III SIMPÓSIO DA FORMAÇÃO
DO PROFESSOR DE MATEMÁTICA
DA REGIÃO NORTE

CRIAÇÃO DE ATIVIDADES NO GEOGEBRA COM A POSSIBILIDADE DE *FEEDBACK* AUTOMÁTICO

Diogo Meurer de Souza Castro
José Aparecido Sousa Santos

ANPMat
Associação Nacional dos Professores
de Matemática na Educação Básica

CRIAÇÃO DE ATIVIDADES NO GEOGEBRA COM A POSSIBILIDADE DE *FEEDBACK* AUTOMÁTICO

Criação de atividades no GeoGebra com a possibilidade de Feedback Automático.

Capa: Gabriel Brasil Nepomuceno
Projeto gráfico: Gabriel Brasil Nepomuceno
Diagramação e Assessoria editorial: Yunelsy Nápoles Alvarez

ISBN: 978-65-88013-12-0

Distribuição
Associação Nacional dos Professores de Matemática na Educação Básica
http://www.anpmat.org.br / email: editoraanpmat@anpmat.org.br

III SIMPÓSIO DA FORMAÇÃO
DO PROFESSOR DE MATEMÁTICA
DA REGIÃO NORTE

CRIAÇÃO DE ATIVIDADES NO GEOGEBRA COM A POSSIBILIDADE DE *FEEDBACK* AUTOMÁTICO

Diogo Meurer de Souza Castro
José Aparecido Sousa Santos

1ª edição

2021

Belém

ANPMat
Associação Nacional dos Professores
de Matemática na Educação Básica

Sobre os autores

José Aparecido Sousa Santos
ja83sousa@gmail.com

Licenciado em Matemática pela Universidade Federal de Alagoas - UFAL, mestre em Matemática pelo Mestrado Profissional em Matemática em Rede Nacional – PROFMAT/UFAL. Atualmente é professor da Prefeitura Municipal de Pilar e atua em pesquisas com ênfase em Educação Matemática e o uso de tecnologias.

Formado em Licenciatura em Matemática pela Universidade Federal de Alagoas - UFAL em 2008. No início de 2015, tornou-se mestre em Matemática pelo Mestrado Profissional em Matemática em Rede Nacional – PROFMAT/UFAL. Desde 2013 é professor do Instituto Federal de Educação, Ciência e Tecnologia de Alagoas e, atualmente, leciona no Campus Maceió e coordena o Curso de Licenciatura em Matemática do campus. Atua em pesquisas com ênfase em Educação Matemática e o uso de tecnologias.

Diogo Meurer de Souza Castro
diogo.castro@ifal.edu.br

Dedicamos este trabalho a nossos amigos e alunos.

Sumário

Lista de Figuras

Lista de Tabelas

Prefácio

Desde que entramos em 2013 no PROFMAT - Mestrado Profissional em Matemática em Rede Nacional (polo Universidade Federal de Alagoas), nós temos discutido e aprendido mais sobre o GeoGebra. O que começou mais como uma curiosidade, acabou finalizando nosso mestrado, tendo, cada um, optado por dissertações que utilizavam o *software*. Com o tempo, fomos estudando mais as possibilidades que o GeoGebra nos entregava além da manipulação dos objetos matemáticos em tela e de forma dinâmica. E uma dessas possibilidades é a construção de atividades em que os usuários podem aprender algum conteúdo da disciplina de forma orientada pelo(a) docente ou até sozinhos em seus *smartphones/tablets/*computadores. Ou, simplesmente, treinar atividades que são geradas automaticamente e, dentro do próprio *applet*, o(a) usuário(a) ter *feedbacks* automáticos.

No final de 2020, começamos a nos encontrar com o grupo de pesquisa "O GeoGebra como Estratégia para Ensino Remoto: Criando Atividades com *Feedback* Automático". Esse grupo é liderado pelos professores Celina Abar, José Manuel dos Santos e Marcio Almeida, cuja proposta do grupo é

> Oferecer a professores da escola básica, preferencialmente de escolas públicas, uma formação para sua atuação no contexto do ensino remoto com a utilização do *software* GeoGebra não apenas como mais um recurso tecnológico, mas como um recurso que colabore no desenvolvimento da prática docente, envolvendo conceitos matemáticos e métodos de avaliação automática. ([1], 2020, p.1)

Então, com a possibilidade de submetermos uma proposta de minicurso no III Simpósio da Formação do Professor de Matemática da Região Norte, pensamos em apresentar o que temos estudado e desenvolvido no GeoGebra cujo foco está na criação de atividades com a possibilidade de *feedback* automático.

Acreditamos que apresentar aos professores e alunos de graduação tal possiblidade que o GeoGebra nos oferece faz com que os leitores possam ter mais uma ferramenta para o ensino e aprendizagem da matemática. Mesmo não sendo nosso intuito de trazer um manual completo do GeoGebra, esperamos que esse livro possa ajudar o(a) leitor(a) a compreender um pouco mais dos comandos e ferramentas disponíveis no *software* fazendo com que as atividades fiquem mais elaboradas e atrativas para o(a) usuário(a).

Maceió, 27 de setembro de 2021.

Diogo Meurer, José Aparecido.

Capítulo 1

Introdução

O GeoGebra foi desenvolvido, em 2011, por Markus Hohenwarter em sua tese de doutorado. Com a proposta de agregar a Geometria e a Álgebra em um mesmo ambiente e de forma dinâmica, o *software*, desde então, tem conquistado prêmios e a admiração de vários professores e estudantes formando uma comunidade de milhões de usuários. Além da Geometria e da Álgebra, o *software* permite-nos trabalhar com Planilha de Cálculo, Gráficos, Probabilidade, Estatística e Cálculos simbólicos, e podendo trabalhar e visualizar as diversas representações de um mesmo objeto matemático [5].

Há vários estudos que analisaram o potencial dessa ferramenta para o ensino e aprendizagem da matemática. Em um estudo desenvolvido, [3] encontraram evidências de que o GeoGebra tem impacto positivo nos estudantes como melhora do entusiasmo, confiança e motivação para aprender. Já [2], encontram em seus resultados que "Os ambientes de geometria dinâmica potencializam o raciocínio lógico-dedutivo, através da visualização de resultados invariantes perceptíveis pelos movimentos das figuras dinâmicas."

Além disso, temos a Base Nacional Curricular Comum [4] , onde

> orienta-se pelo pressuposto de que a aprendizagem em Matemática está intrinsecamente relacionada à compreensão, ou seja, à apreensão de significados dos objetos matemáticos, sem deixar de lado suas aplicações. [...] Desse modo, recursos didáticos como malhas quadriculadas, ábacos, jogos, livros, vídeos, calculadoras, planilhas eletrônicas e **softwares de geometria dinâmica têm um papel essencial para a compreensão e utilização das noções matemáticas**. [grifo nosso]

Para além de termos o GeoGebra como ferramenta para uso pessoal/estudo ou para utilização em sala de aula para a explicação de algum conteúdo/conceito matemático (por exemplo, mostrar, de forma dinâmica, que a função y = ax+b intercepta o eixo-y no ponto (0,b)), o *software* permite que os usuários criem suas próprias construções/tarefas. Esses *applets* podem ser disponibilizados no próprio *site* do GeoGebra ou transformá-los em arquivos html.

Seguindo a linha da criação de atividades, podemos desenvolvê-las de tal forma que o *applet* forneça *feedbacks* para o(a) usuário(a) dos mais variados possíveis: em forma de texto, imagem, áudio etc. Podemos, por exemplo, indicar se a resposta está correta ou errada, indicar qual erro foi cometido, dar dicas a partir de uma certa quantidade de erros etc. Tudo isso é possível com um pouco de lógica e comandos que o próprio GeoGebra nos fornece e que não precisa de nenhum conhecimento mais rebuscado de programação computacional. Ou seja, abre-se um leque de possibilidades para nós professores e futuros professores construirmos nossas próprias atividades em que o(a) aluno(a) pode aprender/exercitar sozinho(a).

Como exemplo de atividade com *feedback*, mostramos a atividade que construímos chamada Dois quadrados em um só?. Nela, buscamos que o(a) aluno(a) investigue se é possível, com dois quadrados de lados inteiros, preencher a área de um quadrado de lado inteiro maior. Inicialmente, apresentamos algumas instruções para a atividade e, após as instruções, temos a aplicação em si. À medida que o aluno vai tentando achar um valor para l e n, a atividade vai dando *feedbacks* (Figura 1.1) e dicas para o(a) usuário(a) ou sugerindo maiores detalhes da tarefa.

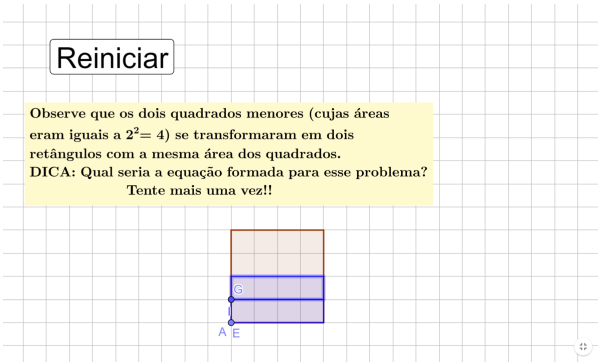


Figura 1.1: *Feedbacks* da atividade

Posto isso, neste e-*book* iremos apresentar, além das atividades que foram trabalhadas no minicurso, quais são as principais ferramentas e comandos para a criação de tais atividades no GeoGebra, com a possibilidade, também, de *feedback* automático. Nos dois próximos capítulos traremos algumas ferramentas e comandos que utilizamos em construções de atividades desse tipo. Além disso, para cada explicação, apresentaremos um *link* com um exemplo de como utilizar cada ferramenta/comando. No quarto capítulo deste trabalho, apresentaremos as atividades que foram desenvolvidas no minicurso com todo o passo a passo necessário para a construção, além de dicas e sugestões. No último capítulo, apresentaremos nossas considerações finais.

Capítulo 2

Ferramentas

2.1 Controle Deslizante

O controle deslizante tem várias funcionalidades e que podem nos ajudar a dar maior dinamicidade para nossas construções. Para criarmos um controle deslizante, basta ir ao *menu* de ferramentas e escolher a ferramenta de controle deslizante (Figura 2.1).

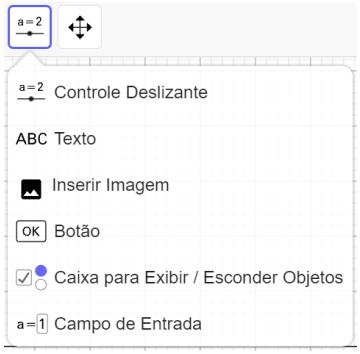


Figura 2.1: Controle deslizante

Após isso, clicamos na janela de visualização e aparecerá uma tela como na Figura 2.2. Essa tela nos dá algumas opções para esse controle: **Número, Ângulo e Inteiro** além de três abas:

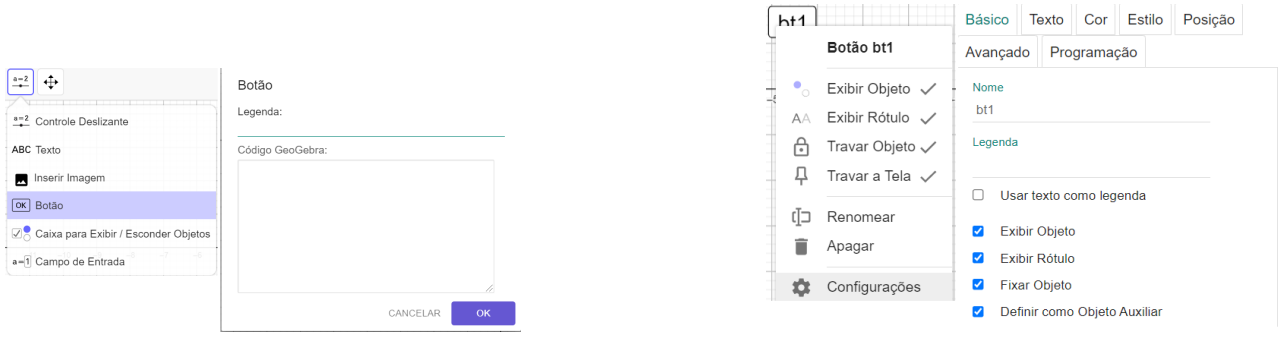
- Na aba **Intervalo**, podemos determinar qual será o menor e o maior valor para o nosso controle e o seu incremento.
- Na aba **Controle deslizante** podemos escolher se a linha será vertical ou horizontal (caso você não decida, o GeoGebra escolhe o horizontal).
- Na aba **Animação** podemos escolher a velocidade que o controle irá passar os números, além de qual tipo de repetição queremos para o controle.

Figura 2.2: Opções para o controle deslizante

Exemplo
https://www.geogebra.org/m/qcq3nabt

2.2 Botão

Esta ferramenta é usada para executar uma série de comandos do GeoGebra com um único clique em um botão. Embora os comandos possam ser acionados clicando em qualquer outro objeto (por exemplo, uma imagem), o uso de botões deixa o seu *applet* mais intuitivo. Para criar um botão, vamos no *menu* de ferramentas, ao clicar em qualquer parte da Janela de Visualização aparece a tela abaixo (Figura 2.3(a)) onde é pedida a legenda que o botão terá na tela e o(s) código(s) que serão necessários. Ao criarmos qualquer botão, podemos acessar mais propriedades (Figura 2.3(b)) clicando com o botão direito em cima do botão e ir em **Configurações**.



(a) Criação do botão.

(b) Configurações do botão.

Figura 2.3: Botão

Aconselhamos o(a) leitor(a) a testar as funcionalidades e verificar o que cada opção faz no botão. Aqui, chamaremos atenção à aba **Programação**. Caso seja necessário alterar um comando/código ou até acrescentar um novo, é nessa aba que podemos ajustar nosso botão (ou a programação de qualquer outro objeto). Nela, como é possível ver na imagem abaixo, há outras três abas:

- A aba **Ao Clicar** faz com que todo comando inserido nela seja executado ao clicar no botão;
- Todo comando que seja inserido na aba **Ao Atualizar** é executado quando a construção no GeoGebra é atualizada;
- Na aba **JavaScript** Global é para ser inserida programação em Java.

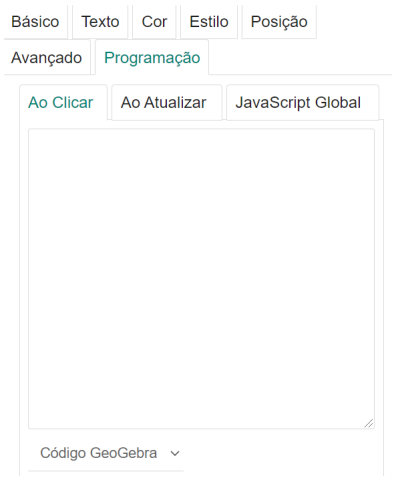


Figura 2.4: Aba Programação

Exemplo
https://www.geogebra.org/m/bvavdcyn

2.3 Texto

Essa ferramenta dar-nos-á imensas possibilidades de *feedbacks* em nossas atividades. Podemos criar textos estáticos, dinâmicos, atrelados a uma certa condição para que ele apareça etc. Além disso, há vários comandos relacionados a textos. A criação de um texto pode ser feita através do *menu* de Ferramentas, como mostrado na imagem abaixo.

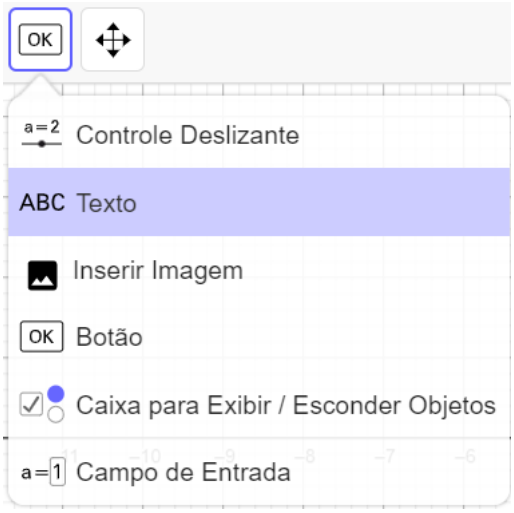


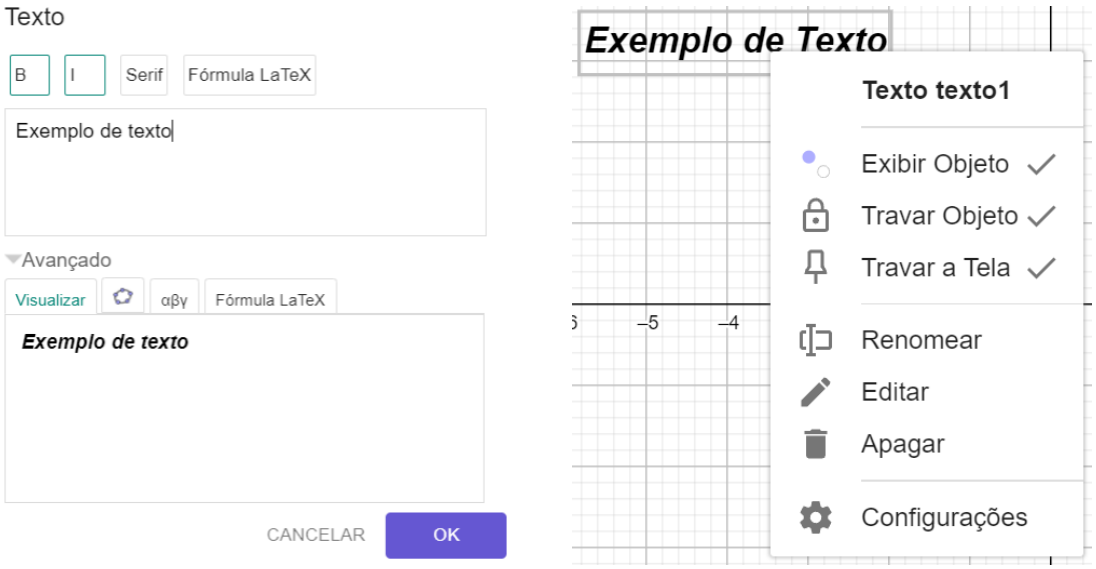
Figura 2.5: Ferramenta Texto

Quando criamos um texto na Janela de Visualização, as primeiras opções de edição (Figura 2.6(a)) são colocar o texto em negrito, itálico, modificar a fonte ou escrever no formato LaTeX. Ao clicar em **Avançado**, uma nova parte abre-se onde podemos visualizar o que está sendo digitado, inserir alguns caracteres especiais (aba $\alpha\beta\gamma$) e algumas fórmulas LaTeXpredefinidas. A segunda aba permite-nos incorporar no texto objetos criados no próprio GeoGebra, e falaremos sobre ela posteriormente. É muito importante lembrarmos de, ao posicionar o seu texto no lugar desejado, travá-lo na tela. Para isso, clique com o botão direito sobre o texto e ative as opções **Travar Objeto** e **Travar a Tela** (Figura 2.6(b)).

Exemplo
https://www.geogebra.org/m/yn8tn5pj/pe/1173037

2.4 Exibir/Esconder Objeto

Com esta ferramenta você pode criar uma caixa de seleção (com valores booleanos: *true* e *false*) que permite mostrar e ocultar um ou mais objetos. Na janela de diálogo exibida, você pode especificar quais objetos devem ser afetados pela caixa de seleção.



(a) Ambiente Texto.

(b) Travar texto e tela.

Figura 2.6: Texto

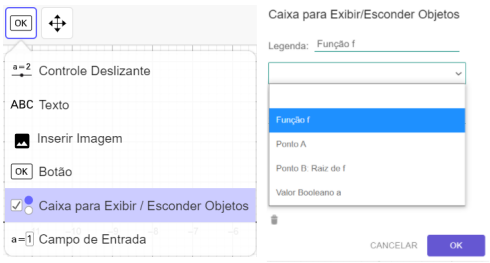


Figura 2.7: Caixa para exibir/esconder objetos

Exemplo
`https://www.geogebra.org/m/mpuwurnz`

2.5 Campo de Entrada

A ferramenta **Campo de Entrada** é usada para alterar interativamente a definição de vários objetos. Ao selecionar a ferramenta, clique na janela de visualização que será exibida em uma caixa de diálogo, onde você pode definir sua legenda e o objeto vinculado.

Figura 2.8: Campo de entrada

Como essa ferramenta abrirá um teclado para que seja digitado algo, temos que ter em mente o que

queremos que seja digitado para poder escolher melhor a sua configuração. Quando criamos qualquer campo de entrada no GeoGebra, nas configurações do campo, há a aba **Álgebra**, e podemos escolher se queremos o teclado simbólico ou não. Esse teclado simbólico irá abrir um maior leque de opções para o(a) usuário(a). Nele, conseguimos colocar raízes, potências, letras gregas etc. Se usamos no celular, o teclado abre automaticamente (figura 2.9); se usamos no computador, podemos abri-lo clicando na imagem de um teclado no canto inferior esquerdo (figura 2.10).

Figura 2.9: Teclado simbólico no celular

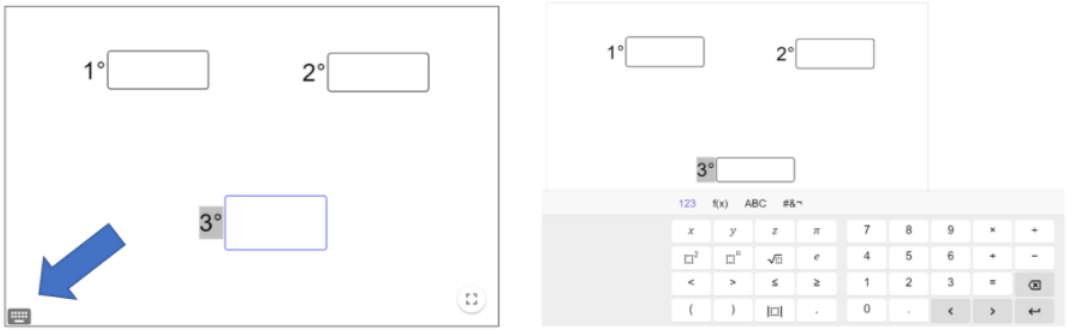


Figura 2.10: Teclado simbólico no computador

Outro ponto importante a ser alertado é que, quando utilizamos o celular, podemos ter duas situações: a primeira é quando vinculamos um número ao campo de entrada. Ao tentarmos digitar algo, dependendo do sistema operacional, o teclado numérico que se abre pode não ter algumas funcionalidades, como o sinal da subtração. Isso pode acarretar algum problema na realização da atividade porque, por exemplo, podemos querer colocar um número negativo e no IOS não conseguiríamos. A segunda situação é quando vinculamos um texto ao campo de entrada. Nesse caso, em ambos os sistemas operacionais, abre-se o teclado normal como é mostrado na Figura 2.11 (da esquerda, no IOS; da direita, no Android).

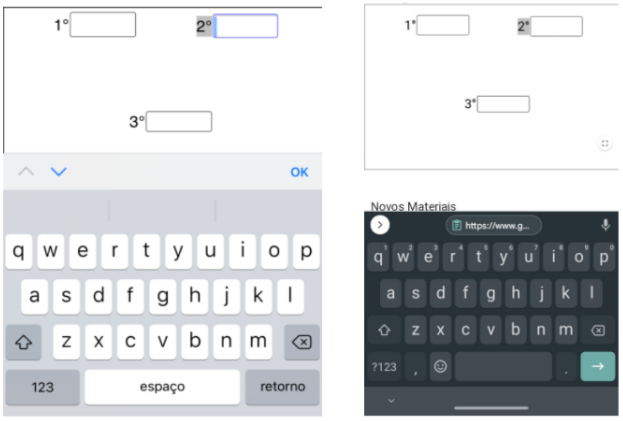


Figura 2.11: Campo de entrada vinculado a um texto

Exemplo
`https://www.geogebra.org/m/rxtxyth9`

Capítulo 3

Comandos

O GeoGebra contém vários comandos que podem ser inseridos nas programações de outros objetos ou serem inseridos no próprio campo de entrada. Para ter acesso a uma lista desses comandos, na Janela de Álgebra, clique no botão + e em **Ajuda**.

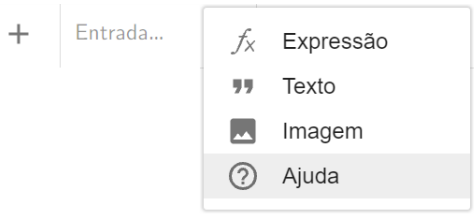


Figura 3.1: Ajuda

O programa dá-nos uma lista com os comandos separados por tipo, e, ao clicar em um, aparece a estrutura do comando. Caso precise de mais ajuda, basta clicar em **Exibir Ajuda Online** para ter maiores informações no *site* do programa (ainda com explicações em inglês).

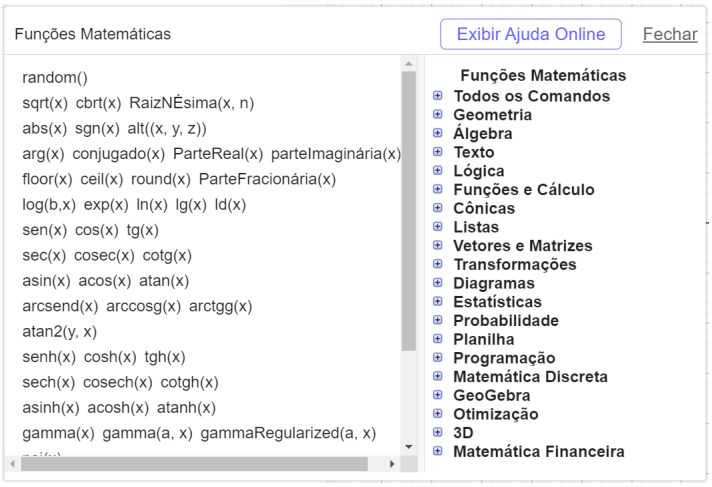


Figura 3.2: *Menu* ajuda

Neste capítulo iremos abordar alguns comandos que utilizamos com frequência e que achamos importante que o(a) leitor(a) possa se aprofundar mais sobre eles.

3.1 Comandos de Lista

3.1.1 Sequência

O comando sequência é um dos mais poderosos comandos do GeoGebra. Muitas coisas que deveriam ser construídas manualmente (por exemplo, construir cem quadrados de lado 1 dentro de um quadrado de lado 10) podem ser feitas simplesmente utilizando esse comando. Além disso, ele não fica limitado a somente sequências numéricas, pois podemos inserir qualquer comando de objetos. As possíveis sintaxes são:

- **Sequência(<Valor Final>)** - sequência numérica de números naturais onde <Valor Final> será o último número da sequência.
- **Sequência(<Valor Inicial>, <Valor Final>)** - sequência numérica de números inteiros com valor inicial e final determinado

- **Sequência(<Valor Inicial>, <Valor Final>, <Incremento>)** - progressão aritmética cuja razão é o valor do <Incremento>

Exemplo
https://www.geogebra.org/m/x6xgepvc

- **Sequência(<Expressão>, <Variável>, <Valor Inicial>, <Valor Final>)** - nessa sintaxe é inserida uma expressão com uma variável. Tal expressão pode ser uma função ou um outro comando do próprio Geogebra
- **Sequência(<Expressão>, <Variável>, <Valor Inicial>, <Valor Final> , <Incremento>)** - uma extensão do comando acima, onde podemos acrescentar um incremento

Exemplo
https://www.geogebra.org/m/bfxt6dvb

3.1.2 Escolher elemento aleatoriamente

Esse comando escolhe um elemento aleatório de uma lista que pode ser criada separadamente ou pode ser inserida dentro do próprio comando. A sintaxe é bem simples:

- **EscolherElementoAleatoriamente(<Lista>)**

Exemplo
https://www.geogebra.org/m/rpdmmx3n

3.1.3 Concatenar, anexar e remover

O comando concatenar tem duas sintaxes:

- **Concatenar(<Lista>, <Lista>, ...)** - Cria uma nova lista unindo duas ou mais listas. A nova lista contém todos os elementos das listas iniciais, mesmo que sejam iguais. Os elementos da nova lista não são reordenados.
- **Concatenar (<Lista de Listas>)** - Cria uma nova lista juntando as sublistas da lista. A nova lista contém todos os elementos das listas iniciais, mesmo que sejam iguais. Os elementos da nova lista não são reordenados.

Exemplo
https://www.geogebra.org/m/gajcbtcw

O comando anexar tem duas sintaxes:

- **Anexar(<Lista>, <Objeto>)** - Acrescenta o objeto à lista e produz os resultados em uma nova lista.
- **Anexar(<Objeto>, <Lista>)** - Acrescenta a lista ao objeto e produz os resultados em uma nova lista.

Exemplo
https://www.geogebra.org/m/khqc3u7r

O comando Remover(<Lista>,<Lista>) remove objetos da primeira lista sempre que eles aparecem na segunda lista. Já o comando RemoverIndefinidos(<Lista>) remove objetos indefinidos de uma lista.

Exemplo
https://www.geogebra.org/m/szzeer72

3.2 Comandos de Lógica

3.2.1 Está definido, É inteiro, e Pertence à região

- O comando **EstáDefinido(<Objeto>)** retorna um valor verdadeiro ou falso, dependendo se o objeto está definido ou não.
- O comando **ÉInteiro(<Número>)** retorna um valor verdadeiro ou falso, dependendo se o número é inteiro ou não.
- O comando **PertenceARegião(<Ponto>, <Região>)** retorna um valor verdadeiro, se o ponto está em determinada região, e falso, caso contrário.

Exemplo
https://www.geogebra.org/m/fupytbsc

3.2.2 Comando Se

É um comando que realiza um teste lógico de uma expressão. O GeoGebra apresenta três sintaxes para o comando Se:

- **Se[<Condição>, <Então>]**

  <Condição> : é a expressão que você deseja testar; <Então>: é o valor que você deseja retornar se o resultado do teste lógico for VERDADEIRO;

  Exemplo
  https://www.geogebra.org/m/kzsddvxn

- **Se[<Condição>, <Então>, <Senão>]**

  <Condição> : é a expressão que você deseja testar; <Então>: é o valor que você deseja retornar se o resultado do teste lógico for VERDADEIRO, e <Senão>: é o valor que você deseja retornar se o resultado do teste lógico for FALSO.

  Exemplo
  https://www.geogebra.org/m/x84ncfm5

- **Se[<Condição 1>, <Então 1>,<Condição 2>, <Então 2>,..., <Senão>(opcional)]**

  <Então 1> é o valor que você deseja retornar se a <Condição 1> for satisfeita, <Então 2> é o valor que você deseja retornar se a <Condição 2> for satisfeita etc. Se nenhuma das condições for satisfeita e <Senão> for fornecido, esse comando produzirá o valor de <Senão>. Caso contrário, objeto indefinido é retornado.

Exemplo
https://www.geogebra.org/m/azqpbnbw

3.2.3 Contar Se

Esse comando conta o número de elementos na lista que satisfazem a condição. O GeoGebra apresenta duas sintaxes para o comando **ContarSe[].**

- **ContarSe(<Condição>, <Lista>)**
- **ContarSe(<Condição>, <Variável>, <Lista>)**

Exemplo
https://www.geogebra.org/m/efrsyrzd

3.3 Comandos de Programação

3.3.1 IniciarAnimação[]

O GeoGebra apresenta quatro sintaxes para o comando **IniciarAnimação[]**.

- **IniciarAnimação[]** - Retoma todas as animações se elas estiverem pausadas.
- **IniciarAnimação[<true | false>]** - Quando o valor booleano é falso, pausa todas as animações; caso contrário, as retoma.
- **IniciarAnimação[<Controle Deslizante ou Ponto>, <Controle Deslizante ou Ponto>, ...]** - Começa a animar determinados pontos e controles deslizantes, os pontos devem estar em caminhos.
- **IniciarAnimação[<Controle Deslizante ou Ponto>, <Controle Deslizante ou Ponto>, ..., <true | false>]** - Inicia (para booleano = verdadeiro) ou para permanentemente (para booleano = falso), animando pontos e controles deslizantes dados, os pontos devem estar em caminhos.

Exemplo
https://www.geogebra.org/m/k9kyqq5b

3.3.2 DefinirCor e DefinirCordeFundo

O GeoGebra apresenta duas sintaxes para o comando **DefinirCor[]**

- **DefinirCor(<Objeto>, <Cor>)** - Esse comando altera a cor de um determinado objeto. A cor é inserida como texto, que pode ser:
  - um nome de uma cor em inglês (*Black, Gray, Dark Blue, Blue, ...*)
  - uma *string* hexadecimal do tipo #AARRGGBB ou #RRGGBB, onde AA define a transparência (00 transparência total a FF opacidade total), RR define o componente vermelho, GG o verde e BB o azul.
- **DefinirCor(<Objeto>, <Vermelho>, <Verde>, <Azul>)** - altera a cor de um determinado objeto. O vermelho, o verde e o azul representam a quantidade do componente de cor correspondente, sendo 0 mínimo e 1 máximo.

Exemplo
https://www.geogebra.org/m/sac3bnew

O GeoGebra apresenta duas sintaxes para o comando **DefinirCorDeFundo[]**:

- **DefinirCorDeFundo(<Objeto>, <Cor>)** - altera a cor de fundo de um determinado objeto. Isso é usado para textos, botões, campo de entrada etc. A cor é inserida como texto, que pode ser:
  - um nome de uma cor em inglês (*Black, Gray, Dark Blue, Blue, ...*)
  - uma *string* hexadecimal do tipo #AARRGGBB ou #RRGGBB, onde AA define a transparência (00 transparência total a FF opacidade total), RR define o componente vermelho, GG o verde e BB o azul.
- **DefinirCorDeFundo(<Objeto>, <Vermelho>, <Verde>, <Azul>)** - Esse comando altera a cor de fundo de um determinado objeto. O vermelho, o verde e o azul representam a quantidade do componente de cor correspondente, sendo 0 mínimo e 1 máximo.

Exemplo
https://www.geogebra.org/m/t3eajkcg

Existem vários *sites* que nos auxiliam a escolher uma cor desejada e, principalmente, qual código iremos inserir no comando do GeoGebra. Uma sugestão que damos é o *site* Color hex onde é possível escolher a cor e pegar qual o código da cor. Por exemplo, veja na imagem abaixo que estamos escolhendo a cor avermelhada indicada. Já aparece o código #ee3a3a para ela. Será esse código que colocaremos no comando do GeoGebra.

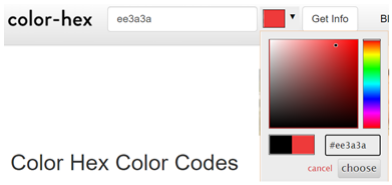


Figura 3.3: *Site* Color Hex

Então, se temos um ponto A que queremos que fique com essa cor, usaremos o comando DefinirCor(A, "#ee3a3a").

3.3.3 Definir valor

O GeoGebra apresenta três sintaxes para o comando **DefinirValor()**:

- **DefinirValor(<Variável Booleana>, <0|1>)** - Esse comando define o estado de uma caixa de seleção / booleana: 1 = verdadeiro, 0 = falso
- **DefinirValor(<Objeto>, <Objeto>)** - Seja A o primeiro objeto e B o segundo. Se A for um objeto livre ou um ponto restrito ao caminho ou região, seu valor é definido para o valor atual de B.
- **DefinirValor(<Lista>, <Número>, <Objeto>)** - Seja n um número. Esse comando define o enésimo elemento de uma lista livre L com o valor atual do objeto. O número n pode ser no máximo 1 + comprimento de L.

Exemplo
https://www.geogebra.org/m/shpjxjfc

3.3.4 Definir coordenadas

O GeoGebra apresenta a seguinte sintaxe para o comando **DefinirCoordenadas[]**:

- **DefinirCoordenadas(<Objeto>, <x>, <y>)** - Esse comando altera as coordenadas cartesianas de objetos livres. O comando usa valores das coordenadas, não suas definições, portanto o objeto permanece livre. O comando funciona para controles deslizantes, botões, caixas de seleção, caixas de entrada e imagens. Se a "Posição absoluta da tela" for selecionada, x , y estarão em *pixels* da tela. Também funciona para pontos em caminhos e regiões. O ponto será movido para o local mais próximo possível.

Exemplo
https://www.geogebra.org/m/wkewtxpq

3.4 Comandos de Texto

3.4.1 Tabela de texto

Esse comando cria um texto que contém uma tabela dos objetos da lista. Há duas sintaxes:

- **TabelaDeTexto(<Lista>, <Lista>, ...)**
- **TabelaDeTexto(<Lista>, <Lista>, ..., <Alinhamento do Texto>)**

O texto opcional "Alinhamento do texto" controla a orientação e o alinhamento do texto da tabela, bem como o alinhamento do separador em valores decimais.
Veja alguns valores possíveis: "vl_", "hc_|", "vr_|", "v", "h" , "{", etc. O valor padrão é "hl"

- "v" = vertical, ou seja, as listas são colunas
- "h" = horizontal, ou seja, as listas são linhas
- "l" = alinhado à esquerda
- "r" = alinhado à direita
- "c" = centralizado
- "_" = separadores de linha
- "|" = separadores de coluna
- "{" = insere chave à esquerda

Exemplo
https://www.geogebra.org/m/tnzyq7yw

3.4.2 Texto para unicode e unicode para texto

Unicode é o padrão de codificação de caracteres. O comando '**TextoParaUnicode(<Texto>)** transforma o texto em uma lista de números Unicode, um para cada caractere. O comando **UnicodeParaTexto(<Lista de Inteiros>)** converte os números Unicode inteiros de volta em texto.

Exemplo
`https://www.geogebra.org/m/xhmqba9z`

Capítulo 4

Atividades

Neste capítulo traremos as atividades que foram apresentadas no minicurso e algumas outras que achamos interessante mostrar, mas sempre enfatizando a possibilidade de *feedback* nas atividades. Vocês perceberão que a programação para os *feedbacks* envolve, basicamente, a utilização de caracteres de lógica: se, e, ou, negativo etc. Abaixo, apresentamos alguns caracteres que são muito utilizados:

Tabela 4.1: Símbolos lógicos

Símbolos	**Nomes**
==	igual
!=	diferente
&&	e
||	ou
!	negativa
<=	menor igual
>=	maior igual

4.1 Atividade 1 - Determinar as coordenadas de um ponto dado

Nessa primeira atividade, vamos criar um ponto aleatório A e o objetivo é que a pessoa determine as coordenadas desse ponto inserindo nos dois campos de entrada (Figura 4.1). Serão três tipos de *feedback* inseridos:

1) Caso a resposta seja correta, aparecerá o texto PARABÉNS

2) Caso a resposta seja incorreta, aparecerá o texto TENTE NOVAMENTE

3) Caso haja a troca das coordenadas, ou seja, o(a) usuário(a) escreva o ponto (y_A, x_A) nos campos de entrada, aparecerá o texto LEMBRE QUE UM PONTO É DO TIPO (x, y).

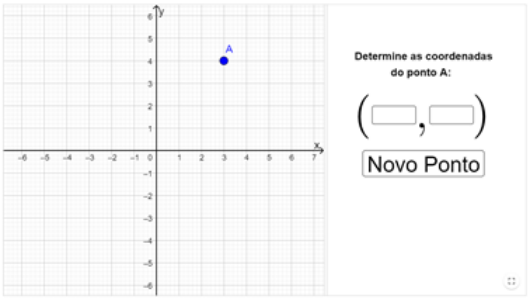


Figura 4.1: Atividade 1

Para a criação dessa atividade, siga os passos abaixo:

1. Abra uma segunda Janela de Visualização. Para isso, clique no *menu* superior direito e clique em **Exibir**:

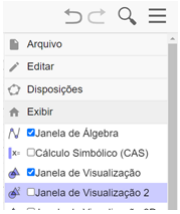


Figura 4.2: Habilitar segunda janela de visualização

2. Crie uma sequência onde limitará os valores de x e y do ponto com o comando **l1=Sequência(-5,5)**

3. Crie as coordenadas aleatórias do ponto:

   a=EscolherElementoAleatoriamente(l1)

   b=EscolherElementoAleatoriamente(l1)

4. Construa o ponto **A=(a,b)**

5. Agora, determine dois números onde ficarão as respostas do(a) usuário(a). Para isso, digite no campo de entrada: **ar=?** e depois **br=?**

6. Construa os dois campos de entrada, vinculando a ar e br sem nenhuma legenda.

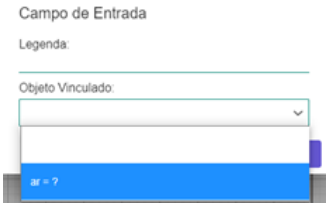


Figura 4.3: Primeiro campo de entrada vinculando a ar

7. Em cada campo de entrada criado vá em **Configurações** e desmarque a opção **Exibir Rótulo** na aba **Básico**; e na aba **Estilo** coloque o comprimento igual a 5.

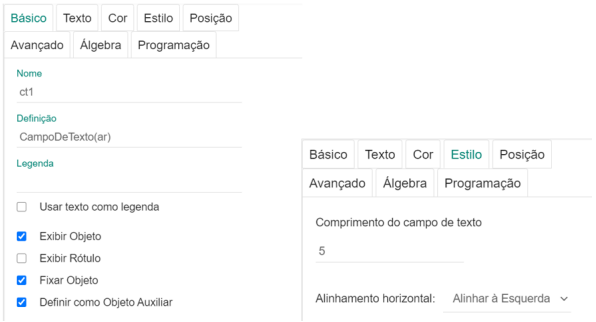


Figura 4.4: Configurações para cada campo de entrada

8. Posicione os campos de entrada um do lado do outro e escreva um texto para ficar em torno dos campos de entrada, para mostrar que estão sendo digitadas as coordenadas de um ponto.

Figura 4.5: Sugestão para a posição dos campos de entrada

9. Crie um botão e coloque a Legenda **REINICIAR**. Além disso, coloque os comandos abaixo:

   AtualizarConstrução() – Atualiza todos os objetos

   DefinirValor(ct1,“”) – Deixa o primeiro campo de entrada vazio

   DefinirValor(ct2,“”) – Deixa o segundo campo de entrada vazio

10. Agora, para começarmos a parte do *feedback*, vamos criar dois valores booleanos para que o GeoGebra saiba se há algo digitado no campo de entrada ou não. Para isso, vamos utilizar o comando **EstáDefinido(<Objeto>)**:

    c=EstáDefinido(ar)

    d=EstáDefinido(br)

11. O primeiro texto que será feito é para o caso em que acertem. Então, crie o texto **PARABÉNS** na segunda Janela de Visualização e acesse as suas configurações. É agora que começamos a programar em qual(is) momento(s) cada texto deve aparecer. Para que o ponto digitado esteja correto, as condições devem ser: ar = a e br = b, certo? Então, na aba **Avançado**, na parte **Condição para Exibir Objeto(s)**, você digita como aparece na Figura 4.6:

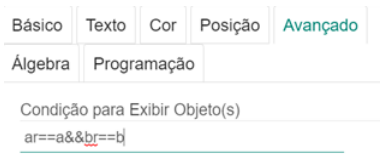


Figura 4.6: Condição para exibir o texto

No GeoGebra, quando queremos utilizar o igual, nessa parte de condições, utilizamos o == (verificar a tabela no início do capítulo). Por isso que ficou ar == a, br == b. Para o conectivo lógico E utilizamos o &&. Quando damos o ENTER, o programa já transforma para os códigos do próprio *software*. Veja:

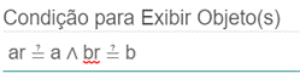


Figura 4.7: Código no formato do GeoGebra

Feito isso, faça um teste digitando as coordenadas corretas e veja que o texto irá aparecer. Ao clicar em REINICIAR, o texto desaparece.

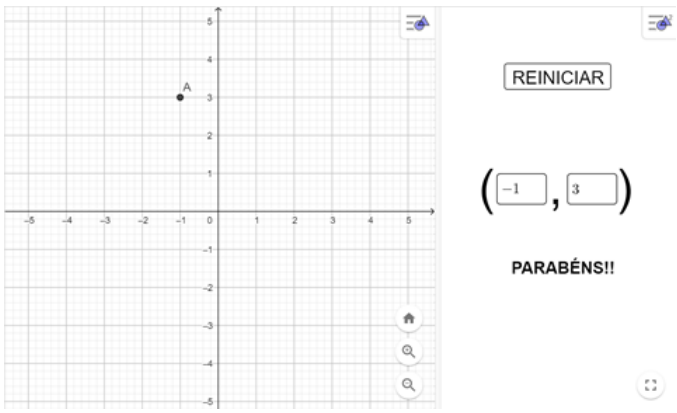


Figura 4.8: Código no formato do GeoGebra

12. Crie o texto **TENTE NOVAMENTE** que irá aparecer quando forem digitados os valores errados do ponto, ou seja, quando ar for diferente de a, e br for diferente de b. No GeoGebra, quando queremos utilizar o diferente, usamos o código !=. Mas isso só não basta, porque precisamos ter algo digitado no campo de entrada para fazer a comparação. Por isso que criamos os dois valores booleanos anteriormente. Então, além da comparação entre ar e br com a e b, devemos "exigir" que ar e br estejam definidos (valor booleano *true*). Para deixarmos todas essas condições para esse texto, digitamos o seguinte:

    c==true && d==true && ar!=a && br!=b

13. O terceiro e último *feedback* dessa tarefa é criar o texto **LEMBRE QUE UM PONTO É DO TIPO (x,y).** A condição de aparecer esse texto vai ser um pouco parecida com a do texto anterior, mas queremos que ele apareça quando há uma troca na posição entre os valores, ou seja, ar=b e br=a. Então, a condição será:

    c==true && d==true && ar==b && br==a

    Observe que, no exemplo abaixo, a resposta correta é (3,2), mas foi digitado (2,3). Então, o *feedback* apareceu abaixo o TENTE NOVAMENTE.

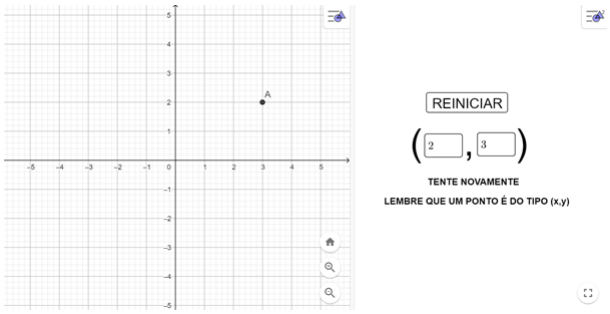


Figura 4.9: *Feedback* textual

4.2 Atividade 2 - Determinar no plano cartesiano um ponto previamente dado

Esta segunda atividade tem como proposta o contrário da atividade anterior. Ela entregará três coordenadas, e o(a) usuário(a) deve posicionar os pontos na posição correta. Os *feedbacks* serão:

1) Quando os três pontos estiverem posicionados corretamente, aparecerá o texto: Parabéns! Você conseguiu!!

2) Quando aparecerá o texto: Resposta Incorreta.

3) Quando o ponto for posicionado com a abscissa e a ordenada na forma contrária, ou seja, (y,x), aparecerá o texto: Atenção! Um ponto é representado por um par ordenado (x,y)

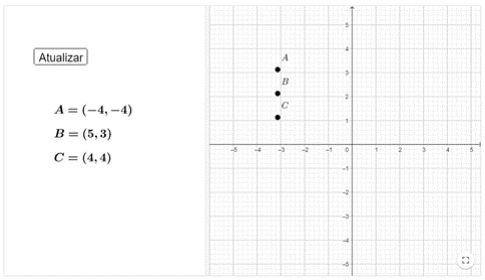


Figura 4.10: Atividade 2

Para isso, siga os seguintes passos:

1. Abra uma segunda Janela de Visualização deixando os eixos nessa janela.

2. Digite os comandos no campo de entrada e esconda todos esses objetos:

   l1=Sequência(Sequência((i,j),i,-5,5),j,-5,5) - Formará listas de pontos (para mais detalhes, ver o tópico Comando de Listas do capítulo anterior).

   l2=Concatenar(l1) - juntará todas as listas em uma só

   l3=Embaralhar(l2) - irá embaralhar os elementos da lista anterior

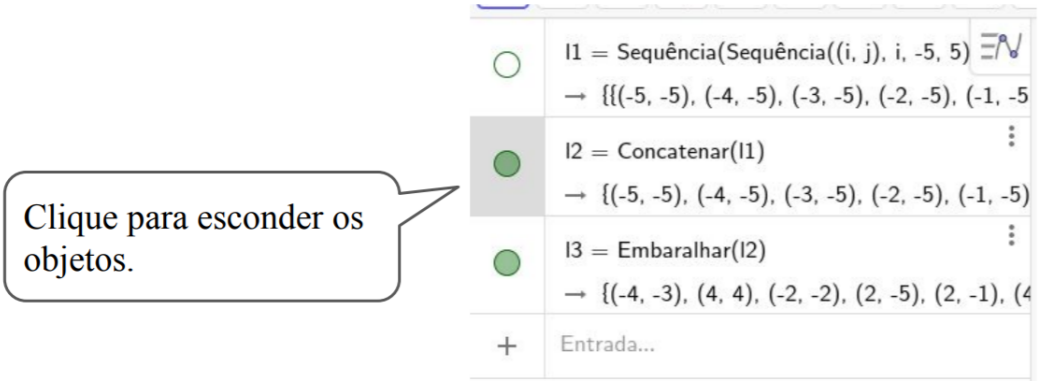


Figura 4.11: Esconder a lista

3. Crie um texto para o ponto A, onde estará ligado ao primeiro elemento da lista l3.

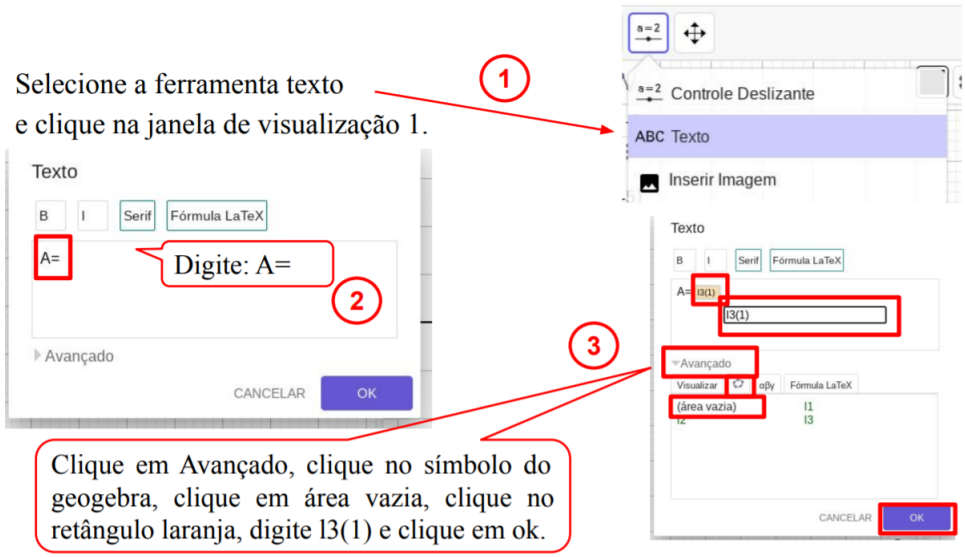


Figura 4.12: Texto para o ponto A

4. Faça o mesmo para os outros dois pontos.

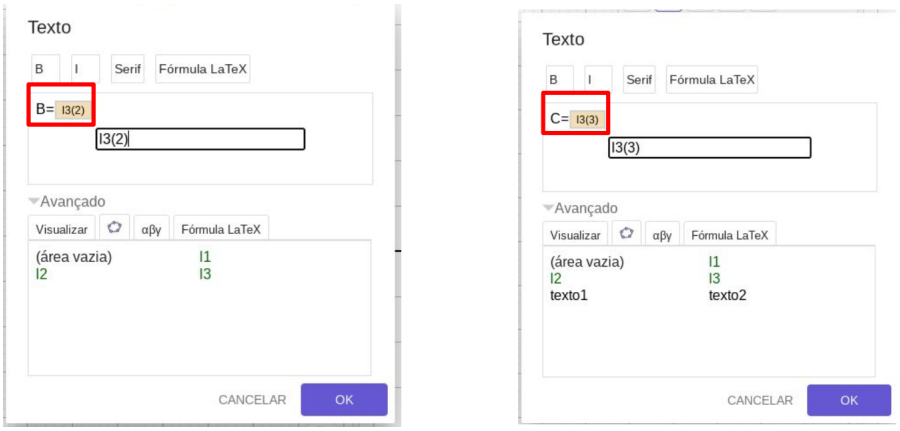


Figura 4.13: Texto para os pontos B e C

5. Posicione os três textos na primeira Janela de Visualização:

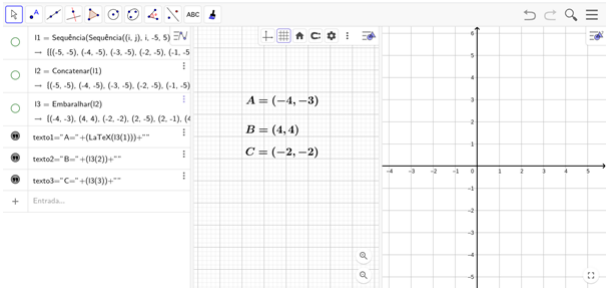


Figura 4.14: Posicionamento dos textos

6. Agora, construa três pontos com posições iniciais fixas que serão utilizados pelo(a) aluno(a).
   Ar = (-pi,pi) - com legenda $\math{A}$
   Br = (-pi,pi-1) - com legenda $\math{B}$
   Cr = (-pi,pi-2) - com legenda $\math{C}$
7. Construa três textos que servirão como *feedback* para a nossa atividade:

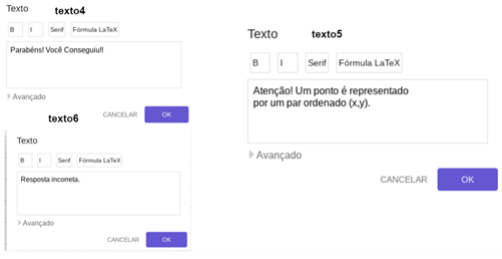


Figura 4.15: Textos para *feedback*

8. Crie uma variável booleana para cada texto criado, utilizando a ferramenta Caixa para Exibir/Esconder Objetos, e remova essas variáveis da Janela de Visualização
9. Construa um botão com a legenda Atualizar na primeira Janela de Visualização e coloque os seguintes comandos abaixo. Ele servirá para colocar a atividade em sua posição inicial.

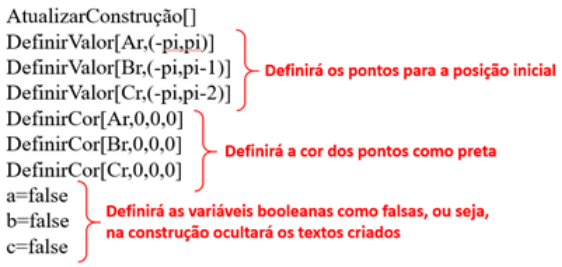


Figura 4.16: Comandos para o botão Atualizar

10. Agora, vamos criar um botão com a legenda Verificar para que o programa faça o teste se a resposta dada pelo(a) usuário(a) estará correta ou não. Além disso, caso esteja errado, o programa vai analisar qual tipo de erro foi cometido, para decidir pelo *feedback* apropriado. Os comandos que devem ser inseridos são:

    Se[Ar == l3(1) && Br == l3(2) && Cr == l3(3), DefinirValor[a,true], DefinirValor[a,false]] - Se os três pontos estiverem corretos, ou seja, o ponto Ar for igual ao primeiro ponto da lista 3 etc. vamos definir o valor do valor booleano a (relacionado ao texto 4) como verdadeiro (irá aparecer), caso contrário, deixará o valor de a como falso.

    Se[Ar != l3(1) && Ar == (y(l3(1)), x(l3(1)))||Br != l3(2) && Br == (y(l3(2)), x(l3(2))) || Cr != l3(3)&& Cr == (y(l3(3)), x(l3(3))),DefinirValor[b,true], DefinirValor[b,false]] - Esse comando fará a análise de cada resposta observando se o ponto é diferente do que deveria ser e se há a troca das coordenadas. Por exemplo, estamos analisando se a resposta do ponto A (Ar) é igual à resposta que ele deveria ter (l3(1)). Além disso, se a resposta é igual a (y(l3(1)), x(l3(1))). Isto é,

    Ar != l3(1) && Ar == (y(l3(1)), x(l3(1)))

    Essa análise é feita para cada ponto. Se acontecer uma dessas coisas (por isso que utilizamos o OU para conectá-los), o texto 5 aparecerá (ele foi conectado à variável b)

    Se[Ar != l3(1)&& Ar!=(-pi,pi) || Br != l3(2) && Br!=(-pi,pi-1) || Cr!=(-pi,pi-2) && Cr != l3(3),DefinirValor[c,true],DefinirValor[c,false]] - Esse comando avaliará somente se cada resposta está errada com os pontos corretos. Observe que inserimos o fato de tais respostas não serem iguais aos pontos que os posicionamos inicialmente. Dessa forma, o texto 6 não aparecerá caso os pontos não forem removidos do lugar.

    Se[Ar!=l3(1),DefinirCor[Ar,1,0,0],DefinirCor[Ar,0,0,1]]

    Se[Br!=l3(2),DefinirCor[Br,1,0,0],DefinirCor[Br,0,0,1]]

    Se[Cr!=l3(3),DefinirCor[Cr,1,0,0],DefinirCor[Cr,0,0,1]]

    Os três comandos acima mudarão a cor dos pontos para vermelho (1,0,0) caso eles sejam diferentes dos pontos corretos. Caso estejam corretos, eles ficarão com a cor azul (0,0,1).

11. Agora, na segunda Janela de Visualização, vamos deixar a malha fixa. Isso servirá para que o(a) usuário(a) só possa posicionar o ponto em valores inteiros (ajuda, principalmente, quem vai utilizar o celular para resolver a atividade)

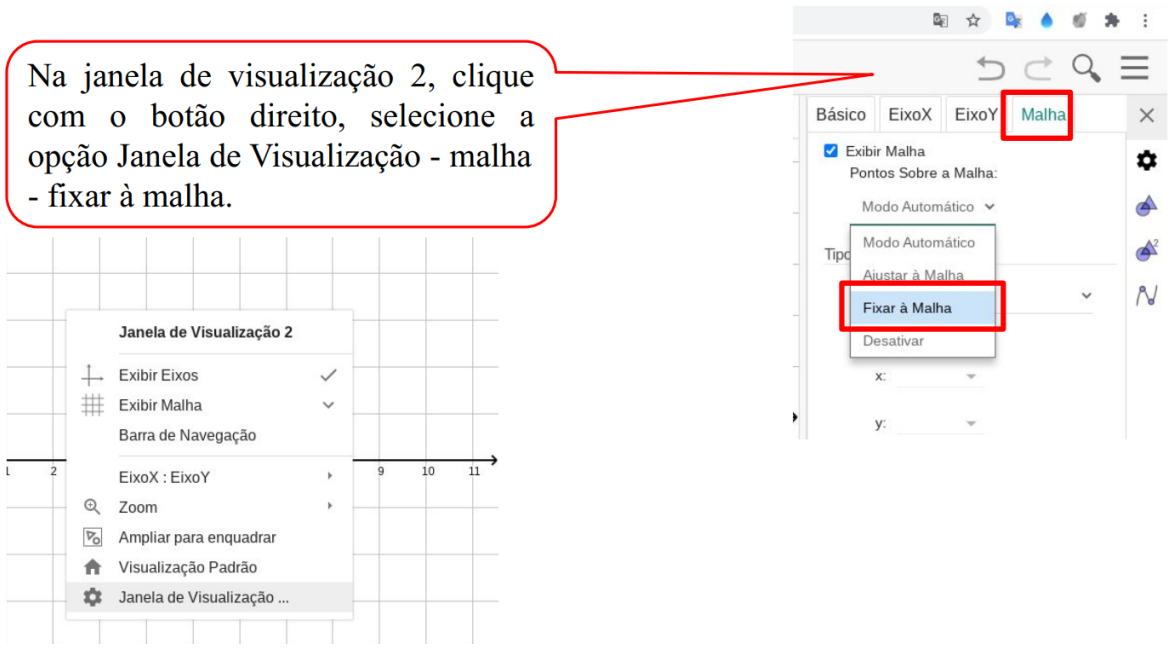


Figura 4.17: Fixar malha

12. O próximo passo é configurar que o botão Verificar só apareça quando o(a) usuário(a) remover os três pontos da posição inicial. Para isso, na aba Avançado, coloque o código abaixo:

    Ar != (-pi, pi) && Br != (-pi, pi - 1) && Cr != (-pi, pi - 2)

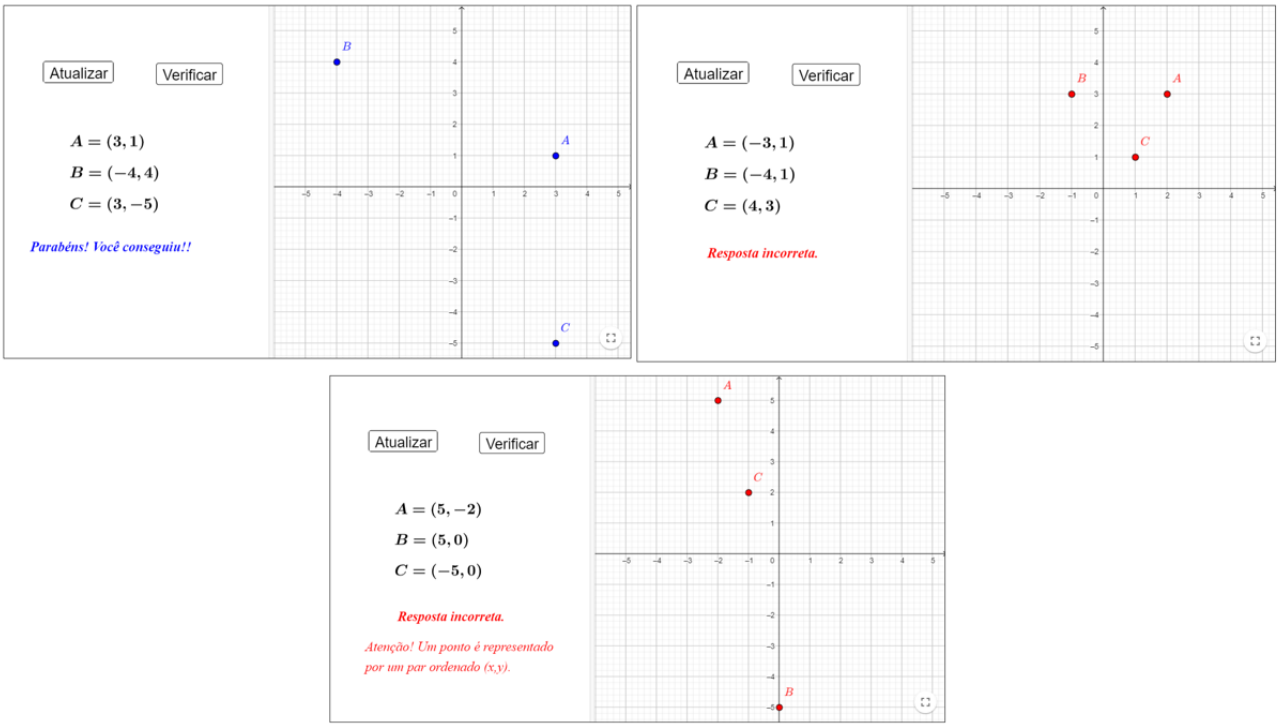


Figura 4.18: Três possibilidades de *feedback*

4.3 Atividade 3 - Função do primeiro grau

Essa terceira atividade apresenta uma função do primeiro grau para o(a) usuário(a) e pede que sejam respondidos alguns valores sobre a função. À medida que os valores respondidos estão corretos, uma tabela dinâmica é apresentada, e os respectivos pontos são mostrados na outra Janela de Visualização. Os *feedbacks* que serão mostrados são:

1) Quando a resposta for incorreta, aparecerá o texto: $f(n) \neq$ resposta dada!
2) Quando for finalizada a tabela, aparecerá o texto: Parabéns!! e o gráfico da função.

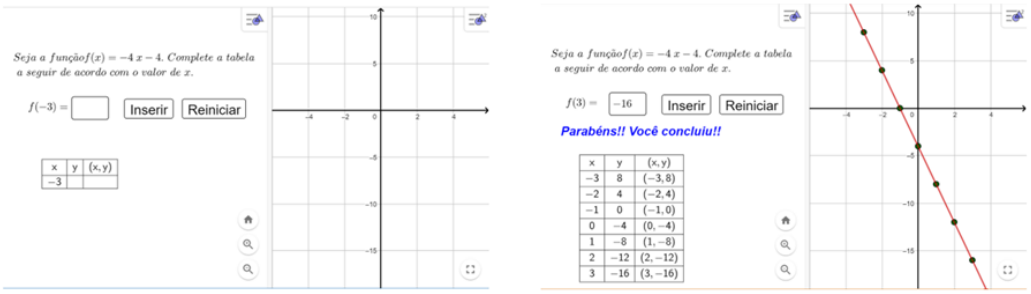


Figura 4.19: Atividade 3

Para a construção siga os passos abaixo:

1. Primeiramente, vamos criar a função do primeiro grau $f(x) = ax + b$ aleatoriamente. Para isso, vamos criar uma lista limitada de -5 até 5. E escolhermos aleatoriamente os valores de a ($a \neq 0$) e b.

   l1=Sequência(-5,5)

   a=EscolherElementoAleatoriamente(Remover(l1,{0})) - nesse comando, perceba que estamos removendo o 0 da lista 1.

   b=EscolherElementoAleatoriamente(l1)

   f(x)=Polinômio(a*x+b)

2. Insira o texto na primeira janela de visualização:

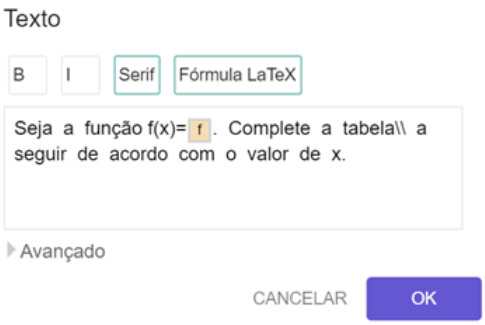


Figura 4.20: Texto 1

3. Agora, crie o controle deslizante n, $n \in \{-3, -2, -1, 0, 1, 2, 3\}$, para os valores de x que serão utilizados na função. E coloque o texto, que ficará antes do campo de entrada, na primeira janela de visualização. Observe que estamos anexando o objeto n no texto.

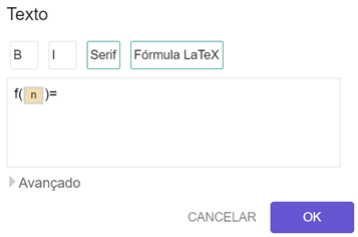


Figura 4.21: Texto 2

4. Vamos preparar o ambiente de resposta do(a) usuário(a). Para isso, crie uma variável **ry=?** que guardará a resposta digitada no campo de entrada que também deverá ser criado. Deixe esse campo de entrada sem Legenda e, nas configurações, sem exibir o rótulo. Se achar necessário, diminua o tamanho do campo de entrada.

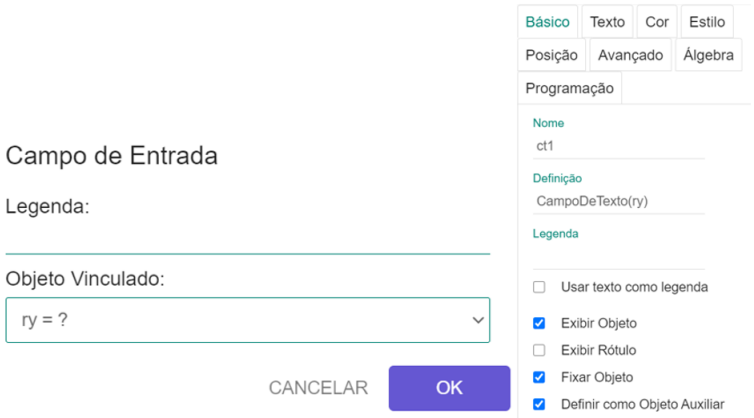


Figura 4.22: Campo de entrada para a resposta

5. Crie um botão com a legenda **Inserir**. Na parte final da construção faremos as devidas programações.

6. Agora, vamos preparar a tabela de texto. Para isso, precisaremos criar três listas que servirão para cada coluna da tabela.

   Para a primeira coluna:

   l2={"x"} - criará o texto x

   l3=Sequência(-3,n) - criará os valores de x (observe que está dependendo de n)

   l4=Concatenar(l2, l3) - juntará as duas listas acima. Será essa lista que irá compor a tabela

   Para a segunda coluna:

   l5={"y"}

   l6={ } - essa lista vazia será configurada no botão Inserir

   l7=Concatenar(l5, l6) - lista que irá compor a tabela

   Para a terceira coluna:

   l8={"(x,y)"}

   l9={ } - essa lista vazia será configurada no botão Inserir. Mas você já pode configurá-la para aparecer somente na segunda Janela de Visualização.

l10=Concatenar({l8, l9}) - lista que irá compor a tabela

Digite no campo de entrada o comando **TabelaDeTexto(l4,l7,l10,“ |_vc ”)** para criar a tabela de texto.

7. Nesse momento, vamos configurar o botão Inserir para que, caso a resposta esteja correta, a tabela seja atualizada. Veja, na imagem da esquerda, o primeiro valor de x na tabela é o -3 (que foi o valor de n que começamos). Quando digitamos o valor correto para f(-3) e clicamos em Inserir, automaticamente, a primeira linha da tabela é atualizada, e uma nova linha é inserida.

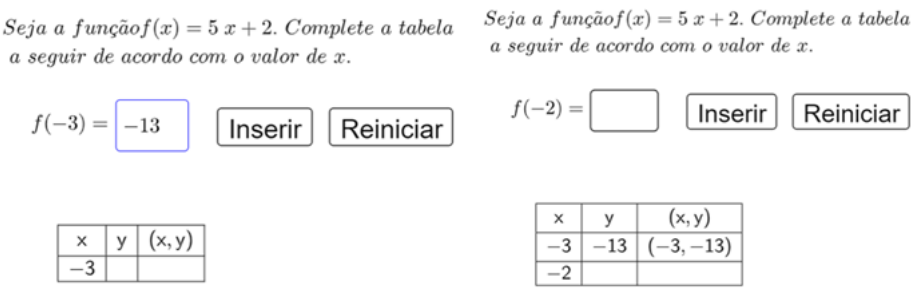


Figura 4.23: Valor digitado inserido na tabela

Para isso, vá até nas configurações do botão Inserir e coloque os comandos abaixo na aba **Ao Clicar** da aba Programação.

Se[ry==f(n),DefinirValor[l6,Anexar[l6,ry]]] - esse comando vai atualizar a lista l6 (que é a lista com os valores do y), anexando a ela o valor ry digitado no campo de entrada. Perceba que foi usado um SE, ou seja, a lista l6 só vai ser atualizada se ry=f(n).

Se[ry==f(n),DefinirValor[l9,Anexar[l9,(n,ry)]]] - seguindo a linha de raciocínio acima, esse comando irá atualizar a lista l9 acrescentando o ponto (n,ry).

Se[ry==f(n),DefinirValor[n,n+1]] - esse comando irá atualizar o valor de n para (n+1). Além da tabela ser atualizada, o texto criado anteriormente que fica do lado do campo de entrada também será atualizado.

Se[ry==f(n-1),DefinirValor[ct1,“”]] - esse comando tem como objetivo apagar o que foi digitado previamente. Observe na imagem acima que, ao ter acertado o valor de f(n), quando o valor de n atualizou para n+1, o campo de entrada foi limpado. E por que utilizamos o f(n-1)? No comando anterior, programamos para que o valor de n fosse mudado para n + 1. Então, não teríamos como comparar o ry com o f(n).

8. Antes de iniciarmos a parte de *feedback*, vamos criar um botão **Reiniciar**. Nele, atualizaremos a função, voltaremos o n para -3 e limparemos a tabela. Para isso, digite o comando abaixo na aba **Ao Clicar** nas configurações do botão.

   AtualizarConstrução()

   DefinirValor(ct1,“”) - limpará o campo de entrada

   DefinirValor(n,-3) - colocará o n para -3

   DefinirValor(l6,{}) - limpa a lista com os valores de y

   DefinirValor(l9,{}) - limpa a lista com os pontos.

9. Agora, vamos configurar os *feedbacks*. Primeiro, crie o texto abaixo com os comandos na aba **Avançado**:

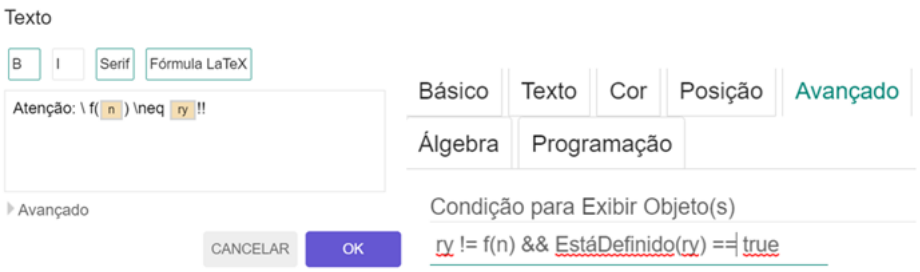


Figura 4.24: Primeiro *feedback*

Esse texto aparecerá quando o que o valor digitado não for o valor de f(n) , ou seja, **ry != f(n)**). E quanto o valor ry existir (**EstáDefinido(ry)==true**).

10. O segundo texto será o **PARABÉNS!!** Você Concluiu!! e deverá aparecer só quando toda a tabela estiver preenchida, ou seja, quando o n for 3 e o valor de f(3) esteja correto. Então, configure para que o texto apareça com a programação **n == 3 && f(3) == ry**

11. O terceiro *feedback* é o gráfico da função aparecer quando toda a atividade for concluída. Então, use a mesma programação que foi usada para o texto anterior.

12. Uma possibilidade que podemos considerar é a de programarmos a segunda tela de visualização de forma automática, ou seja, ela se ajustar dependendo da função que o GeoGebra escolher. Por exemplo, se a função for f(x)=2x+5, temos que f(3)=11 e esse ponto pode não aparecer no gráfico. Então, o(a) usuário(a) teria que ajustar o *zoom* do gráfico toda vez que uma nova função fosse escolhida para poder visualizar todos os pontos (n,f(n)), $n \in [-3, 3]$. Para evitar isso, podemos ir nas configurações da segunda Janela de Visualização e, na aba **Básico**, configuramos os valores máximo e mínimo para os eixos como:

    x Mín: -6

    x Máx: 6

    y Mín: Se(a < 0, f(3) - 3, f(-3) - 3)

    y Máx: Se(a < 0, f(-3) + 3, f(3) + 3)

4.4 Atividade 4 - Círculos no triângulo

Apresentamos a seguir como criar um applet que foi desenvolvido a partir de um material do *site* Portal da Obmep (`https://portaldaobmep.impa.br/index.php/modulo/ver?modulo=124`), cujo objetivo é colorir 6 círculos no triângulo de modo que dois círculos de mesma cor nunca se toquem.

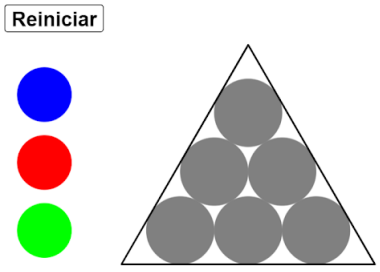


Figura 4.25: Atividade 4

O *Applet* funciona da seguinte maneira: o usuário deverá clicar em um dos círculos laterais para escolher a cor (azul, vermelha ou verde) e depois em um dos círculos internos do triângulo. Ao clicar em cada círculo lateral, haverá uma mudança no valor das variáveis P, Q e R que irá definir a cor dos círculos internos do triângulo. Por exemplo, ao clicar no círculo p, será atribuído o valor 1 à variável P e 0 para as variáveis Q e R. Assim, quando algum círculo interno do triângulo for clicado ficará com a mesma cor do círculo p (azul). Cada círculo interno estará associado às variáveis C1, C2, C3, C4, C5 e C6 que podem assumir os valores: 1 (Azul), 4 (Vermelho) ou 10 (Verde) quando clicados. Desse modo, para a solução do desafio, a soma das variáveis de cada grupo de três círculos empilhados deverá ser 15. Note que as possíveis somas são: 3, 6, 9, 12, 15, 18, 21, 24 e 30.

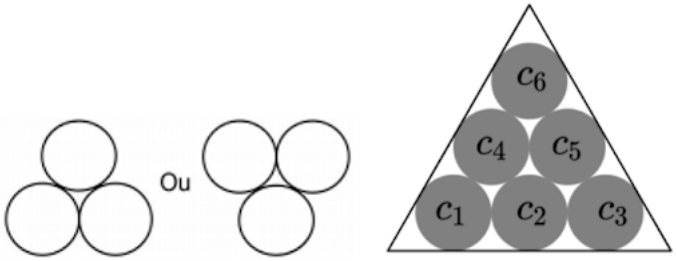


Figura 4.26: Posição para os círculos

Como há 4 grupos de três círculos empilhados, o *applet* verificará as somas N1, N2, N3 e N4. Assim, se N1 = N2 = N3 = N4 = 15, aparecerá a mensagem: Parabéns! Você conseguiu!!. Se as variáveis C1, C2, C3, C4, C5 e C6 forem todas diferente de 0 e alguma das variáveis N1, N2, N3 e N4 for diferente de 15, aparecerá a mensagem: Configuração incorreta. Tente novamente!.

1. Com o comando **Círculo(<Ponto>, <Raio>)** vamos construir os 6 círculos cinzas (Tabela 4.2) os 3 círculos laterais (Tabela 4.3). Na segunda coluna da tabela há algumas configurações que devem ser consideradas para os objetos.

Tabela 4.2: Círculos internos ao triângulo

Comando	Configurações para todos
c_1= Círculo((1,1),1) c_2= Círculo((3,1),1) c_3= Círculo((5,1),1) c_4=Círculo((2,sqrt(3)+1),1) c_5=Círculo((4,sqrt(3)+1),1) c_6= Círculo((3,2 sqrt(3)+1),1)	Cor: cinza Transparência: 100 Espessura da linha:1 Opacidade do traço: 100

Tabela 4.3: Círculos laterais

Comando	Configurações para todos
p= Círculo((-3,5),0.8) q= Círculo((-3,3),0.8) r= Círculo((-3,1),0.8)	Cor: p - Azul, q - vermelho e r - verde Transparência: 100 Espessura da linha:1 Opacidade do traço: 100

2. Com o comando **Polígono(<Ponto>, <Ponto>, <Número de Vértices>)** construa o triângulo como indica a Tabela 4.4:

Tabela 4.4: Construção do triângulo

Comando	Configurações
t=Polígono((1-sqrt(3),0),(5+sqrt(3),0),3)	Cor: Preto Transparência: 0 Espessura da linha:5 Opacidade do traço: 100 Camada: 1

3. Crie as seguintes variáveis que constam na Tabela 4.5 digitando no campo de entrada:

Tabela 4.5: Criação das variáveis

P=1	C1=10	N1=C1+C2+C4
Q=1	C2=10	N2=C2+C4+C5
R=1	C3=10	N3=C2+C3+C5
	C4=10	N4=C4+C5+C6
	C5=10	
	C6=10	

4. **Programação dos círculos laterais**: aqui iremos programar que, ao clicar em cada círculo, haverá uma mudança no valor das variáveis P, Q e R. Por exemplo, ao clicar no círculo p, será atribuído o valor 1 à variável P e 0 para as variáveis Q e R. Para isso, siga os comandos da Tabela 4.6

Tabela 4.6: Programação dos círculos laterais

Círculos laterais	Configurações <> Programação <> Ao Clicar:
Círculo p: Azul	DefinirValor[P,1] DefinirValor[Q,0] DefinirValor[R,0]
Círculo q: Vermelho	DefinirValor[P,0] DefinirValor[Q,1] DefinirValor[R,0]
Círculo r: Verde	DefinirValor[P,0] DefinirValor[Q,0] DefinirValor[R,1]

5. **Programação dos círculos internos do triângulo**: Agora, vamos programar que, quando o círculo interno for clicado, altere sua cor de acordo com a cor do círculo lateral clicado anteriormente, lembre-se que a nova cor dependerá dos valores de P, Q e R. Além disso, haverá mudança no valor das variáveis C1, C2, C3, C4, C5 e C6 que também depende dos valores de P, Q e R. Por exemplo, se P=1, a cor do círculo mudará para azul (lembre-se que a variável P está relacionada com o círculo lateral azul) e o valor de C1 será atualizado para 1. Para isso, siga os passos da Tabela 4.7.

Tabela 4.7: Programação dos círculos internos do triângulo

Círculos internos	Configurações <> Programação <> Ao Clicar:
c_1	Se[P==1,DefinirCor[c_1,"blue"]] Se[P==1,DefinirValor[C1,1]] Se[Q==1,DefinirCor[c_1,"red"]] Se[Q==1,DefinirValor[C1,4]] Se[R==1,DefinirCor[c_1,"green"]] Se[R==1,DefinirValor[C1,10]]
c_2	Se[P==1,DefinirCor[c_2,"blue"]] Se[Q==1,DefinirCor[c_2,"red"]] Se[R==1,DefinirCor[c_2,"green"]] Se[P==1,DefinirValor[C2,1]] Se[Q==1,DefinirValor[C2,4]] Se[R==1,DefinirValor[C2,10]]
c_3	Se[P==1,DefinirCor[c_3,"blue"]] Se[Q==1,DefinirCor[c_3,"red"]] Se[R==1,DefinirCor[c_3,"green"]] Se[P==1,DefinirValor[C3,1]] Se[Q==1,DefinirValor[C3,4]] Se[R==1,DefinirValor[C3,10]]
c_4	Se[P==1,DefinirCor[c_4,"blue"]] Se[Q==1,DefinirCor[c_4,"red"]] Se[R==1,DefinirCor[c_4,"green"]] Se[P==1,DefinirValor[C4,1]] Se[Q==1,DefinirValor[C4,4]] Se[R==1,DefinirValor[C4,10]]

6. Crie os seguintes textos para o *feedback* automático:

Tabela 4.8: Programação dos círculos internos do triângulo

Textos	Configurações <> Avançado <> Condição para exibir objeto(s)
Parabéns! Você concluiu!!	N1 == N2 == N3 == N4 == 15
Configuração incorreta. Tente novamente!	$(\neg(0 \in \{C1, C2, C3, C4, C5, C6\})) \land (N1 \neq 15 \lor N2 \neq 15 \lor N3 \neq 15 \lor N4 \neq 15)$

Um dos resultados pode ser como na Figura 4.27 abaixo:

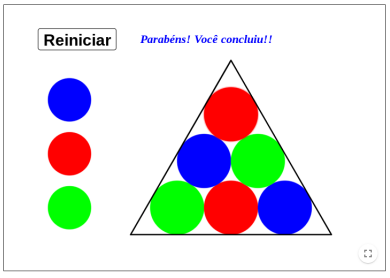


Figura 4.27: *Feedback* ao acertar

7. Crie um botão com a legenda **Reiniciar** com a seguinte programação:

Tabela 4.9: Programação para os *feedbacks*

Configurações <> Programação <> Ao Clicar:	
DefinirValor[P,0]	DefinirCor[c_6,“gray”]
DefinirValor[Q,0]	DefinirValor[C1,0]
DefinirValor[R,0]	DefinirValor[C2,0]
DefinirCor[c_1,“gray”]	DefinirValor[C3,0]
DefinirCor[c_2,“gray”]	DefinirValor[C4,0]
DefinirCor[c_3,“gray”]	DefinirValor[C5,0]
DefinirCor[c_4,“gray”]	DefinirValor[C6,0]
DefinirCor[c_5,“gray”]	

Capítulo 5

Considerações Finais

Não é nossa pretensão apresentar todas as ferramentas e comandos que existem no GeoGebra. Acessando a lista de comandos no manual do GeoGebra (`https://wiki.geogebra.org/en/Category:Commands`) conseguimos ver que há mais de 500 comandos, então seria praticamente impossível apresentar um material com tais expectativas. Mas acreditamos que conseguimos apresentar, como dito na introdução, as principais ferramentas e os principais comandos para criar atividades com a possibilidade de *feedback* automático.

O GeoGebra consegue nos dar a possibilidade de, além da questão dinâmica que é a alma do próprio programa, criarmos atividades, das mais simples às mais complexas. Além disso, como vimos, podemos programar a atividade de tal forma que o(a) usuário(a) pode interagir com a aplicação, sem a necessidade de um(a) docente. Percebam também que a escolha do *feedback* vai depender muito da compreensão do(a) professor(a) em prever quais os erros mais comuns que os alunos cometem e inserir isso na atividade. Por isso que pensar a atividade antes é muito importante, sempre escrevendo o que se quer nela, quais serão as possibilidades de *feedback* e em quais situações cada *feedback* irá aparecer. Depois que o(a) criador(a) da atividade faz esse mapa geral da atividade, a implementação no GeoGebra com toda sua programação ficará mais fácil.

Para finalizar, queremos estimular aos professores que pensem e criem suas próprias atividades. Há fóruns do GeoGebra em várias redes sociais, onde os usuários se ajudam e tiram as dúvidas. Sabemos que, cada dia mais, a tecnologia torna-se ferramenta imprescindível para o ensino e aprendizagem da matemática, e acreditamos que o GeoGebra é uma peça importantíssima para chegarmos ao nosso objetivo.

Referências Bibliográficas

[1] ABAR, Celina Aparecida Almeida Pereira; DOS SANTOS, José Manuel Dos Santos. O GeoGebra como Estratégia para Ensino Remoto: Criando Atividades com Feedback Automático. Disponível em: <https://ined.ese.ipp.pt/sites/default/files/2020-12/Projeto_PIPRINT..pdf >. Acesso em: 25 de abr. de 2021.

[2] AMADO, Nélia; SANCHEZ, Juan; PINTO, Jorge. A Utilização do Geogebra na Demonstração Matemática em Sala de Aula: o estudo da reta de Euler. Bolema: Boletim de Educação Matemática, v. 29, n. 52, p. 637-657, 2015.

[3] ARBAIN, Nazihatulhasanah; SHUKOR, Nurbiha A. The effects of GeoGebra on students achievement. Procedia-Social and Behavioral Sciences, v. 172, p. 208-214, 2015.

[4] BRASIL. Base Nacional Comum Curricular. Brasília: Ministério da Educação. 2018.

[5] DE OLIVEIRA, Vania Doneda et al. Uso do GeoGebra à Luz da Teoria dos Registros de Representação Semiótica. In: Anais dos Workshops do Congresso Brasileiro de Informática na Educação. p. 614, 2019.

III SIMPÓSIO DA FORMAÇÃO
DO PROFESSOR DE MATEMÁTICA
DA REGIÃO NORTE

Realização e Organização

ANPMat
Associação Nacional dos Professores
de Matemática na Educação Básica

ISBN: 978-65-88013-12-0