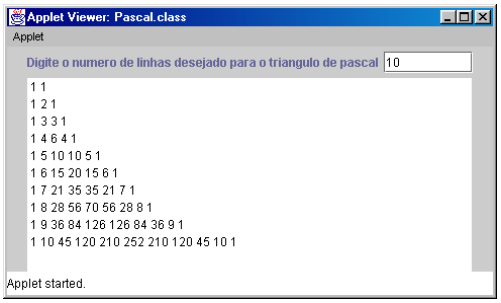
4º Laboratório ECOP15 – 27 e 28 setembro 2021

Utilizar como referência os exemplos:
InitArray.java: Inicializar arrays;
SumArray.java: Soma os elementos de um vetor;
Cap. 7. Java How to Program 10ed;

1ª Tarefa: Escreva um programa em Java que use um array irregular para representar um *Triângulo de Pascal*. O triângulo de Pascal é uma série de séries de valores onde cada valor é obtido somando-se o valor acima (linha anterior) e o valor à esquerda do valor acima. Parte do triângulo de Pascal é mostrado abaixo.

1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1



Apresentar o resultado como na figura acima

Dicas: cada linha *l* do array irregular terá *l* +1 colunas. A primeira coluna de cada linha do array vale 1, e a segunda vale o índice da linha mais um. Cada linha do array é reflexiva, podendo ser lida da direita para a esquerda ou da esquerda para a direita da mesma forma. Assumir que o programa só aceita até a linha 30

2ª Tarefa: Para encontrar uma raiz de um polinômio $p(x)=a_0+a_1x+...+a_nx^n$, ($n \geq 2$), pode-se aplicar o método de Newton, que consiste em refinar uma aproximação inicial x_0 dessa raiz através da expressão:

$$x_{n+1} = x_n - \frac{p(x_n)}{p'(x_n)}$$

onde:
n= 0,1,2,...,
$p'(x)$ é a primeira derivada de $p(x)$.

- Usualmente, repete-se esse refinamento até que $|x_{n+1} - x_n| < \varepsilon$, $\varepsilon \geq 0$, ou até que *m* iterações tenham sido executadas.
- Dados um polinômio $p(x)=a_0+a_1x+...+a_nx^n$, uma aproximação inicial x_0 da raiz de $p(x)$, $\varepsilon \geq 0$ e o número máximo de iterações que devem ser executadas, determine uma aproximação da raiz de $p(x)$ pelo método de Newton.
- Utilize uma função para ler um polinômio.
- Utilize uma função que, dado um polinômio $p(x)$, calcula a derivada $p'(x)$
- Para calcular $p(x_n)$ e $p'(x_n)$ em cada iteração, uma função que calcula o valor de um polinômio em um ponto.

Obs:
- represente o polinômio através de um vetor;
- crie um programa em Java que utilize as funções descritas acima.

3ª Tarefa: Crie um programa em Java que declare e inicialize dois arrays unidimensionais do mesmo tamanho, e calcule o produto escalar deles. Por exemplo, se os dois arrays forem { 9; 2; 6; 7; 0 } e { 1; 4; 5; 9; 2 } o produto escalar será:

9 x 1 + 2 x 4 + 6 x 5 + 7 x 9 + 0 x 2 = 110.

Baseado na implementação do produto escalar descrito nesta questão, crie um método que verifica se dois vetores são ortogonais.

Monte uma interface gráfica (utilizando dois JTextField) com eventos de ação para pegar os valores dos vetores usando o método split da classe String. Rpresentar o produto escalar em um JLabel.

4ª Tarefa:

Crie um aplicativo Java com interface gráfica que tenha dois campos de texto e um botão. O botão deverá disparar uma função que gere dois números aleatórios entre 1 e 6, colocando cada número em um dos campos. Armazene os valores de cada vez que o usuário apertar o botão em um ArrayList e mostre em um label a média dos valores obtidos bem como o desenho de um histograma com a frequência de aparição da soma dos dados.

Classes usadas neste Laboratório:
JPanel
JLabel
JTextField
JButton
JTextArea

```
// opções para pegar o valor da caixa de texto e converter para numero
int linha = Integer.parseInt(e.getActionCommand());
// ou
int linha = Integer.parseInt(texto.getText());

//Alocar a Matriz triagular
int mat[][];
mat = new int [linha][];
for(int i = 0 ; i< mat.length; i++)
      mat[i] = new int [i+2];

// montar a interface de saída.
String triangulo="";
for(int i = 0; i< mat.length; i++){
  for(int j = 0; j< mat[i].length; j++)
      triangulo += " " + mat[i][j];
  triangulo += "\n";
}
display.setText(triangulo);
```