

Laboratório de Extensão

Disciplina: MÍDIAS DIGITAIS E METODOLOGIAS ATIVAS
Autor/Aluno: Alexandre Cardoso Garcia Leite
Professor/Anuente:

Juazeiro-BA, 22/08/2024
Aluno: Alexandre Cardoso Garcia Leite
Juazeiro-BA, 22/08/2024
Professor ou Gestor:: Escola: Colégio Modelo Luís Eduardo Magalhães

INTRODUÇÃO/DIAGNÓSTICO E TEORIZAÇÃO

Vivemos num mundo cercado por tecnologias da informação que estão substituindo os trabalhos humanos tediosos e repetitivos cada vez mais. Para a educação no século XXI, é necessário tornar os estudantes cientes de duas coisas: eles devem entender o que está por trás da tecnologia (independente da área que eles forem atuar), eles devem entender que eles, provavelmente, não serão mais apenas trabalhadores, mas trabalhadores e sempre estudantes ao mesmo tempo.

Situação-problema

Mostrar ao aluno que por trás de toda tecnologia há um raciocínio-lógico-matemático.

Demanda sociocomunitária e motivação acadêmica

Não temos ainda no currículo das escolas no Brasil uma disciplina específica para ensinar Ciência da Computação, pelo menos, desde o ensino médio. Isso é um problema, porque os algoritmos usados nas mais diversas áreas, mas principalmente nas engenharias, estão cada vez mais complexos. Sendo assim, torna-se necessário chegar na Universidade já com uma bagagem teórica e prática e, assim, acelerar o aprendizado quando se chegar no nível superior.

Objetivos a serem alcançados em relação à situação-problema identificada

Neste trabalho, mostraremos um algoritmo simples para desenhar polígonos regulares. Explorando um pouco de geometria plana básica e mostrando como usar na prática os conceitos de geometria plana para desenhar polígonos regulares.

PROJETO

Mostraremos através da linguagem de programação Python, um algoritmo de poucas linhas de código, necessário para desenhar um polígono centralizado na tela do computador e inscrito numa circunferência de raio determinado.

Nessa demonstração, exploraremos alguns conceitos básicos de como uma linguagem de programação de computadores funciona.

Usaremos o ambiente de desenvolvimento integrado IDLE, por ser mais leve e pode ser utilizado em um computador mesmo com poucos recursos de memória, disco e CPU. Este ambiente de desenvolvimento é o mais simples e já vem junto com a instalação do Python, mesmo sendo leve, nos permite demonstrar, passo a passo como os cálculos são feitos e como as estruturas de código são executadas.

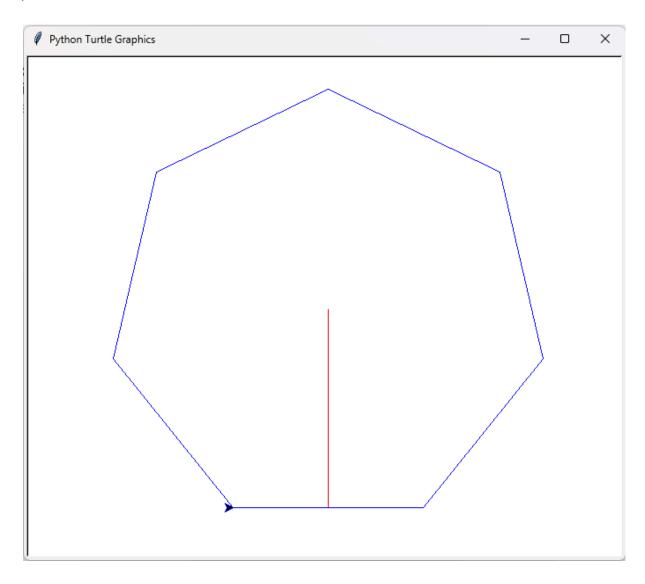
Algoritmo V2 (Input):

```
# -*- coding: utf-8 -*-
Created on Tue Jul 16 04:27:06 2024
Última Atualização: 19/08/2024
@author: aleitebr
Script Name: PoligonosRegulares_v2.py
# Importa bibliotecas necessárias para a execução do algoritmo
import turtle
import math
# Define o valor do raio da circunferência
RAIO = 250
# Instancia um objeto da Classe turtle.Pen()
t = turtle.Pen()
t.speed(1) # Seta a velocidade que o desenho vai ser executado
t.pencolor('blue') # Define a cor da caneta para 'azul'
turtle.bgcolor('white') # Define o fundo da tela para branco
lados_poligono = 2 # não existe polígonos com menos de 3 lados
while (lados_poligono < 3): # enquanto o usuário não digitar um valor</pre>
                            # n >= 3, continua perguntando
                            # por um valor válido
     # pergunta ao usuário o número de lados do polígono que ele
     # quer desenhar
     lados_poligono = int(input('Digite o número de lados para o polígono
(n >= 3)?')
    if lados_poligono < 3: # se lados do poligno for menor que 3, envia</pre>
                          # uma mensagem para o usuário e refaz a pergunta
        print('Erro: Digite num número maior que 2.')
# Calcula o ângulo interno do polígono reguar, note que a soma dos ângulos
# internos podem ser divididas pelo numero de lados poligono
# por cause que estamos tratando de polígonos regulares
angulo_interno = (lados_poligono - 2) * 180 / lados_poligono
```

```
# chamamos de angulo central o ângulo que a cincunferência faz com cada
# lado do polígono
angulo_central = 360 / lados_poligono
# aplica a lei dos cossenos para calcular o comp lado poligono
comp_lado_poligono = math.sqrt(2*RAIO**2-RAIO**2*math.cos(angulo_central /
180 * math.pi))
# para se obter o apótema é necessário dividir o polígono em triângulos e
# calcular a altura
apotema = ( comp_lado_poligono / 2 ) / math.tan( angulo_central / 2 / 180
* math.pi)
# o módulo "turtle" incia com uma seta na posição central
# da tela indicando para a direita
t.right(90) # move a direita 90º para a direita
#t.penup() # levanta a caneta para desenhar o apótema
t.pencolor('red')
t.forward( apotema ) # desenha apótema
t.right(90) # gira novamente 90º para a direta
t.pencolor('blue') # retorna a cor azul para desenhar o polígono
t.forward( comp lado poligono / 2 )
t.right(180) # move a seta 180º
# baixa a caneta para desenhar o polígono
#t.pendown()
# enquato x estiver entre [0, número lados poligono] desenha as arestas
for x in range( lados_poligono ):
   t.forward( comp_lado_poligono )
   t.left( 180 - angulo_interno ) # move para a esquerda a caneta
                                   # observe que giramos o ângulo
                                   # externo do polígno,
                                    # não o ângulo interno
```

Output Algoritmo V2:

para n = 7,



Algoritmo V4 (Input):

```
# -*- coding: utf-8 -*-
Created on Tue Jul 16 04:27:06 2024
Última Atualização: 19/08/2024
@author: aleitebr
Script Name: PoligonosRegulares_v4.py
Desenha a circunferência que circunscreve o polígono
# Importa bibliotecas necessárias para a execução do algoritmo
import turtle
import math
# Define o valor do raio da circunferência
RAIO = 250
# Instancia um objeto da classe turtle.Pen()
t = turtle.Pen()
t.speed(1) # Seta a velocidade que o desenho vai ser executado
t.pencolor('blue') # Define a cor da caneta para 'azul'
turtle.bgcolor('white') # Define o fundo da tela para branco
lados_poligono = 2 # não existe polígonos com menos de 3 lados
while (lados_poligono < 3): # enquanto o usuário não digitar um valor</pre>
                            # n >= 3, continua perguntando por
                            # um valor válido
    # pergunta ao usuário o número de lados do polígono
    # que ele quer desenhar
    lados_poligono = int(input('Digite o número de lados para o polígono
(n >= 3)?')
    if lados_poligono < 3:# se lados do poligno for menor que 3, envia</pre>
                          # uma mensagem para o usuário e refaz a pergunta
        print('Erro: Digite num número maior que 2.')
lista_dados_poligonos = [60, lados_poligono]
angulo_interno = []
angulo_central = []
comp_lado_poligono = []
apotema = []
```

```
for i in range(2):
  # Calcula o ângulo interno do polígono regular, note que
  # a soma dos ângulos internos podem ser divididas pelo numero de
   # lados poligono por causa que estamos tratando de polígonos regulares
 angulo_interno.append( (lista_dados_poligonos[i] - 2) * 180 /
lista_dados_poligonos[i] )
   # chamamos de ângulo central o ângulo que a cincunferência faz com
   # cada lado do polígono
   angulo_central.append( 360 / lista_dados_poligonos[i] )
   # aplica a lei dos cossenos para calcular o comp_lado_oposto
   comp_lado_poligono.append( math.sqrt((2*RAIO**2 -
2*RAIO**2*math.cos(angulo_central[i] / 180 * math.pi))))
   # para se obter o apótema é necessário dividir o polígono em triângulos
   # e calcular a altura
 apotema.append( ( comp_lado_poligono[i] / 2 ) / math.tan( (
angulo_central[i] / 2 ) / 180 * math.pi) )
posicao inicial = t.pos() # vai para o centro da tela, reinicia posição
inicial de desenho
for i in range(2):
   t.penup() # levanta a caneta
   t.goto(posicao inicial) # retorna a posição inicial
   # o módulo "turtle" incia com uma seta na posição central da tela
   # indicando para a direita, por causa disse devemos girar
   # ela 90º a direita
   t.right(90) # move a direita 90º para a direita
   t.forward( apotema[i] ) # desenha apótema
   t.right(90) # gira novamente 90º para a direta
   t.forward( comp lado poligono[i] / 2 )
   t.right(180) # move a seta 180º
   # baixa a caneta para desenhar o polígono
   t.pendown()
   # enquato x estiver entre [0, número lados poligono] desenha as arestas
   for j in range(lista_dados_poligonos[i]):
      t.forward( comp_lado_poligono[i] )
      t.left( 180 - angulo_interno[i] ) # move para a esquerda a caneta
                                        # observe que giramos o ângulo
                                        # externo do polígno,
                                        # não o ângulo interno
```

Outup Algoritmo V4:

Para n = 5

