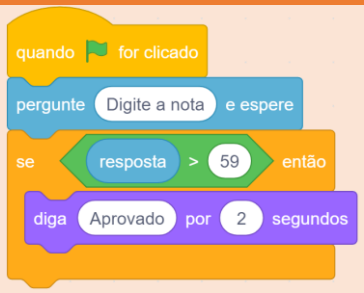
Algoritmo

PROGRAMA MEIDA;
VAR
Nota: real;
INICIO
    Leia (nota);
    Se (a nota do aluno é maior que 59)
    Então
        Imprima **("Aprovado")**;
    Fim-se
FIM.

A mesma situação-problema

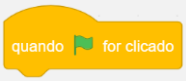
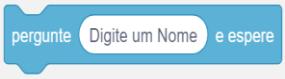
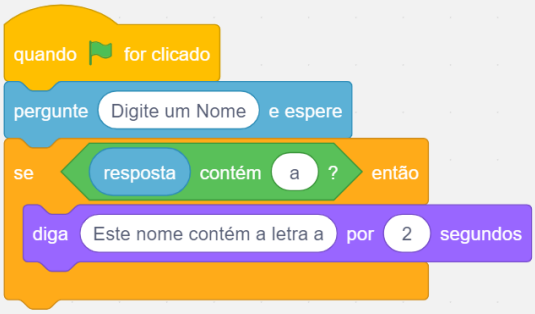
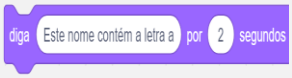
Sua resolução no *Scratch*

Scratch



Construindo um exemplo no *Scratch*

Crie um algoritmo em que o usuário deve digitar um nome qualquer, e deve ser informado se o nome digitado contém a letra A.

Descrição: Resolução	Código final	Resposta
Clique em "Eventos" e selecione o bloco "Quando for clicado" e mova o bloco.		
Clique na opção "Sensores", e selecione o bloco "Pergunte e espere", mova o bloco para baixo do primeiro bloco.		
Clique na opção "Controle", e selecione o bloco "Se - então", mova o bloco para baixo do bloco anterior.		
Clique na opção "Operadores" e selecione o bloco "contém?", e mova o bloco para dentro do bloco de "Controle".		
Clique na opção "Sensores", e selecione o bloco "Resposta", e mova o bloco para dentro do bloco de "Operadores".		
Clique na opção "Aparência", e selecione o bloco "Diga por segundos", selecione e mova o bloco para dentro do bloco de "Controle".		

Execute o programa

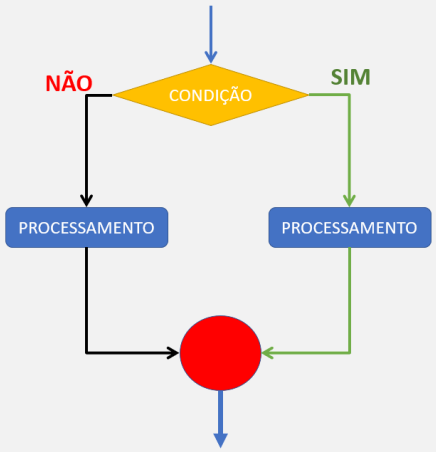
ESTRUTURA DE SINTAXE DE SELEÇÃO DE DECISÃO COMPOSTA

Estrutura de decisão Composta – utiliza o comando de sintaxe

"se .. então.. senão"

Trata-se de uma condição ou instrução de uma única seleção, pois seleciona ou ignora uma única ação (ou bloco de ações) caso seja "Verdadeira" ou "Falsa".

REPRESENTAÇÃO GRÁFICA DO FLUXOGRAMA DE COMANDO DA ESTRUTURA SEQUENCIAL



EXEMPLO: DE ESTRUTURA SELEÇÃO DE DECISÃO COMPOSTA

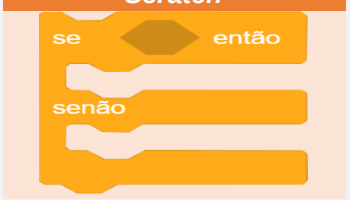
NO ALGORITMO

Algoritmo

```
Se (condição)
Então
    Processamento
Senão
    Processamento
Fim-se
```

NO *SCRATCH*

Scratch



EXEMPLO: DA ESTRUTURA DE SELEÇÃO COMPOSTA

NESTA ESTRUTURA, UMA DECISÃO PODE SER TOMADA PELA VERDADE OU PELA FALSIDADE

Construa um algoritmo que verifique a nota do aluno. Se esta nota for maior que 59, expresse a mensagem que o aluno encontra-se "Aprovado", caso a nota seja menor, expresse a mensagem que o aluno econtra-se "Reprovado".

Algoritmo

```
PROGRAMA MEIDA;
VAR
nota: real;
INICIO
  Leia (nota);
   Se (a nota do aluno é maior que 59)
   Então
        Imprima ("Aprovado")
   Senão
        Imprima ("Reprovado")
   Fim-se
FIM.
```

A mesma situação-problema

Sua resolução no *Scratch*

Scratch

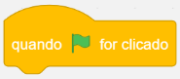
Construindo um exemplo no *Scratch*

Faça um programa de leia duas palavras, compare e imprima se são iguais ou diferentes

Descrição: Resolução

Crie duas variável, Clique em "Variáveis" "Criar uma variável" e coloque o nome delas como "palavra1" e "palavra2".

Clique em "*Eventos*" e selecione o bloco "Quando for clicado", e mova o bloco.

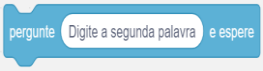


Clique na opção "*Sensores*", e selecione o bloco "Pergunte e espere", mova o bloco para baixo do primeiro bloco.

Clique na opção "*Sensores*", e selecione o bloco "Resposta", e mova o bloco para dentro do bloco de "*Operadores*".

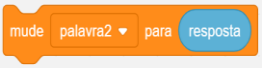
Clique na opção "*Variáveis*", e selecione o bloco "mude para", muda "palavra1" e acrescente resposta, mova o bloco para baixo do bloco anterior.

Clique na opção "*Sensores*", e selecione o bloco "Pergunte e espere", mova o bloco para baixo do bloco anterior.



Clique na opção "*Sensores*", e selecione o bloco "Resposta", e mova o bloco para dentro do bloco de "*Operadores*".

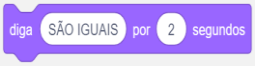
Clique na opção "*Variáveis*", e selecione o bloco "mude para", muda "palavra1" e acrescente resposta, mova o bloco para baixo do bloco anterior.



Clique na opção "*Controle*", e selecione o bloco "Se – então - senão", mova o bloco para baixo do bloco anterior.

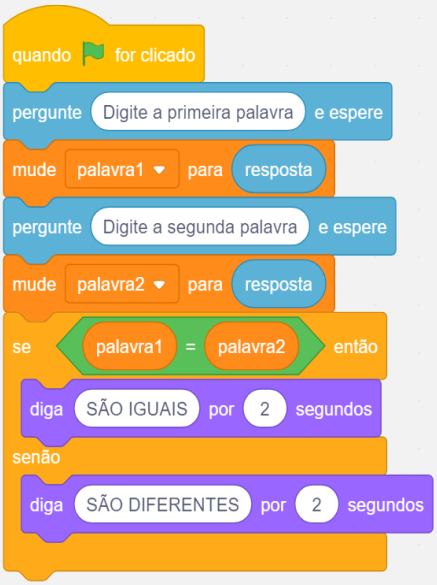
Clique na opção "*Operadores*" e selecione o bloco "igual", e mova o bloco para dentro do bloco de "*Controle*".

Clique na opção "*Aparência*", e selecione o bloco "Diga por segundos", selecione e mova o bloco para dentro do bloco de "*Controle*", na opção "então".



Clique na opção "*Aparência*", e selecione o bloco "Diga por segundos", selecione e mova o bloco para dentro do bloco de "*Controle*" na opção "senão".

Código final



Resposta

Execute o programa

ESTRUTURA DE SINTAXE DE REPETIÇÃO “REPITA OU PARA”

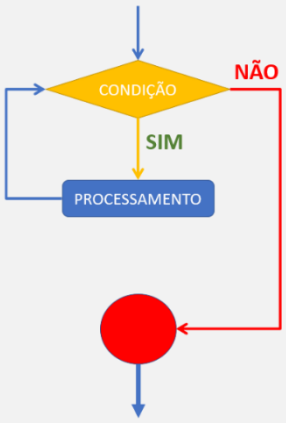
Estrutura de Repetição

A estrutura de repetição serve para efetuar um conjunto de ações que podem ser repetidas várias vezes. Neste modelo de estrutura existem três tipos básicos de repetição, que são: para, repita até, e enquanto.

Estrutura de Repetição – utiliza o comando de sintaxe “repita”

Trata-se da ação ou dos blocos de comandos que está contito no comando **“Repita”** e executa **“X”** de vezes. Sendo o **“X”** um número inteiro que representa a quantidade de vezes que será repetido, ou seja, até que atinja o **“Valor final”**.

REPRESENTAÇÃO GRÁFICA DO FLUXOGRAMA DE COMANDO DA ESTRUTURA DE REPETIÇÃO



EXEMPLO: DE ESTRUTURA REPETIÇÃO DE COMANDOS

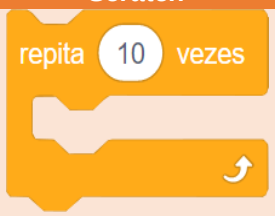
NO ALGORITMO

Algoritmo

Para i ← **início** até **fim** faça
    Processamento
Fimpara

NO *SCRATCH*

Scratch



EXEMPLO: DA ESTRUTURA DE REPETIÇÃO

NESTA ESTRUTURA, A REPETIÇÃO É O MOVIMENTO DO PERSONAGEM, ATÉ QUE SEJA APERTADA UMA TECLA

Construa um algoritmo que leia um valor qualquer. Faça o somatório deste número com 1, e repita este processo 10 vezes.

Algoritmo

```
Programa somatorio;
Var
Numero, Resposta, i: inteiro;
INICIO
    Leia (numero);
    Resposta:=numero;
    Para i:=1 até 10 faça
        Resposta:= resposta + 1;
   Fim-para
   Imprima (Resposta);
FIM.
```

A mesma situação-problema

SUA RESOLUÇÃO NO *SCRATCH*

Scratch

Construindo um exemplo no *Scratch*

Construa um algoritmo que peça para você digitar 10 números. Ao final deve avisar que terminou.

Descrição: Resolução

Clique em "*Eventos*" e selecione o bloco "Quando for clicado" e mova o bloco.

Clique na opção "*Controle*", e selecione o bloco "repita vezes", mova o bloco para baixo do bloco anterior.

Clique na opção "*Sensores*", e selecione o bloco "Pergunte e espere", mova o bloco para baixo do segundo bloco.

Clique na opção "*Sensores*", e selecione o bloco "Resposta", e mova o bloco para dentro do bloco de "*Aparência*".

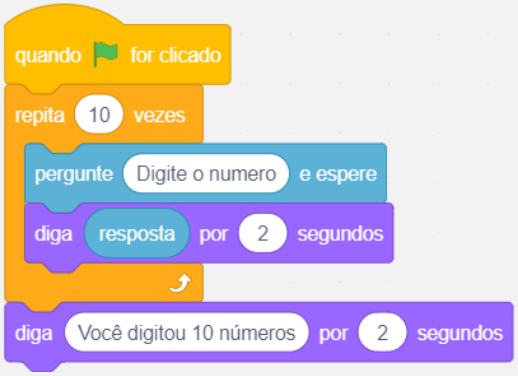
Clique na opção "*Aparência*", e selecione o bloco "Diga por segundos", selecione e mova o bloco para dentro do bloco de "*Controle*".

Clique na opção "*Aparência*", e selecione o bloco "Diga por segundos", selecione e mova o bloco para dentro do bloco de "*Controle*", para baixo do bloco anterior.

Execute o programa

Código final

Resposta

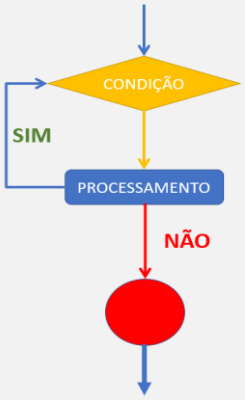


ESTRUTURA DE SINTAXE DE REPETIÇÃO “REPITA ATÉ”

Estrutura de Repetição – utiliza o comando de sintaxe “Repita até”

Trata-se da ação ou dos blocos de comandos contidos no comando “Repita até”, que será executada até que a condição de parada seja “Verdadeira”. Para que a Verdade aconteça, deve-se elaborar uma sintaxe de comandos semelhante ao “Repita até”.

REPRESENTAÇÃO GRÁFICA DO FLUXOGRAMA DE COMANDO DA ESTRUTURA DE REPETIÇÃO



EXEMPLO: DE ESTRUTURA REPETIÇÃO DE COMANDOS

NO ALGORITMO

Algoritmo

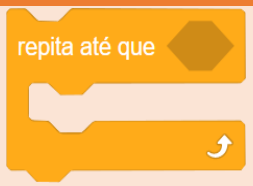
Repita

Processamento (sequência de comandos)

Até (Condição Lógica)

NO *SCRATCH*

Scratch



EXEMPLO: ESTRUTURA DE REPETIÇÃO “REPITA ATÉ”

NESTA ESTRUTURA, A REPETIÇÃO OCORRE ATÉ ENCONTRAR O DISPOSITIVO DE SAÍDA

Faça um programa que some um número e seu concecutivo. O programa termina quando for apertada a tecla de espaço. Obs: a tecla de espaço tem valor 32 na tabela ASCII.

Algoritmo

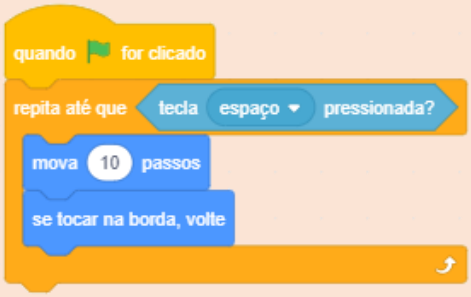
```
Programa tecla;
Var
   contador,soma, tecla: inteiro;
INICIO
   contador:=1;
   Repita até (contador <>1)
      soma:=soma+1;
      se (tecla=’32’)
         então
            a:=0;
      fim-se;
  fim-repita;
  imprima (soma);
FIM.
```

Situação-problema

Faça um programa no qual o personagem *Scratch* movimente-se pela tela. O programa terminada quando for apertada a tecla de espaço.

Sua resolução no *Scratch*

Scratch



Construindo um exemplo no *Scratch*

Construa um programa que solicite para digitar um número. Este deve fazer a soma de cada número lido. O programa termina quando for digitado "999". Apresente o resultado da soma.

Descrição: Resolução

Clique em "*Eventos*" e selecione o bloco "Quando for clicado", e mova o bloco.

Clique na opção "*Sensores*", e selecione o bloco "Pergunte e espere", mova o bloco para baixo do bloco anterior.

Clique na opção "*Variáveis*", e selecione o bloco "mude para", muda a variável para "soma" e mova o bloco para baixo do bloco anterior.

Clique na opção "*Controle*", e selecione o bloco "repita até que", mova o bloco para baixo do bloco anterior.

Clique na opção "*Operadores*" e selecione o bloco "igual", e mova o bloco para dentro do bloco de "*Controle*".

Clique na opção "*Sensores*", e selecione o bloco "Resposta", e mova o bloco para dentro do bloco de "*Operadores*".

Clique na opção "*Variáveis*", e selecione o bloco "mude para", mude a variável para "soma" e mova o bloco para dentro do bloco de "Controle".

Clique na opção "*Aparência*", e selecione o bloco "Diga por segundos", acrescente a variável "soma" e mova o bloco para dentro do bloco de "*Controle*".

Clique na opção "*Sensores*", e selecione o bloco "Pergunte e espere", mova o bloco para baixo do bloco "Aparência".

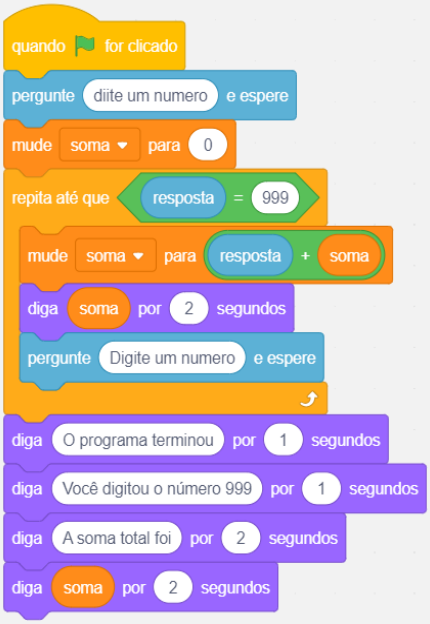
Clique na opção "*Aparência*", e selecione o bloco "Diga por segundos", acrescente mensagem e mova o bloco, para baixo do bloco de "*Controle*". Repita 3 vezes o processo.

Clique na opção "*Aparência*", e selecione o bloco "Diga por segundos", acrescente a variável "soma" e mova o bloco para baixo do anterior.

Execute o programa

Código final

Resposta

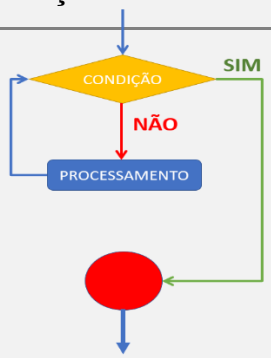


ESTRUTURA DE SINTAXE DE REPETIÇÃO "REPITA ATÉ"

Estrutura de Repetição – utiliza o comando de sintaxe "Enquanto"

Trata-se da ação ou dos blocos de comandos, contida no comando "Sempre", que será executado infinitamente. Esta estrutura repete uma sequência de comandos "Enquanto" uma determinada condição (expressão lógica) for verdadeira ou falsa, ou satisfaça a sentença. Para que a "Verdade" aconteça, deve-se elaborar uma sintaxe de comandos semelhantes ao enquanto.

REPRESENTAÇÃO GRÁFICA DO FLUXO DE COMANDO DA ESTRUTURA DE REPETIÇÃO



EXEMPLO DE ESTRUTURA REPETIÇÃO "ENQUANTO"

NO ALGORITMO

Algoritmo

Enquanto (condição) faça

Processamento

Fim-enquanto;

NO *SCRATCH*

Scratch

CUIDADO COM O LOOP "INFINITO"

O QUE É LOOP INFINITO"

O processo de repetição que gera o "Loop infinito" é atribuído a um erro de lógica na programação. Este problema pode travar o programa, sistema, até mesmo o aparelho. Neste sentido, isto representa um "ERRO" na execução de um determinado programa, ou seja, este passa a executar repetidamente a mesma sequência de instruções. Por isso deve-se tomar "*muito cuidado*" quando utiliza-se este modelo de laço.

EXEMPLO DA ESTRUTURA DE REPIÇÃO "ENQUANTO"

No Algoritmo

Algoritmo

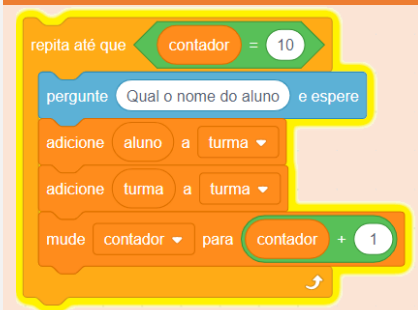
Enquanto (condição) faça

Processamento

Fim-enquanto;

NO *SCRATCH*

Scratch



EXEMPLO: DA ESTRUTURA DE REPETIÇÃO "SEMPRE"

NESTA ESTRUTURA A REPETIÇÃO OCORRE ATÉ VALIDAR A SAÍDA

Faça um programa que leia um número e some a ele mesmo. O programa termina quando o valor for maior que 50.

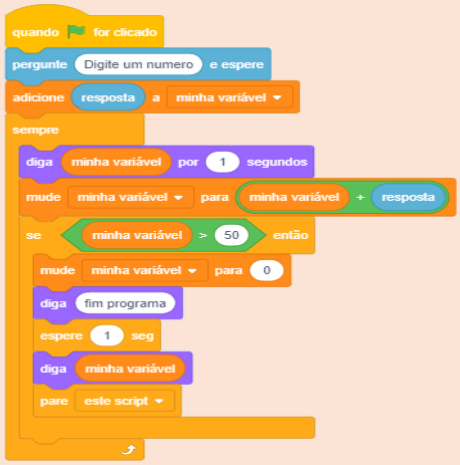
Algoritmo

```
Programa maiorque50;
Var
Num, maior: inteiro;
INICIO
    Leia (num)
    Enquanto (num < 50) faça
        num:= num + num;
    Fim-enquanto.
    Impirma (num);
FIM.
```

A mesma situação-problema

Sua resolução no *Scratch*

Scratch



Construindo um exemplo no *Scratch*

Construa um algoritmo que leia qualquer número. O programa termina quando for digitado o número 50.

Descrição: Resolução

Clique em "*Eventos*" e selecione o bloco "Quando for clicado" e mova o bloco.

Clique na opção "*Controle*", e selecione o bloco "Sempre", mova o bloco para baixo do bloco anterior.

Clique na opção "*Sensores*", e selecione o bloco "Pergunte e espere", mova o bloco para baixo do bloco anterior.

Clique na opção "*Controle*", e selecione o bloco "Se - então", mova o bloco para baixo do bloco anterior.

Clique na opção "*Operadores*" e selecione o bloco "Igual", e mova o bloco para dentro do bloco de "*Controle*".

Clique na opção "*Sensores*", e selecione o bloco "Resposta", e mova o bloco para dentro do bloco de "*Operadores*".

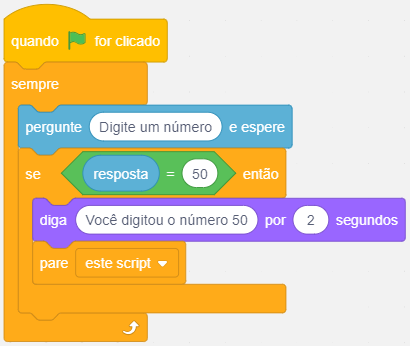
Clique na opção "*Aparência*", e selecione o bloco "Diga por segundos", acrescente texto mova o bloco para baixo do anterior.

Clique na opção "*Aparência*", e selecione o bloco "Pare", mova o bloco para baixo do anterior.

Execute o programa

Código final

Resposta



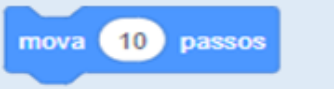
Classificados

PALETA DE BLOCOS DE COMANDOS

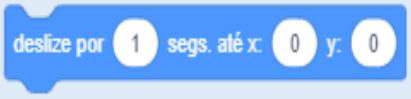
BLOCO DE MOVIMENTO

Os blocos de comandos de "Movimento" servem para criar movimento aos personagens escolhidos. Os personagens da animação podem se movimentar para algum lugar usando coordenadas (que podem ser encontradas abaixo da animação) ou usando indicadores, como partes do cenário ou outros personagens.

Os códigos de movimento são usados para configurar movimentos no programa. Você pode utilizar esses comandos para fazer o personagem se mover para os lados, fazer giros, fazer uma sequência de movimentos, apontar para certa direção.



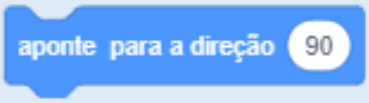
Move a quantidade especificada de passos;



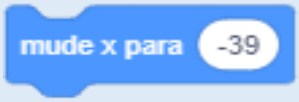
Desliza pela quantidade de segundos especificados escolhidos até 'X': e 'Y';

Adiciona um valor desejado a 'X';

Vira para a quantidade de graus especifiado para a direita;

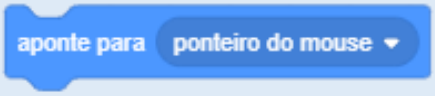


Apontar para uma direção específica;



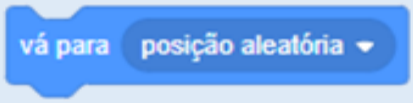
Muda o valor de 'X' para o valor especificado;

Vira para a quantidade de graus especifiado para a esquerda;



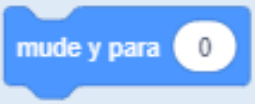
Aponta para o ponteiro do mouse;

Adiciona um valor desejado a 'Y';

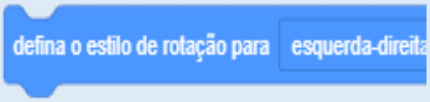


Movimenta o *Sprite* para uma posição aleatória no palco;

Vai para uma posição específica de 'X' e 'Y';

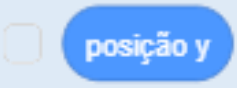


Muda o valor de 'Y' para o valor especificado;



Define o estilo da rotação para uma tecla desejada;

Indica o número de 'X';



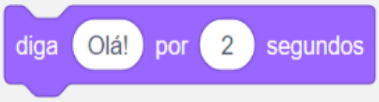
Indica o número de 'Y';

Indica a direção do objeto;

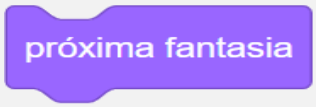
BLOCO DE APARÊNCIA

Os códigos de "Aparência" do personagem servem para mudança de cenário, fantasia, tamanho, efeito, mostrar/esconder personagem, pensamento do personagem e, inclusive fazê-lo falar. Esses códigos mexem e mudam o cenário, mostram falas, pensamentos e tamanhos de personagens.

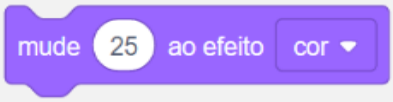
Os códigos de aparência são usados para configurar as atividades do *Sprite* no momento da programação. Estas atividades podem demonstrar as possibilidades de ações que o ator pode ter. Estas ações modelam o processo de interatividade entre personagens e deles com o usuário.



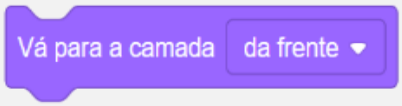
Quando acionado, o personagem diz "Olá" por 2 segundos;



Quando acionado, o personagem vai para a próxima fantasia;

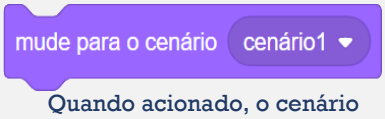


Quando acionado, o personagem muda para a cor do número especificado;

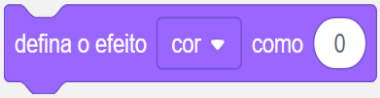


Quando acionado, o personagem selecionado vai para a camada da frente;

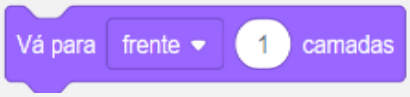
Quando acionado, o personagem diz "Olá";



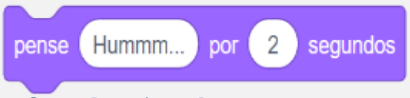
Quando acionado, o cenário muda para outro cenário específico;



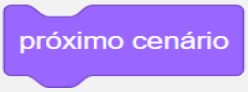
Quando acionado, o personagem muda para a cor como a cor do número;



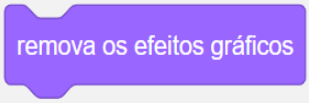
Quando acionado, o personagem selecionado avança 1 camada para frente;



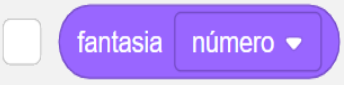
Quando acionado, o personagem pensa "Hummm..." por 2 segundos;



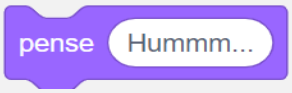
Quando acionado, pula para o próximo cenário definido;



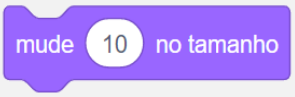
Quando acionado, os efeitos gráficos são removidos;



Habilita a fantasia para o valor ou nome especificado;



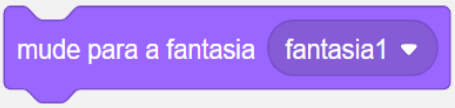
Quando acionado, o personagem pensa "Hummm...";



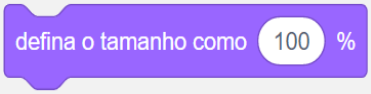
Quando acionado, o personagem aumenta seu tamanho de acordo com o valor em % especificado;

Quando acionado, mostra o personagem selecionado;

Habilita o cenário para o valor ou nome especificado;

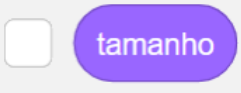


Quando acionado, o personagem muda seu visual para o outro escolhido;



Quando acionado, o personagem cresce no tamanho específico;

Quando acionado, esconde o personagem selecionado;

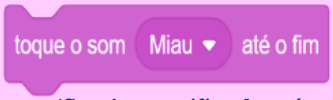


Mostra o valor do tamanho do *Sprite* (esse valor é definido em porcentagem);

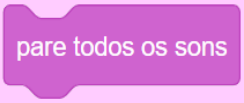
BLOCO DE SOM

Os códigos de "Som" apresentam a parte sonora do aplicativo, são úteis na criação de animações e jogos. Esses códigos servem para acrescentar sons de personagem e "barulhos" no cenário, possibilitando desenvolver processos de interação com o usuário.

Os códigos de "Som" são usados para colocar áudios no programa. Você pode utilizar os comandos para fazer o personagem tocar algum tipo de som. Estes blocos podem ser usados sozinhos ou agrupados com outros comandos.

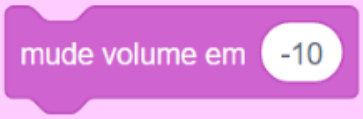


Toca o 'Som' especificado até o fim;

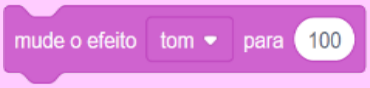


Para todos sons;

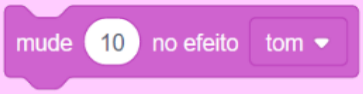
Toca o som especificado



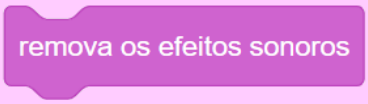
Muda o 'Volume' em número especificado;



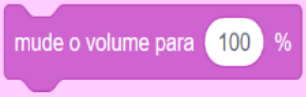
Muda o efeito do tom para número especificado;



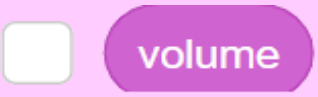
Muda número especificado no efeito tom;



Remover todos efeitos sonoros;



Mudar 'Volume' para número especificado;



Mostrar o volume;

BLOCO DE EVENTOS

Os códigos de "Eventos" são blocos que determinam para o *Sprite* desenvolver as ações que foram programadas. Para executar estas ações, ou a definição das chaves de execução, basta clicar no bloco do evento inicial de atividades para que seja posta em atividade a lógica programada.

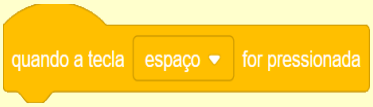
Esses códigos são usados para dar início ao algoritmo desenvolvido, de acordo com a possível situação-problema, como também transmitir informações entre os personagens e o usuário.

Quando acionado, os blocos abaixo desta sessão será executado;

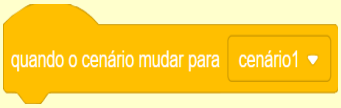
Quando clicar no ator, os blocos abaixo desta sessão será executados;

Quando ruído for maior que 10 ou outro valor, os blocos abaixo desta sessão será executado;

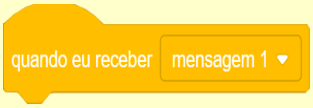
Quando acionado, transmite determinada mensagem;



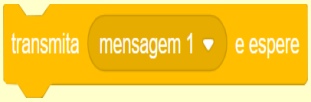
Quando acionado, os blocos abaixo desta sessão será executado;



Quando o cenário mudar para o outro cenário escolhido, os blocos abaixo esta sessão será executados;



Quando receber "Mensagem 1", os blocos abaixo desta sessão será executados;



Quando acionado, transmite determinada mensagem e fica esperando uma ação;

BLOCO DE CONTROLE

Os códigos de "Controle" servem para aplicar condições ao personagem desejado, como a condição (SE), (senão), (sempre), (repita), (pare), (apague), (crie), (quando), (repita) e (espere). Nesses códigos pode ser encontrada a estrutura caso, mais conhecida como Estrutura 'SE', que pode ser usada para criar situações-problemas, estruturas de repetição, entre outros.

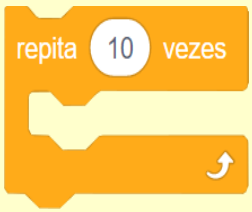
Neste conjunto de blocos são definidos os comandos que controlam ou manipulam os *Sprite*, Cenário, Fantasia, Som e outros. Deste modo, os comandos pré-definidos são responsáveis por fazer as operações entre outros blocos de comandos.

Quando acionado, espera a quantidade de segundos solicitados;

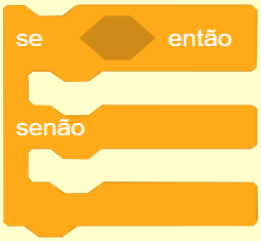
Quando acionado, caso a variável identificada neste bloco for 'Verdadeira', os blocos abaixo serão executados;

Quando acionado, repete a ação que encontra-se neste bloco até que a variável seja executada;

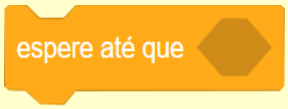
Quando acionado, os comandos inseridos dentro deste bloco será repetido, até que o botão de parar seja pressionado;



Quando acionado, repete a quantidade solicitada dos blocos que se encontram dentro dele. Os comandos inseridos no bloco será repetido conforme o número de vezes solicitado, e então seguirá executando os blocos de comandos de forma linear;

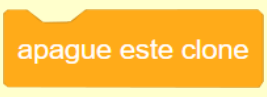


Quando acionado, caso a variável identificada neste bloco for 'Verdadeira', os blocos abaixo serão executados, caso contrário será executada a opção abaixo dessa;

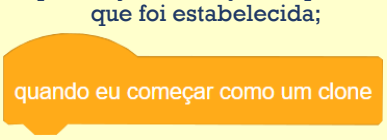


Quando acionado, espera a ação ser realizada, pois os comandos inseridos dentro deste bloco vai se repetir até que atinja a condição de parada que foi estabelecida;

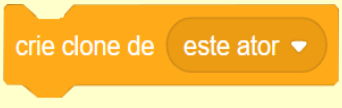
Quando acionado, para a ação solicitada;



Quando acionado, apaga o clone solicitado;

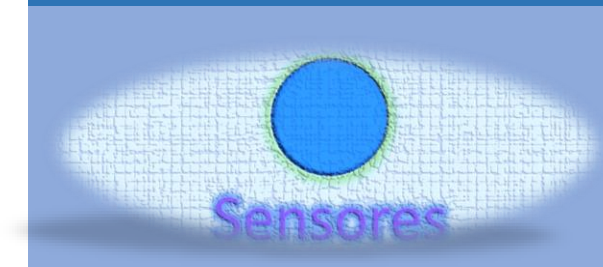


Quando começar com um clone, a ação abaixo deste bloco será realizada;



Quando acionado, cria um clone do personagem;

BLOCO DE SENSORES



Os códigos de "Sensores" são usados para configurar os movimentos do *Sprite* no programa. Utiliza-se esses comandos para fazer o personagem movimentar para os lados, fazer giros, fazer uma sequência de movimentos, apontar para uma certa direção, e nesses movimentos há a possibilidade de mudar de cor, distância, definir posições. Porém, estes comandos, são necessários serem combinados com outros comandos e blocos.

Esta categoria tem como principal objetivo desenvolver o processo de interação entre o *Sprite* e o usuário.

Mostra se é verdadeiro ou falso;

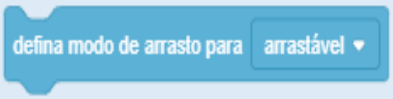
Mostra a resposta no canto superior esquerdo na tela;

Mostra a posição 'Y' do mouse na tela;

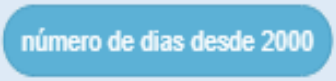
Mostrar qualquer tipo de datas como ano, mês, dia e segundos;

Seleciona a cor específica e mostra se está clicando nela;

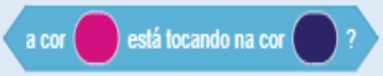
Mostra se uma tecla especifica está pressionada;



Definir se é arrastável ou não;



Mostrar números de dias desde 2000;



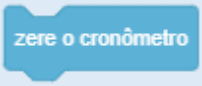
Se cor 'X' estiver tocando cor 'Y' mostrará;

Mostrar se o mouse está pressionado;

Ativar ou desativar ruído;

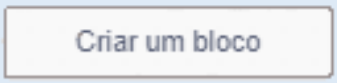
Mostrar nome de usuário;

Mostra a distância até o ponteiro do mouse;

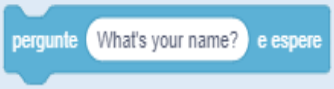


Zerar o cronômetro;

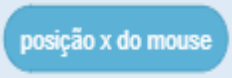
Ativar ou desativar cronômetro;



Permite ao usuário criar seu próprio bloco de sensores;



Permite fazer uma pergunta especifica e esperar a resposta;



Mostrar a posição 'X' do mouse na tela;

Mostrar informações gerais de palco;

BLOCOS DE OPERADORES

Os códigos de "Operadores" geralmente são usados junto com a parte de Controle, para indicar uma condição. Estes blocos têm a função de mostrar os operadores matemáticos lógicos e relacionais, como também fazer operações entre outros blocos e comandos. Estas combinações permitem que cálculos possam serem feitos nos programas.

Outras ações dos operadores lógicos são também usadas em estruturas, principalmente as condicionais de "se então", "se então..senão", "repita", "repita até", "sempre". Outros recursos destes operadores estão em englobar alguns elementos como variáveis e blocos de movimento como os de posição "X" e "Y".

Faz a soma dos dois números escolhidos;

Colocar dois números e comparar se o 1° é maior que o 2°;

Condição 'NÃO', é usada para comparar se é verdadeira ou falso ;

Trás o resto de uma divisão;

Faz a subtração dos dois números escolhidos;

Colocar dois números e comparar se o 1° é menor que o 2°;

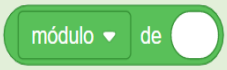
Juntar duas palavras, ou concatenar as palavras;

Arredondar os números com vírgula;

Faz a multiplicação dos dois números escolhidos;

Comparar se dois números são iguais;

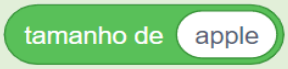
Mostrar o a posição que uma letra encontra-se;



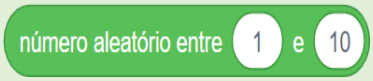
Opções gerais para modificar número específico;

Faz a divisão dos dois números escolhidos;

Condição "E", serve para comparar se é verdadeiro ou falso;



Mostrar quantas letras contém uma palavra especifica;



Escolher dois números e mostra um aleatório entre eles;

Condição "OU", serve para comparar se é verdadeiro ou falso;

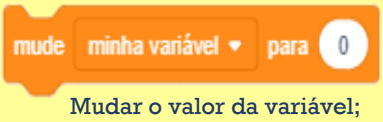
Mostrar se uma palavra ou frase contém uma letra específica;

BLOCO DE VARIÁVEIS

Local em que se pode declarar as variáveis, para quando for mexer com algoritmos. As variáveis do programa são muito úteis para criação de recursos de armazenamento de valores. Ao declarar a variável, estamos definindo e reservando um espaço da memória para armazenar um valor que será usado pelo programa.

Uma variável sempre armazena apenas um único valor. Em um processo de atribuição, o valor da variável sustitui o antigo.

Opção para criar uma variável;



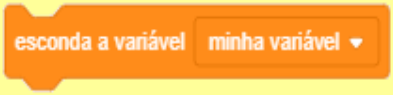
Mudar o valor da variável;

Mostrar a variável;

Modela um "VETOR" de Lista, que podem ser guardadas informações;

Ativar a variável;

Adicionar valor da variável;



Esconder a variável

BLOCO DE VARIÁVEIS E RECURSOS EXTRAS

Ativar e mostrar lista no canto superior esquerdo;

Inserir item desejado em uma posição desejada;

Adicionar texto e/ou números a lista;

Substituir item desejado por outro;

Verificar se a lista contém item especificado;

Apagar item selecionado da lista;

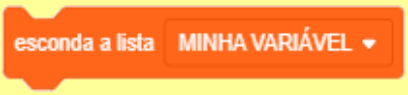
Procurar por algo específico na lista;

Mostrar o tamanho da lista;

Apagar todos itens da lista;

Acrescentar um item digitado ou solicitado à variável criada;

Mostrar a lista de variáveis criada no palco, para monitoramento dos seus valores;



Esconder a lista;

Possibilita criar seus próprios blocos;

BLOCOS MEUS BLOCOS

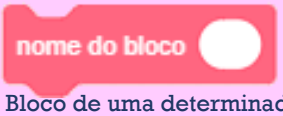
O objeto "Meus Blocos" serve para criar um bloco dentro do aplicativo. A parte "Vazia" serve para a criação de qualquer comando que pode ser feita pelo usuário, utilizando outros blocos. Há mais um código, chamado "Meus Blocos", em que se pode criar seus próprios blocos de comando.

Esta categoria é dependente do usuário (programador), que deve usar seu conhecimento de lógica na construção de seus próprios blocos, com objetivo de facilitar o desenvolvimento do algoritmo.

Permite criar seu próprio bloco de ação;

Bloco de uma determinada ação definida pelo usuário;

Bloco de uma determinada ação de operação, ou controle definida pelo usuário;



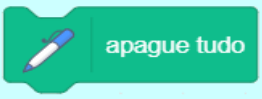
Bloco de uma determinada ação, operação, controle, e atividade definida pelo usuário;

Bloco de uma determinada ação definida pelo usuário, no formato de caixa texto;

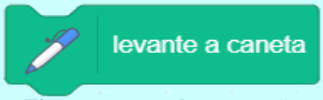
BLOCOS DE LÁPIS

Os códigos de "Lápis" são usados para configurar movimentos do ator no programa. Você pode utilizar esses comandos para fazer o personagem se mover para os lados, fazer giros, fazer uma sequência de movimentos, apontar para certa direção, desenhando suas ações.

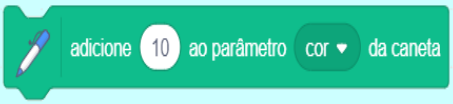
A ferramenta de edição, o lápis, permite fazer alterações em determinadas ações dos *Sprites*;



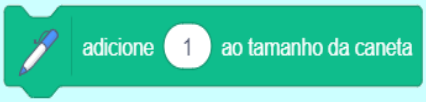
Apaga todos os traços e carimbos feitos no palco;



Tira a caneta do ator, não permitindo que seja desenhado no palco;

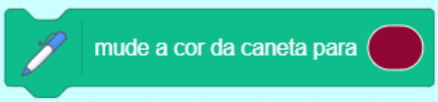


Muda a cor da caneta, para o valor especificado;



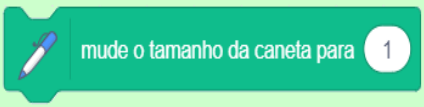
Muda a espessura do traço da caneta, de acordo com o valor especificado;

Carimba uma imagem do ator no palco;

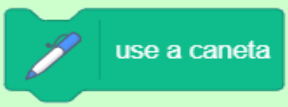


Muda a cor da caneta, para a cor especificada;

Muda o parâmetro de cor da caneta.



Muda a espessura do traço da caneta, de acordo com o valor especificado;



Faz um risco quando o ator movimenta-se no Palco;

NOTÍCIA: A NECESSIDADE DA LÓGICA MATEMÁTICA

INTRODUÇÃO À LÓGICA DE PROGRAMAÇÃO

PROGRAMA

É a forma de representar um algoritmo escrito em uma linguagem computacional.

LINGUAGEM DE PROGRAMAÇÃO

Há softwares que permitem o desenvolvimento de algoritmos. Seus resultados são **programas** como jogos, editores, sistemas empresarias, sites, até mesmo um sistema operacional.

DESENVOLVIMENTO DE PROGRAMA

Para o desenvolvimento de programas, necessita-se do processo de criatividade, que apresente soluções que possam ser não somente eficazes, mas também eficientes. Assim, supostos problemas que criamos, envolvem a lógica, que podem ser resolvidos e representados pela programação. O início do processo da construção do programa, é o aprendizado da **lógica de programação**, que permite a associação direta com o desenvolvimento do raciocínio matemático, levando à interpretação do problema.

LÓGICA E O RACIOCÍNIO MATEMÁTICO

Adicionar e subtrair valores, muitas crianças aprendem facilmente. Porém, as dificuldades matemáticas começam aparecer quando elas se deparam com situações--problemas, que exigem a identificação de quais operações devem ser usadas, e que tipo de ações podem envolver mostrando o mesmo resultado.

Deste modo, necessita-se desenvolver o raciocínio matemático, para que possam ser construídas as habilidades matemáticas, sendo estas facilitadoras na busca por possíveis soluções. Assim, a construção de programas tem a necessidade do desenvolvimento e construção destas habilidades.

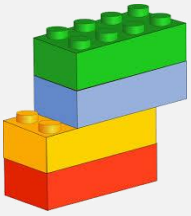
"SEQUÊNCIA LÓGICA"

Usada para executar um comando passo a passo, em que todos os elementos serão executados. Uma sequência pode possuir um ou vários comandos (formas, modelos, ações) que produzem a solução para um problema.

O PROCESSO DE SEQUÊNCIA – USANDO OS BLOCOS DO *SCRATCH*

BLOCOS

Para resolver uma situação-problema, você deve encaixar os blocos de acordo com sua funcionalidade, arrastando o bloco desejado para a área de programação. Este movimento lembra o encaixe dos blocos LEGO, ou seja, arrastar e encaixar o bloco independente de cor.



Notícia Extra – Os blocos ilustram

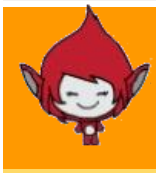
Ao escolher um dos blocos, eles ilustram a **sintaxe algoritmica,** que molda um conjunto de determinadas instruções. Porém, estas instruções podem resultar em ações diferenciadas ao *Sprite*.

GAZETINHA

DA LÓGICA MATEMÁTICA E PROGRAMAÇÃO

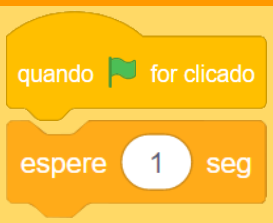
DESENVOLVIMENTO DA LÓGICA

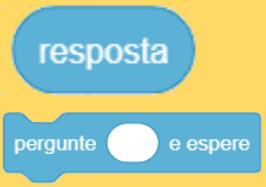
VAMOS CONHECER OS PERSONAGENS



VAMOS PROGRAMAR NO SCRATCH

Você acabou de ver nossa apresentação, então queremos saber quem é você. Construa seu primeiro programa no *Scratch*. Peça para o *Scratch*, perguntar "Qual seu nome?". Você irá usar somente os blocos de entrada de dados (Pergunte), e de saída (Resposta). Para resolver utilize as seguintes blocos de programação.





ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR UMA SEQUÊNCIA DE PERGUNTAS E RESPOSTAS?

VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

NESTA ATIVIDADE

Para resolver a situação--problema, você deve seguir a direção das setas, riscando as linhas. Ao final perceberá que um desenho será formado no tabuleiro. Tente descobrir o segredo escondido.

Movimente para cima

Movimente para baixo

Movimente para a direita

Movimente para a esquerda

Movimenta para diagonal direita abaixo

Movimenta para diagonal esquerda abaixo

Movimenta para diagonal esquerda acima

Movimenta para diagonal direita acima

Notícia Extra: *Retas, Segmentos de retas e semiretas*

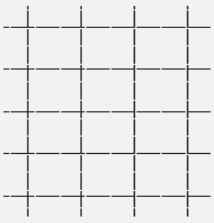
Reta: formada por infinitos pontos, alinhados, de modo ilimitado nos dois sentidos, e representada por uma letra miniscula. Possui 3 posições: horizontal, vertical ou inclinada.

Segmento de reta: limitado por 2 pontos distintos da reta.

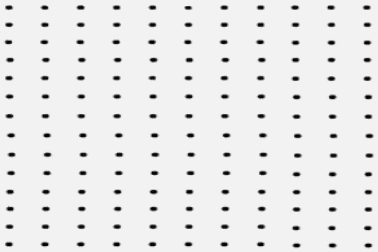
Semireta: possui origem, mas é ilimitada, ou seja, possui início, mas não tem fim.

ATIVIDADE: *Tabuleiro do segredo. Siga as setas e descubra*

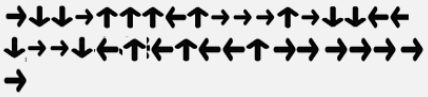
SIGA AS SETAS

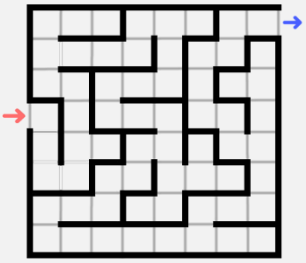


SIGA AS SETAS



SIGA AS SETAS

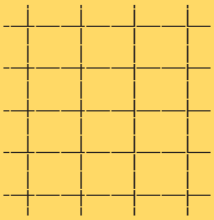




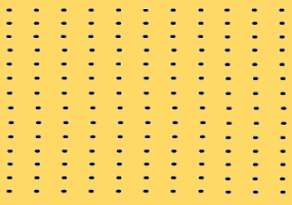
VAMOS CONSTRUIR!

Agora construa seu próprio tabuleiro de segredo.

DESENHE AS SETAS DE SEU SEGREDO AQUI



DESENHE AS SETAS DE SEU SEGREDO AQUI

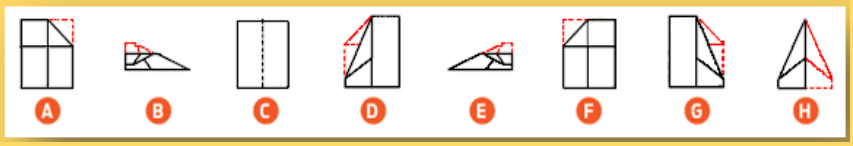


VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

NESTA ATIVIDADE

Para resolver a situação-problema, você deve observar a sequência de desenhos. Mostre quais são as possibilidades para construirmos o avião de papel.

Etapas para confecção do avião de papel



Pergunta: *Qual é a sequência de letras para construção do avião de papel*

Resposta Algoritmo 1: .....

Resposta Algoritmo 2: .....

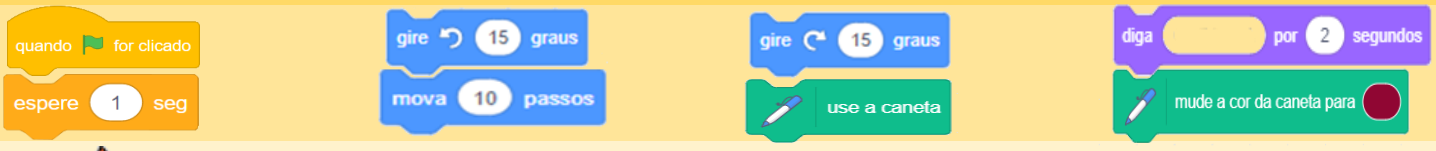
Há outra possibilidade?.....Se sim, qual? .....

VAMOS PROGRAMAR NO SCRATCH

Construa o segredo das setas que gera o desenhe de um número. Para resolver utilize os seguintes blocos de programação.

Lembre-se: os blocos podem se repetir, e há várias possibilidades de resolver.

Agora, construa seu segredo no *Scratch*. Para resolver utilize as seguintes blocos de programação.



Lembre-se: os blocos podem se repetir, e há várias possibilidades de resolver.

ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR ESTES DESENHOS COM OS MESMOS BLOCOS?

NOTÍCIA: AS SEQUÊNCIAS

SEQUÊNCIA NUMÉRICA

SUCESSÕES OU SEQUÊNCIAS

Sucessões ou sequências é uma listagem de elementos ou termos de um conjunto qualquer que estão dispostos em certa ordem, permitindo-se identificar o primeiro termo.
Exemplo:

- O conjunto (0, 1, 2, 3, 4, 5,...) é chamado sequência ou sucessão dos números.

OBSERVAR A REPRESENTAÇÃO

Ao lado serão apresentadas várias sucessões numéricas obedecendo a certa lógica. Observe a sucessão, encontre a regra que direciona a sua construção, e gere o próximo elemento.

REPRESENTAÇÃO

1, 3, 5, 7, ___

2, 7, 12, 17, 22, 27, ____

0, 1, 4, 9, 16, 25, 36, ____

0, 4, 16, 36, 64, ____

1, 1, 2, 3, 5, 8, ____

VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

NESTA ATIVIDADE

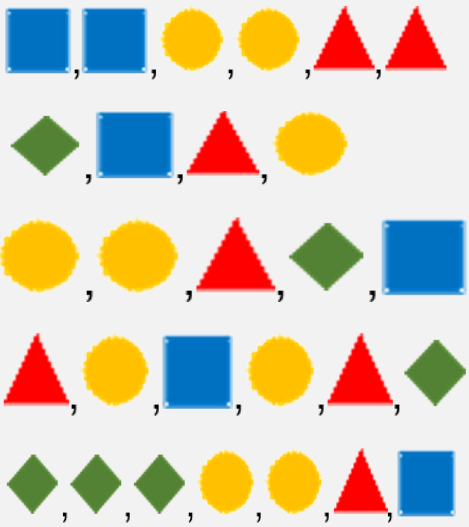
Você deve observar a sequência lógica. Esta sequência é formada por figuras geométricas planas, números e cores.

Figuras Planas:

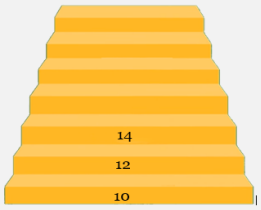
Notícia Extra: *Figuras Planas*

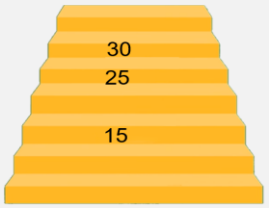
Região plana limitada por segmentos de retas. Para formar uma figura é preciso no mínimo 3 segmentos de reta.

ATIVIDADE: *Qual a sequência das figuras? Desenhe ao lado.*



ATIVIDADE: *Observe a figura, e preencha os degraus com os valores que faltam.*



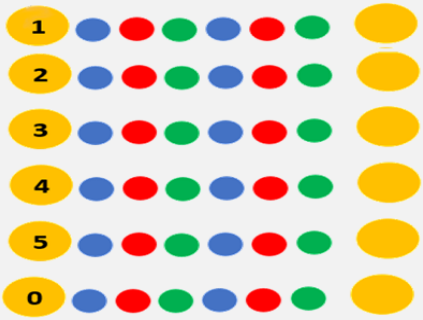


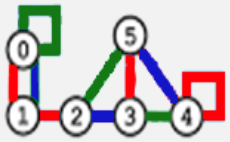
ATIVIDADE: *Observe a sequência de figuras, e desenhe a próxima.*

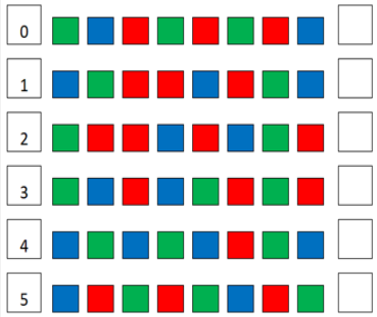
Desenhe a próxima sequência:

Desenhe a próxima sequência:

ATIVIDADE: *Observe a figura, siga o caminho das cores e escreva em que número parou.*





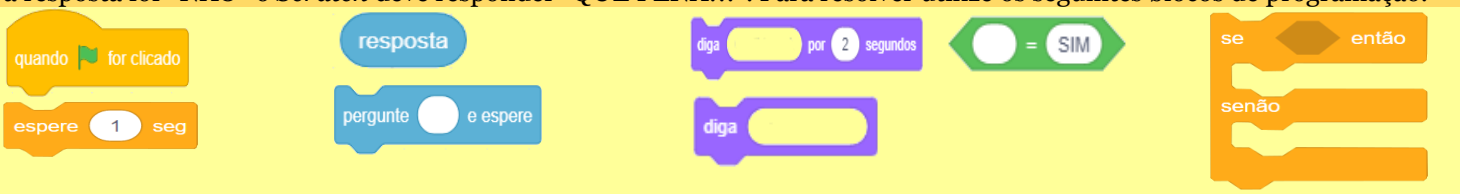


VAMOS CONSTRUIR!
Agora construa sua própria sequência lógica.

Letras **Números** **Figuras Geométricas**

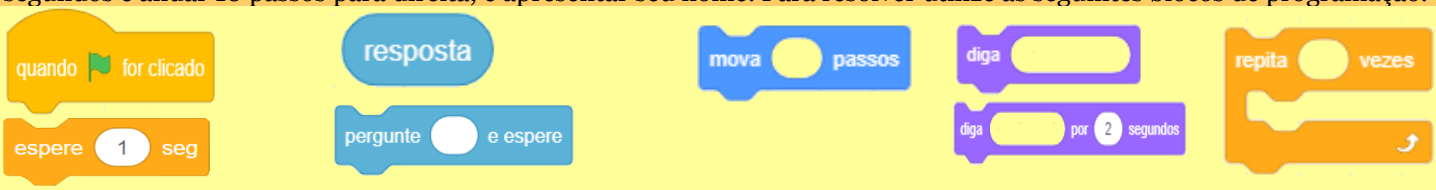
VAMOS PROGRAMAR NO SCRATCH

Construa a sequência no *Scratch*. Nesta sequência você vai pedir para o *Scratch* perguntar "Vamos construir uma sequência de numero?" Se a resposta for "SIM" você deve digitar uma sequência de números (exemplo: 1 – 2 – 3 – 4..), se a resposta for "NÃO" o *Scratch* deve responder "QUE PENA!!!". Para resolver utilize os seguintes blocos de programação.



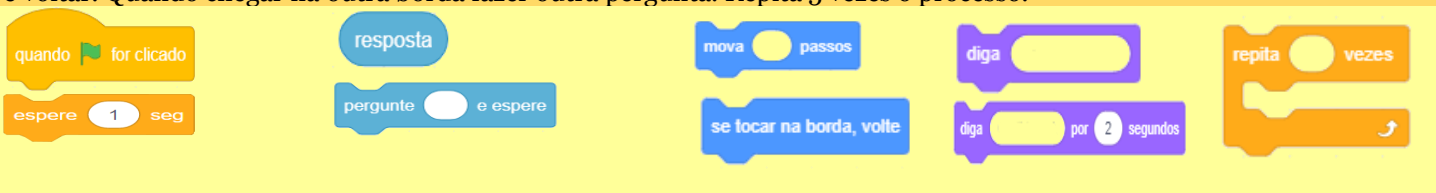
Lembre-se: os blocos podem se repetir, e há várias possibilidades de resolver.

Construa a sequência no *Scratch*. Nesta sequência você vai pedir para o *Scratch* dizer "Seu nome é?". Ele deve esperar 3 segundos e andar 10 passos para direita, e apresentar seu nome. Para resolver utilize as seguintes blocos de programação.



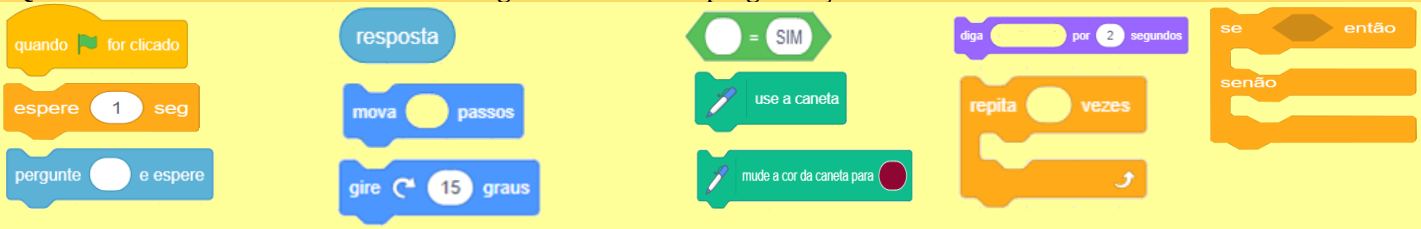
Lembre-se: os blocos podem se repetir, e há várias possibilidades de resolver.

Construa a sequência no *Scratch*. Nesta sequência você vai fazer o *Scratch* andar até a borda, fazer uma pergunta qualquer e voltar. Quando chegar na outra borda fazer outra pergunta. Repita 5 vezes o processo.



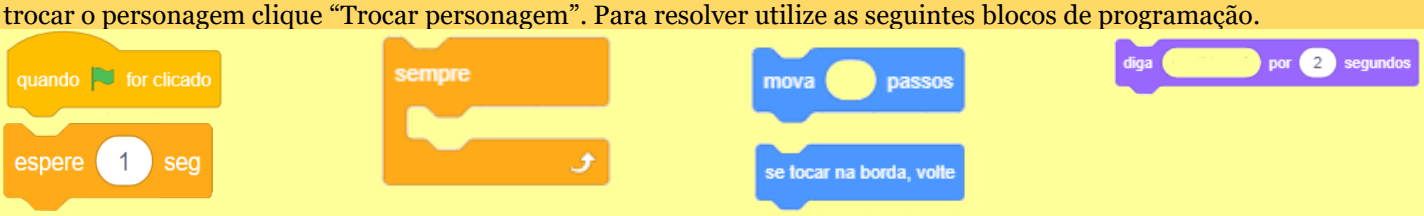
Lembre-se: os blocos podem se repetir, e há várias possibilidades de resolver.

Construa a sequência no *Scratch*. Pergunte ao *Scratch* se que ele quer desenhar uma sequência de linhas. Se ele responder "SIM", faça ele desenhar 4 linhas, que no final formem um quadrado. Se a resposta for "NÃO", o *Scratch* deve responder "QUE PENA!!!". Para resolver utilize as seguintes blocos de programação.



Lembre-se: os blocos podem se repetir, e há várias possibilidades de resolver.

Construa uma sequência no *Scratch*. Nesta sequência você vai fazer um peixinho nadar de uma lado para outro. Para trocar o personagem clique "Trocar personagem". Para resolver utilize as seguintes blocos de programação.



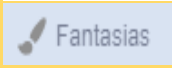
Lembre-se: os blocos podem se repetir, e há várias possibilidades de resolver.

Novidade:

Para trocar o personagem clique no ícone.

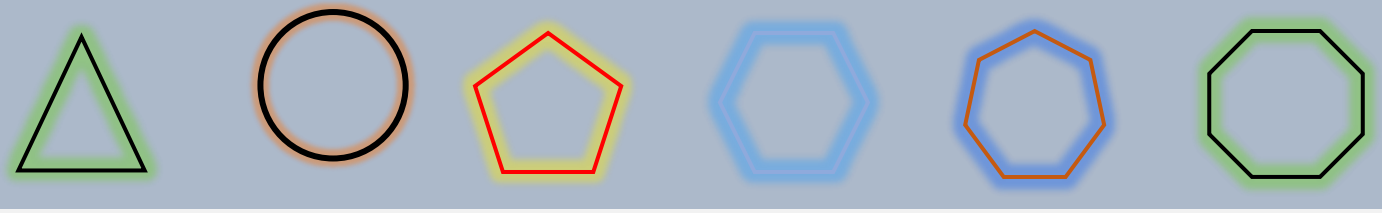
Para mudar de direção clique no ícone.

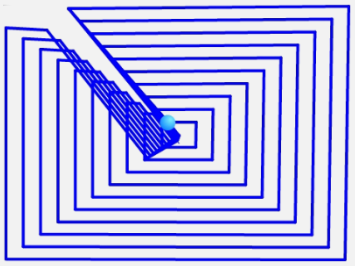
Ative a direção para direita e esquerda.



Na aba fantasia você pode trocar o cenário, para o fundo do mar.

ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR ESTES DESENHOS USANDO OS MESMOS BLOCOS?





NOTÍCIA: MAIOR, MENOR, IGUAL

SINAIS DE RELAÇÃO

SÃO DIVIDOS EM TRÊS

Os sinais de relação são 3, e estão divididos em = (igual a), > (maior que), < (menor que). Estes sinais estabelecem o processo de relação de grandezas entre dois valores.

Exemplo:

•.1=1; 3 > 2; 10 < 11

Formas para não esquecer

Primeira Forma

Consiste em identificar o lado maior do sinal (com duas pontas) sempre fica do lado do maior número. Deste modo os sinais de '**maior que**' e '**menor que**' são sempre aqueles que saem da direção do maior número para o menor.

Segunda Forma - Macete

Consiste em passar um traço vertical na parte de baixo do sinal. Este traço fará os símbolos lembrarem o número 4 e 7. Assim 4 e menor que sete, portanto, < significa '**menor que**', e > significa '**maior que**'.

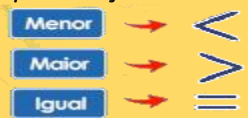
VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

NESTA ATIVIDADE

Para resolver a situação-problema, você deve identificar qual é o desenho maior, menor ou igual.

Notícia Extra: *Símbolos do Maior e Menor*

Representação Simbólica.



ATIVIDADE: *Tabelinha do segredo: Escreva qual das figura é maior e qual é a menor?*

Situação	Qual é a figura maior	Situação	Qual é a cor da figura menor

ATIVIDADE: *Complete a tabela com símbolos de: maior ">", menor "<" e igual "=".*

Nº	Símbolo	Nº	Símbolo	Nº
5		2		8
3		6		3
10		10		23
25		43		45

NOTÍCIA: AS ORDENS CRESCENTE E DECRESCENTE

ORDEM CRESCENTE

SUCESSÕES DO MENOR PARA MAIOR

Ordem Crescente: É a representação dos números, organizados do menor valor numérico para o maior valor numérico.

ORDEM DECRESCENTE

SUCESSÕES DO MAIOR PARA MENOR

Ordem Decrescente: É a representação dos números, organizados do maior valor numérico para o menor valor numérico.

VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

NESTA ATIVIDADE

Para resolver a situação-problema, você deve identificar qual é o maior e o menor valor numérico, colocando em ordem, conforme o solicitado.

Notícia Extra: Sucesso e Antecessor

***Sucessor**: Todo número natural possui um sucessor, que nada mais é do que o número que vem "**depois**" dele. Exemplo: sucesso de 10 é o 11.*

***Antecessor**: Todo número natural maior que zero possui um antecessor, que nada mais é do que o número que vem "**antes**" dele. Exemplo: o antecessor de 9 é 8.*

ATIVIDADE: *Tabelinha do segredo: Vamos resolver as ordens?*

OBSERVE AS CARTAS?

Organize os números do menor para o maior:

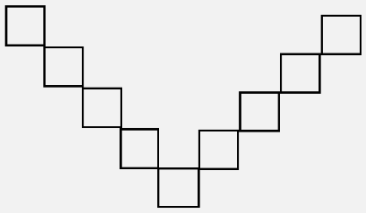
PINTE NA ORDEM CRESCENTE E DAS CORES, SOMENTE AQUELES DA DIAGONAL

1	14	15	4
12	7	6	9
8	11	10	5
13	2	3	16

COLOQUE EM ORDEM

Escreva nos quadrados:

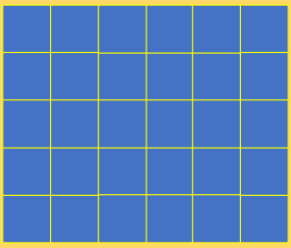
- os seguintes números: "19 – 36 – 3 – 45 – 20", na ordem decrescente;
- os seguintes números: "18 – 37 – 65 – 40 - 95", na ordem crescente;



VAMOS CONSTRUIR!

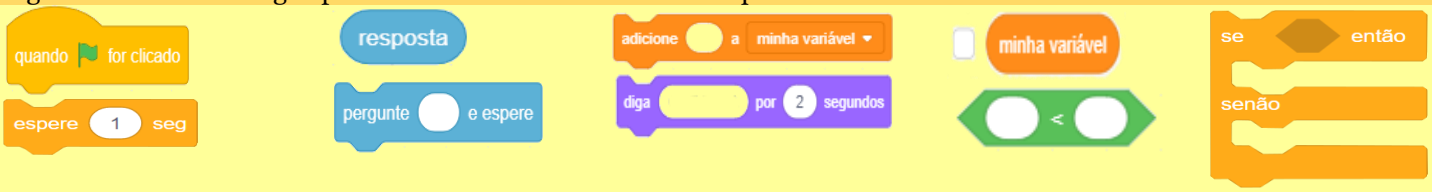
Agora você deve construir seu labirinto. Desenhe barreiras no tabuleiro e mostre uma sequência de números em uma ordem, mostrando a saída.

Sequência de números



VAMOS PROGRAMAR NO SCRATCH

Neste programa você vai construir duas variáveis de nome: numero1 e numero2. Peça para o *Scratch* pedir para você digitar seus valores. Logo após o *Scratch* deve mostrar e dizer quem é o menor.



Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa você vai fazer a leitura de uma sequência de 10 números. Ao final da leitura o *Scratch* deve dizer quem é o maior número. Para resolver utilize as seguintes blocos de programação.

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR ESTES PROGRAMAS USANDO OS BLOCOS QUE JÁ FORAM APRESENTADOS?

Neste programa você deve ler um número qualquer e dizer o seu antecessor e seu sucessor.

Neste programa você deve criar duas variáveis de nome NumeroA e NumeroB, nessa ordem. O *Scratch* deve apresentar a você a ordem inversa. Exemplo: se os dados lidos forem 5 e 9, devem ser impressos na ordem 9 e 5.

Neste programa você deve criar 5 variáveis que receberão os valores escritos por você. O *Scratch* deve apresentar uma das ordens (crescente ou decrescente).

NOTÍCIA: AS OPERAÇÕES

AS QUATRO OPERAÇÕES BÁSICAS

ADIÇÃO

É uma das **operações** básicas da aritmética, que combina dois ou mais números em um único número. Esta combinação é denominada de soma, total ou resultado.

Exemplo:
- 1 + 2 = 3

MULTIPLICAÇÃO

É uma das **operações** matemáticas que indica a evolução da adição, ou seja, a adição sucessiva de um mesmo número, que possui a mesma quantidade de elementos.

Exemplo:
- 3 x 2 = 6

SUBTRAÇÃO

É uma das **operações** matemáticas que indica quanto é um valor numérico é subtraído do outro, ou seja, uma certa quantidade pode ser retirada de outra, e o valor que sobra é o resultado desta operação.

Exemplo:
- 5 – 2 = 3

DIVISÃO

É uma das **operações** matemáticas, inversa da multiplicação. A operação de divisão consiste em dividir dois números, um maior e outro menor em partes iguais.

Exemplo:
- 6 ÷ 2 = 3

VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

NESTA ATIVIDADE

Para resolver a situação-problema, você deve descobrir qual o número que está faltando, que dê o resultado do centro do círculo de cálculo.

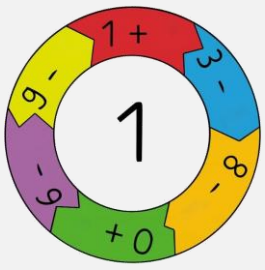
Pergunta: Qual o número de devo preencher para que o resultado das operações seja igual ao valor que esta no centro da figura.

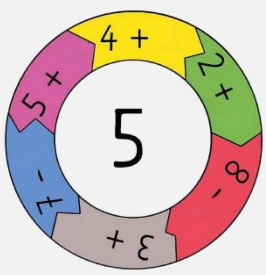
Notícia Extra: Pares e Ímpares

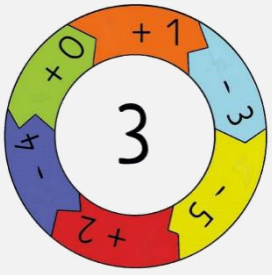
***Números pares**: é um subconjunto dos números inteiros. Em uma definição formal, o número par pode ser compeendido como "múltiplo de 2 iniciado em Zero". Outra forma de identificar o número par, é dada pela divisão por 2 e que seu resto é igual a 'o' (zero).*

***Números ímpares**: também é um subconjunto dos números inteiros. Em uma definição formal, o número impar, pode ser compreendido como "todo número que não é múltiplo de 2". Outra forma de identificar o número impar é dada pela divisão por 2 e que seu resto é igual a 1 (um).*

ATIVIDADE: *Círculo do segredo: Qual é o número a ser preenchido?*

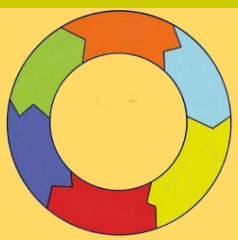
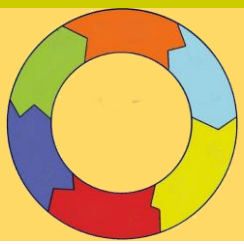






VAMOS CONSTRUIR!

Agora você deve construir seu próprio circulo de segredo, com as operações que envolvam adição e subtração.


VAMOS CONSTRUIR!

Agora você deve construir sua própria sequência de valores numéricos. Pense em valores aleatórios ímpares e regitre. Faça também para os números pares.

Números ímpares **Números pares**

VAMOS PROGRAMAR NO SCRATCH

Neste programa você vai construir duas variáveis de nome: numero1 e numero2. Logo após deve fazer o *Scratch* dizer qual é o resultado da soma dos dois números.

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa você vai construir duas variáveis de nome: numero1 e numero2. Logo após deve fazer o *Scratch* dizer qual o resultado da subtração dos dois números.

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR O PROGRAMA USANDO OS MESMOS BLOCOS?

Você deve fazer um programa que leia 3 valores numéricos. Identifique o maior valor numérico e guarde em uma variável. Para os dois valores restantes, faça a soma deles, compare os valores, do maior e da soma. Faça a subtração do maior pelo menor, e apresente o resultado.

Você deve fazer um programa que leia 10 nomes. O *Scratch* deve apresentar o primeiro, quinto e o décimo, ou apresente outras possibilidades.

VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

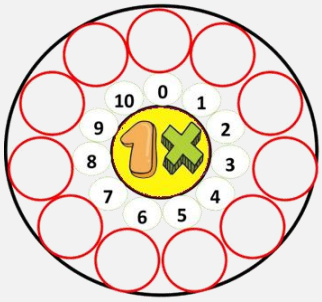
NESTA ATIVIDADE

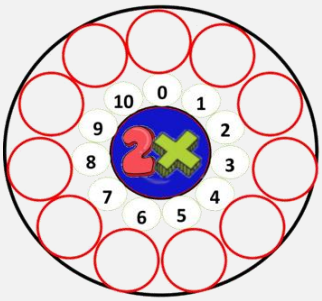
Para resolver a situação-problema, você deve colocar um número no centro do circulo, e fazer sua multiplicação. Logo após escreva seu resultado.

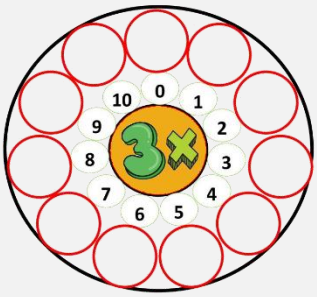
Notícia Extra: Números Primos

***Números Primos**: São os números naturais que possuem apenas dois divisores, o número 1 e ele mesmo.*

ATIVIDADE: *Círculo do segredo: Qual é o resultado da multiplicação? Preencha.*







VAMOS CONSTRUIR!

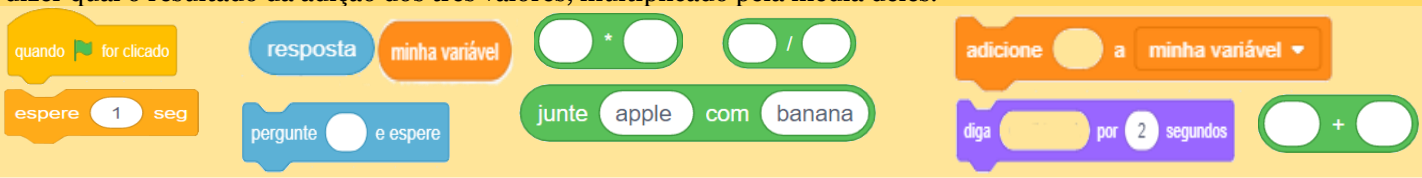
Agora você deve construir seu próprio círculo de segredo, com a operação de multiplicação.

VAMOS PROGRAMAR NO SCRATCH

Neste programa você vai construir duas variáveis de nome: numero1 e numero2. Logo após deve fazer o *Scratch* perguntar qual o valor do numero1 e numero2. O *Scracth* deve dizer qual o resultado da multiplicação dos dois números.

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa você vai construir três variáveis de nome: numero1, numero2 e número3. Logo após deve fazer o *Scratch* dizer qual o resultado da adição dos três valores, multiplicado pela média deles.



Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa você vai construir três variáveis de nome: numero1, numero2 e número3. Logo após deve fazer o *Scratch* dizer qual o resultado da multiplicação do primeiro número pelo terceiro número, subtrair a resposta pelo segundo número.

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa você deve pedir que o usuário digite um numero qualquer. Se o número for igual a 1 e menor que 10 deve imprimir seu nome. Se for maior que 10 e menor que 100 deve desenhar um quadrado ou uma outra figura plana qualquer. Se for diferente diga a mensagem "Vamos recomeçar".

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR O PROGRAMA USANDO OS BLOCOS QUE VOCÊ JÁCONHECE?

Você deve fazer um programa que solicite um número que deve estar entre 5 e 10. Se não estiver peça ao *Scratch* para dizer que o número não se encontra, e você deve digitar novamente o número, até que seja validada a resposta.

Você deve fazer um programa que solicite dois números. O *Scratch* deve dizer o somatório dos valores que encontram-se no intervalo destes números. Observação: o primeiro número deve ser menor que o segundo.

Você deve fazer um programa que solicite um número. O *Scratch* deve informar o somatório de seus 5 sucessores, e a multiplicação de seus 5 antecessores.

Você deve construir um programa que solicite a entrada de dois números inteiros e calcule a potência do primeiro número pelo segundo (x elevado a y). O *Scratch* deve informar o resultado.

Você deve contruir um programa que leia 3 valores menores que 100. Faça a soma e com o resultado o *Scratch* deve desenhar um triângulo.

Vamos construir a situação-problema? Um hotel cobra R$ 60,00 a diária e mais uma taxa de serviços. A taxa de serviços é de: • R$ 5,50 por diária, se o número de diárias for maior que 15; • R$ 6,00 por diária, se o número de diárias for igual a 15; • R$ 8,00 por diária, se o número de diárias for menor que 15. Construa um algoritmo que mostre o nome e o total da conta de um cliente.

VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

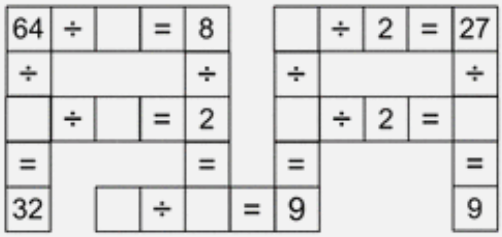
NESTA ATIVIDADE

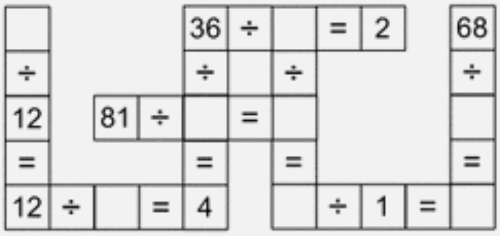
Nesta atividade você deve descobrir o número que completa a cruzadinha. Para isto observe bem os valores de resposta.

Notícia Extra: *A Fração*

***Fração**: é a representação de uma ou mais partes de algo que foi dividido em partes iguais. Desta forma, a fração é uma divisão onde o dividendo é o numerador, e o divisor é o denominador. Assim, uma fração é um número racional.*

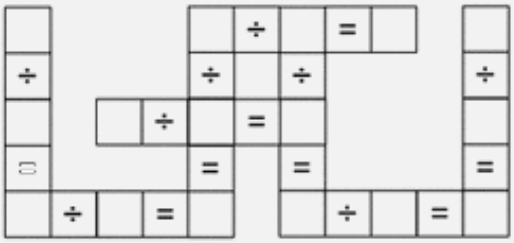
ATIVIDADE: *Cruzadinha da divisão: Encontre os valores.*

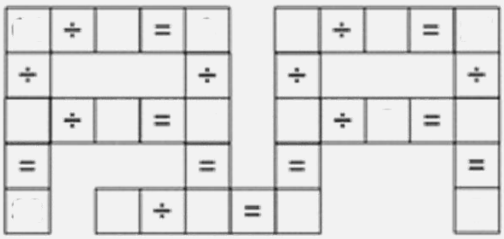




VAMOS CONSTRUIR!

Agora você deve construir sua cruzadinha, com a operação de divisão.



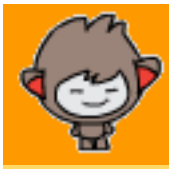


VAMOS PROGRAMAR NO SCRATCH

Neste programa você vai construir duas variáveis de nome: numero1 e numero2. Logo após deve fazer o *Scratch* dizer qual o resultado da divisão do primeiro número pelo segundo.

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR AS OPERAÇÕES QUE VOCÊ MODELOU NA CRUZADINHA DA DIVISÃO?



VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

NESTA ATIVIDADE

Você deve completar a tabelinha com os sinais de adição, subtração, multiplicação e divisão. Observe bem os valores de resposta de cada linha e coluna.

Notícia Extra: *Área e Perímetro*

Área e Perímetro *são utilizados para indicar as medidas de alguma figura.*

Área: *é a medida de uma superfície de uma figura geométrica.*

Perímetro*: é a soma dos segmentos de retas que formam a figura, chamados de lados.*

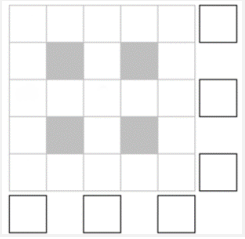
ATIVIDADE: *Cruzadinha das operações: Complete com as operações (+ - x ÷).*

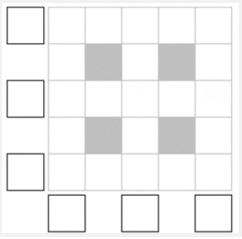
2		9		4	38
7		3		6	27
1		5		8	14
15		-2		-44	

2		5		9	-2
7		4		3	25
6		8		1	-2
8		1		3	

VAMOS CONSTRUIR!

Agora você deve construir sua própria tabelinha das operações.





VAMOS PROGRAMAR NO SCRATCH

Neste programa você vai construir duas variáveis de nome: numero1 e numero2. Logo após deve fazer o *Scratch* dizer qual o resultado da multiplicação do primeiro número pelo segundo, e a divisão da soma destes números por dois.

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa: Um feirante comprou 385 tomates, 233 cebolas e 120 cenouras. Já vendeu 230 tomates, 123 cebolas e 65 cenouras para um único cliente. Ele pretende colocar todos os produtos comprados em 5 sacolas. Quantos produtos devem ir em cada sacola?

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa: Carlinhos comprou 17 pacotes de figurinhas. Cada pocote contém 6 figurinhas. Carlinhos resolveu colar a metade do total de suas figurinhas compradas no seu álbum. Com quantas figurinhas ele ficou?

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Este programa é uma situação problema: Uma sorveteria vende três tipos de picolés. Sabendo-se que o picolé do tipo 1 é vendido por R$ 0,50, o do tipo 2 por R$ 0,60 e o do tipo 3 por R$ 0,75. Faça um programa que pergunte qual tipo de picolé, qual a quantidade e quanto você pagará por eles.

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa o *Scratch* deve perguntar a base e altura de um quadrado, e depois de um retângulo, e mostrar qual a área total destas figuras.

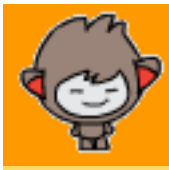
Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR AS SEGUINTES OPERAÇÕES USANDO OS BLOCOS QUE VOCÊ JÁ CONHECE?

Você deve fazer um programa que solicite ao usuário digitar 3 produtos e o preço de cada um. Após a leitura mostre o valor total dos produtos. Pague os produtos com R$ 100,00 e diga qual o troco ou se ficou devendo.

Você deve fazer um programa que leia a carga máxima de peso de um elevador e o peso individual de 5 pessoas. O *Scratch* deve informar se o elevador está liberado para subir ou se excedeu a carga máxima de peso.

Você deve fazer um programa que solicite que o usuário escolha um tipo de operação entre adição, subtração e multiplicação. Após a escolha, peça para o usuário digitar 5 números inteiros quaisquer. Ao final, o *Scratch* deve informar o resultado da operação escolhida.



VAMOS RESOLVER AS SITUAÇÕES-PROBLEMA

NESTA ATIVIDADE

Você deve resolver as seguintes operações:

20 + 5 – 4 + 3 =
15 – (9 – 2 + 3) =
[(20 – 11) – 10] =

Notícia Extra: Parênteses, Colchetes e Chaves

São sinais associativos que possuem uma ordem que deve ser respeitada. Você deve resolver as expressões matemáticas, pela ordem de prioridade, primeiramente os Parênteses ( ), depois os Colchetes [ ] e por último as Chaves { }.

VAMOS PROGRAMAR NO SCRATCH

Neste programa você deve construir um algoritmo que resolva as seguintes expressões numéricas:

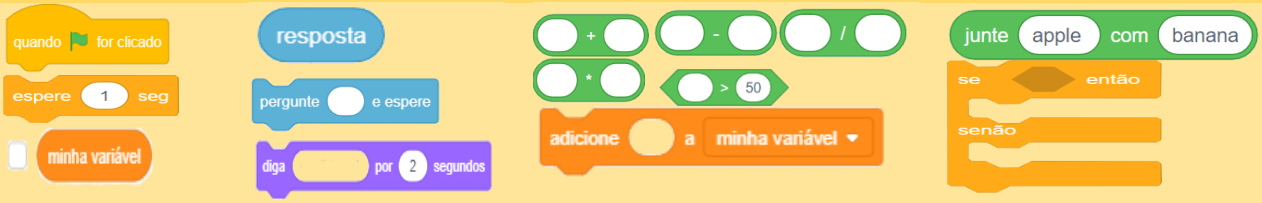
10 – 5 – 2 + 3 =
(10 – 5) – (2 + 3) =
70 – (5 x 4 + 2) x 3 =
10 – (5 – 2 + 3) =
20 + 5 x 8 – 2 =
80 – (3 x 7 + 9) x 2 =

Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

Neste programa, você vai construir duas variáveis resultado1 e resultado2. Elas receberão as respostas das expressões:

[18 - (6 + 4 x 7) : 2] =
{ [20 x (15 : 3)] - [108 - (4 + 6)] }=

Após a resolução diga qual resultado é o maior.



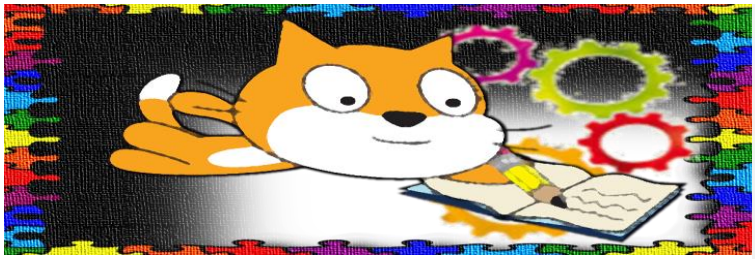
Lembre-se: Observe os blocos. Eles podem se repetir, e há várias possibilidades de resolver.

ESTOU DESAFIANDO VOCÊ! SERÁ QUE CONSEQUE CONSTRUIR AS SEGUINTES OPERAÇÕES USANDO OS BLOCOS QUE VOCÊ JÁ CONHECE?

Faça um programa que leia os valores de ALTURA E LARGURA de um retângulo. Apresente seu desenho e mostre sua área. Utilize para o cálculo, a fórmula: ÁREA = ALTURA x LARGURA.

Faça um programa que calcule e apresente o valor do volume de uma lata de óleo, utilize a fórmula: VOLUME = 3,14 x RAIO² x ALTURA.

Desafio a você construir um desenho somente com figuras geométricas no *Scratch*!



INFORMATIVO: NOTA AO LEITOR

DESPEDIDA: A VOCÊ, CARO LEITOR

O ACESSO E USO DE DIFERENTES METODOLOGIAS DE ENSINO, CONTRIBUI PARA O CRESCIMENTO PROFISSIONAL DO PROFESSOR, E EDUCACIONAL DO ALUNO, PROMOVENDO A OPORTUNIDADE PARA O DESENVOLVIMENTO DAS PRÁTICAS EDUCACIONAIS, E A CONSTRUÇÃO DO CONHECIMENTO.

O produto apresentado teve como objetivo descrever os principais elementos que compõem a estrutura de programação, envolvendo a lógica, matemática e o *software Scratch*, além das orientações para o desenvolvimento de programas. Com resultado final, pretende-se que constitua como um instrumento útil, a você professor e aluno, e que sirva também como um material de auxílio.

Considerações finais

Destacamos nesse produto aspectos importantes do processo de programação no sofware *Scratch*, envolvendo alguns conteúdos de matemática.

Para nós, as tecnologias digitais podem fazer parte das práticas pedagógicas. Porém, isso somente só será possível, quando certas mudanças ocorrerem, principalmente quando o professor e aluno estiverem inserido na cibercultura. Neste espaço, existem ambientes de aprendizagens, que contribuem para o desenvol-vimento e construção do conhecimento. Sendo assim, podemos dizer que a experiência com as tecnologias digitais, podem ofercer recursos significativos para o aprendizado, bem como auxiliar na reflexão das possibilidades ao ensino diferente, dinâmico, criativo e coletivo.