**1. (1,5 pontos)** Faça uma função que receba 3 números inteiros representando o comprimento das arestas de um triângulo, a sua função verifica se os valores lidos podem formar um triangulo e retorna **true** caso seja possível ou **false** caso contrário.

public static Boolean ehTriangulo( int A, int B, int C){...}

**2. (2,0 pontos)** Escreva uma função que imprime a seguinte saída:

9876543210

987654321

98765432

9876543

987654

98765

9876

987

98

9

Na implementação função use dois laços aninhados (um dentro do outro), note que o laço mais externo repete 10 vezes e o laço mais interno não repete um número fixo de vezes, abaixo a declaração da função:

void imprime(){...}

**3. (2,5 pontos)** Escreva uma função que receba por parâmetro um N número natural maior que 1 (não precisa testar, sempre será informado um número > 1), e verifica se o mesmo é primo ou não. Caso o número seja primo é retornado **true** e caso contrário **false**, lembre-se que um número primo é divisível somente por 1 e por ele mesmo. Abaixo a declaração da função:

boolean ehPrimo(int N){...}

1. **(2,0 pontos) Escreva uma função** ehCrescente() **que recebe um vetor de número inteiros, e verifica se os elementos do vetor estão em ordem crescente, função retorna true caso os elementos estejam em ordem crescente e false caso contrário. Abaixo a declaração da função:**boolean ehCrescente( int V[]){...}
2. **(2,0 pontos) Escreva uma função que receba um vetor como parâmetro, a sua função calcula e retorna a maior diferença absoluta entre números armazenados no vetor. Abaixo a declaração da função:**

int maiorABS(int V[]){...}

1