



Universidade Federal de Itajubá

# Atividade Prática 3

COM242 – Sistemas Distribuídos

Alunos: Ygor Salles Aniceto Carvalho      2017014382

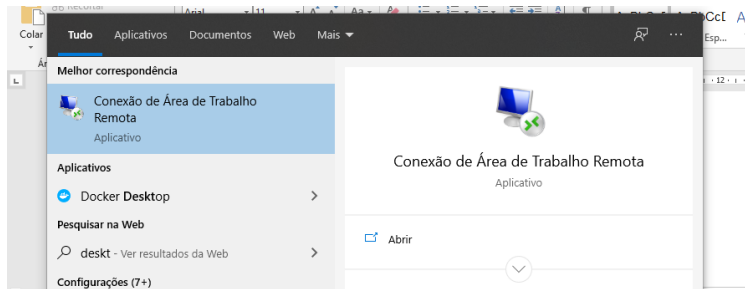
Professora: Rafael de Magalhães Dias Frinhani

Setembro

2020

## CRIANDO UM PROGRAMA QUE REALIZA A FUNÇÃO DO SEGUNDO GRAU DA MATEMÁTICA UTILIZANDO O JAVA RMI

O programa foi testado em duas máquinas, uma assumindo o papel de cliente e a outra do servidor. Para acessar a área de trabalho da máquina do cliente utilizei o aplicativo do windows, desktop remote:



### Intalando o Java:

Para validar se o java está funcionando corretamente executei os seguintes comandos no prompt:

*java*

*javac*

*java -version*

```
Prompt de Comando

C:\Users\Particular>java -version
java version "11.0.8" 2020-07-14 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.8+10-LTS)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.8+10-LTS, mixed mode)

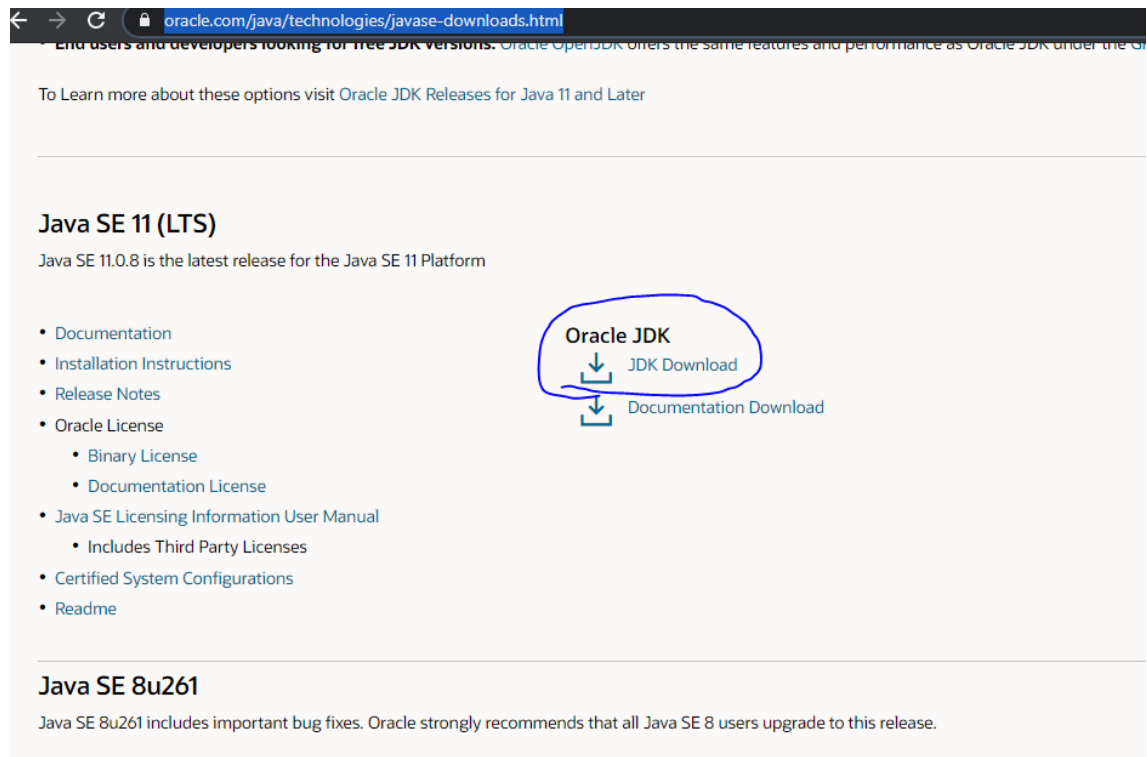
C:\Users\Particular>javac
Usage: javac <options> <source files>
where possible options include:
  @<filename>          Read options and filenames from file
  -Akey[=value]         Options to pass to annotation processors
  --add-modules <module>(<module>)*
                        Root modules to resolve in addition to the initial modules, or all modules
                        on the module path if <module> is ALL-MODULE-PATH.
  --boot-class-path <path>, -bootclasspath <path>
                        Override location of bootstrap class files
```

\*Caso já tenha o java instalado e ao executar esses comandos no CMD não reconheceu o java, desinstale todas as versões do java na sua máquina e depois siga o passo 1º passo.

**1º Passo:** Instalei a penúltima versão do java nas máquinas(cliente e servidor). De preferência a penúltima porque geralmente a última ainda pode não estar completa. Segue o site para instalação:

<https://www.oracle.com/java/technologies/javase-downloads.html>

Foi instalado o JDK 11 – 64 bits na minha máquina:



caso a máquina seja de arquitetura 32 bits nesse mesmo site acima terá uma versão 32 bits.

## 2º Passo:

Após a instalação do JDK, criei um diretório na área de trabalho da máquina servidora onde foi armazenando os códigos do servidor. O código pode ser escrito através de uma IDE NetBeans, Eclipse ou por um editor de texto, notePad++ por exemplo. Nos códigos foram importados as bibliotecas do JavaRMI para que fizesse a conexão cliente/servidor, conforme os prints abaixo:

- ArrancaServidor.java - classe:

```
C:\Users\Particular\Desktop\JavaRMI\ServidorCalc\ArrancaServidor.java - Notepad++
Arquivo  Editar  Localizar  Visualizar  Formatar  Linguagem  Configurações  Ferramentas  Macro  Executar  Plugins  Janela  ?

ArrancaServidor.java  InterfaceServidorMat.java  ServidorMat.java

1  /* COM242 - SISTEMAS DISTRIBUIDOS
2  RMI - Exemplo de implementação.
3  Programa que utiliza funções remotas para realizar operacoes matematicas.
4  14/04/2020
5  */
6
7  import java.rmi.*;
8  import java.net.*;
9  import java.rmi.registry.Registry;
10
11 // Classe que registra o servidor da aplicação junto ao servidor de nomes
12 public class ArrancaServidor
13 {
14     public static void main(String argv[])
15     {
16         try
17         {
18             System.out.println("Subindo servidor...");
19             InetAddress IP = InetAddress.getLocalHost();
20             System.setProperty("java.rmi.server.hostname", IP.getHostAddress());
21             Registry r = java.rmi.registry.LocateRegistry.createRegistry(1099); // Registra a porta da aplicação
22             Naming.rebind("ServidorMat_1", new ServidorMat()); // Associa o servidor a um nome para o acesso do cliente
23         }
24         catch (Exception e)
25         {
26             System.out.println("Ocorreu um problema no arranque do servidor.\n"+e.toString());
27         }
28     }
29 }
```

Foi tratado o erro caso ocorra algum problema para executar o servidor.

- InterfaceServidorMat.java - interface:

```
C:\Users\Particular\Desktop\JavaRMI\ServidorCalc\InterfaceServidorMat.java - Notepad++
Arquivo  Editar  Localizar  Visualizar  Formatar  Linguagem  Configurações  Ferramentas  Macro  Executar  Plugins  Janela  ?

ArrancaServidor.java  InterfaceServidorMat.java  ServidorMat.java

1  /* COM242 - SISTEMAS DISTRIBUIDOS
2  RMI - Exemplo de implementação.
3  Programa que utiliza funções remotas para realizar operacoes matematicas.
4  14/04/2020
5  */
6
7  import java.rmi.*;
8
9 // Definição da interface que descreve os objetos remotos que poderao ser acessados pelo cliente
10 public interface InterfaceServidorMat extends Remote
11 {
12     public String funcaoSegundoGrau(double a, double b, double c) throws RemoteException;
13 }
14 }
```

- ServidorMat.java - classe

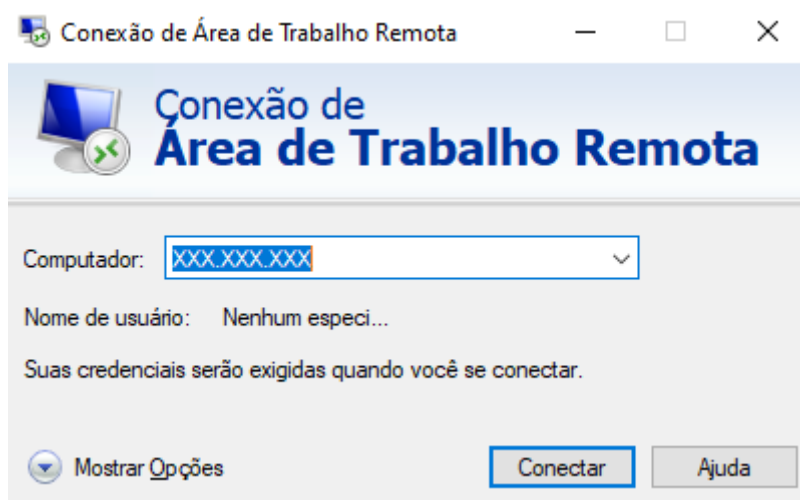
```
*C:\Users\Particular\Desktop\JavaRMI\ServidorCalc\ServidorMat.java - Notepad++
Arquivo Editar Localizar Visualizar Formatar Linguagem Configurações Ferramentas Macro Executar Plugins Janela ?

ArancaServidor.java InterfaceServidorMat.java ServidorMat.java
1 import static java.lang.Math.sqrt;
2 import java.rmi.*;
3 import java.rmi.server.*;
4
5 // Classe no servidor que implementa os métodos remotos
6 public class ServidorMat extends UnicastRemoteObject implements InterfaceServidorMat
7 {
8     public ServidorMat() throws RemoteException
9     {
10         System.out.println("Novo Servidor instanciado...");
11     }
12
13     public String funcaoSegundoGrau(double a, double b, double c) throws RemoteException
14     {
15         float delta, x1, x2;
16
17         System.out.println("Valores recebidos do cliente: a = " + a + " b = " + b + " c = " + c);
18
19         if (a != 0) {
20             delta = (float) (Math.pow(b,2) - 4*a*c);
21
22             if (delta>0){
23                 x1 = (float) ((-b +Math.sqrt(delta))/(2*a));
24                 x2 = (float) ((-b -Math.sqrt(delta))/(2*a));
25                 return "Equação possui duas raízes reais: \n X': "+x1+"\n X'': "+x2;
26             }
27             else if (delta==0){
28                 x1 = (float) ((-b +sqrt(delta))/(2*a));
29                 return "\nEquação possui uma raiz real: \n X: "+x1;
30             }
31             else
32                 return "\nEquação não possui raízes reais!\n";
33         }
34         else
35             return "\nA deve ser diferente de Zero!";
36     }
37 }
```

No arquivo ServidorMat.java se encontra o código que faz a operação de Báskara, tendo como retorno uma string e tratando todos os tipos possíveis de retorno que uma função do segundo grau possa ter, de maneira interativa.

### 3º Passo:

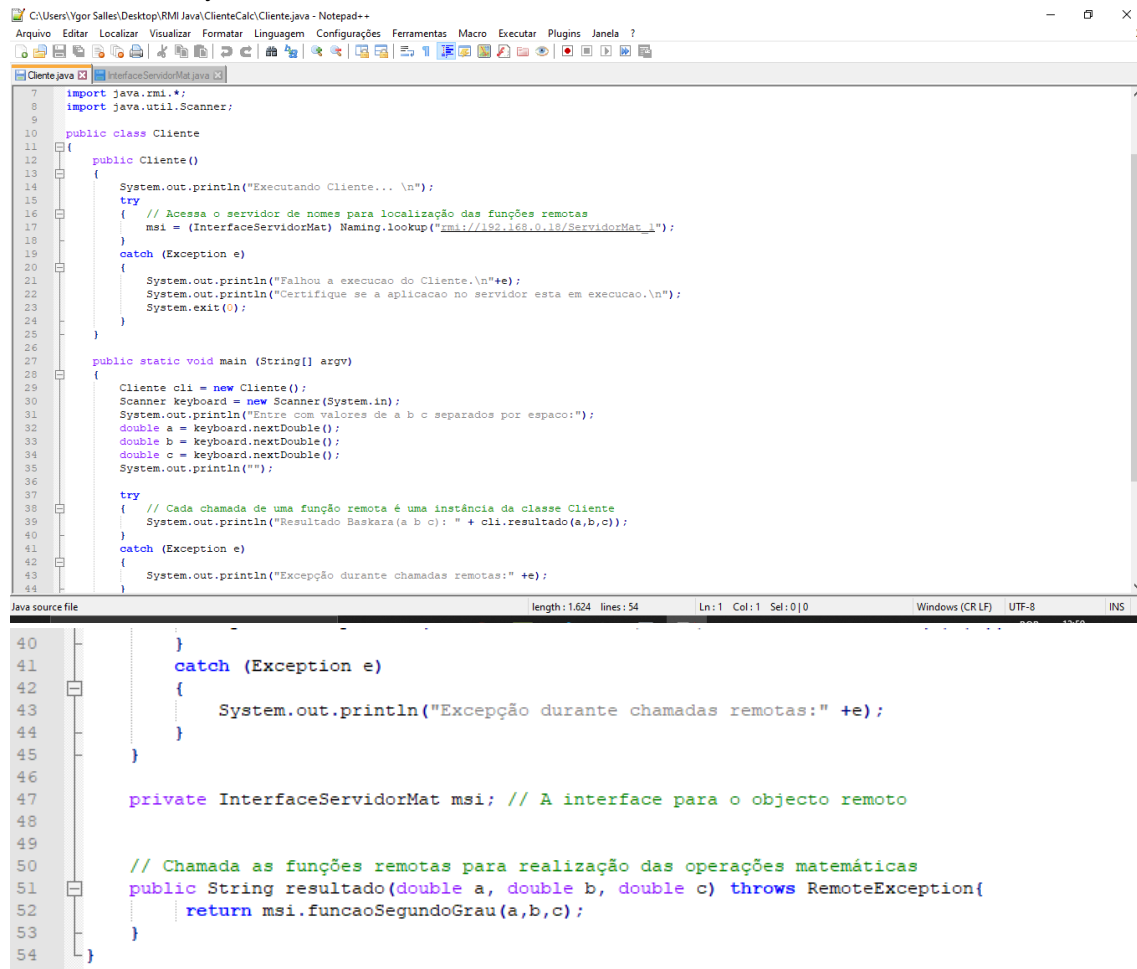
Acessei de maneira remota a máquina do cliente utilizando o desktop remote inserindo o ip da máquina cliente:



Certificado que ambas as máquinas estavam conectadas na mesma rede.

Criei um diretório do cliente na área de trabalho da máquina cliente com os seguintes arquivos:

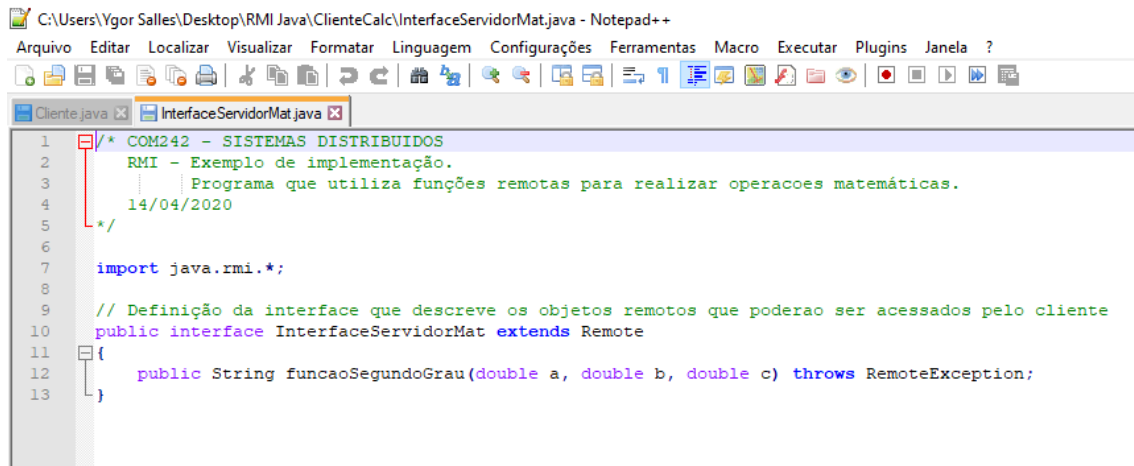
- Cliente.java - classe



```
7 import java.rmi.*;
8 import java.util.Scanner;
9
10 public class Cliente
11 {
12     public Cliente()
13     {
14         System.out.println("Executando Cliente... \n");
15         try
16         {
17             // Acessa o servidor de nomes para localização das funções remotas
18             msi = (InterfaceServidorMat) Naming.lookup("rmi://192.168.0.18/ServidorMat_1");
19         }
20         catch (Exception e)
21         {
22             System.out.println("Falhou a execucao do Cliente.\n"+e);
23             System.out.println("Certifique se a aplicacao no servidor esta em execucao.\n");
24             System.exit(0);
25         }
26     }
27
28     public static void main (String[] argv)
29     {
30         Cliente cli = new Cliente();
31         Scanner keyboard = new Scanner(System.in);
32         System.out.println("Entre com valores de a b c separados por espaço:");
33         double a = keyboard.nextDouble();
34         double b = keyboard.nextDouble();
35         double c = keyboard.nextDouble();
36         System.out.println("");
37
38         try
39         {
40             // Cada chamada de uma função remota é uma instância da classe Cliente
41             System.out.println("Resultado Baskara(a b c): " + cli.resultado(a,b,c));
42         }
43         catch (Exception e)
44         {
45             System.out.println("Exceção durante chamadas remotas:" +e);
46         }
47     }
48 }
49
50 // Chamada as funções remotas para realização das operações matemáticas
51 public String resultado(double a, double b, double c) throws RemoteException{
52     return msi.funcaoSegundoGrau(a,b,c);
53 }
54 }
```

Ajustei o endereço de IP para o IP da máquina servidora. Pode-se verificar o endereço de IP executando no CMD o comando *ipconfig*.

- InterfaceServidorMat - classe:



```
C:\Users\Ygor Salles\Desktop\RMI Java\ClienteCalc\InterfaceServidorMat.java - Notepad++
Arquivo  Editar  Localizar  Visualizar  Formatar  Linguagem  Configurações  Ferramentas  Macro  Executar  Plugins  Janela  ?

1  /* COM242 - SISTEMAS DISTRIBUIDOS
2  RMI - Exemplo de implementação.
3  Programa que utiliza funções remotas para realizar operacoes matemáticas.
4  14/04/2020
5  */
6
7  import java.rmi.*;
8
9  // Definição da interface que descreve os objetos remotos que poderao ser acessados pelo cliente
10 public interface InterfaceServidorMat extends Remote
11 {
12     public String funcaoSegundoGrau(double a, double b, double c) throws RemoteException;
13 }
```

**4º Passo:** Após salvar os códigos, abri o terminal no diretório onde se encontra os arquivos do servidor e, um terminal na máquina cliente onde se encontra os arquivos da máquina cliente. E executei os seguintes passos:

1) Certifique antes se a conexão entre as máquinas cliente e servidor está livre ou se está sendo bloqueada por um firewall ou antivírus.

Para isso, teste o ping sobre as duas perspectivas (ping cliente para servidor e depois servidor para cliente).

Se necessário desabilite o firewall ou antivírus (o ideal é incluir as regras para permitir o acesso).

2) Faça a compilação do programa tanto no Cliente como no Servidor através do comando:

*javac Cliente.java*

PARA A MÁQUINA CLIENTE

*javac ArrancaServidor.java*

PARA A MÁQUINA SERVIDOR

3) Comando para criação dos stubs. Após a execução do comando será criado o arquivo (ServidorMat\_Stub.class). Executar na máquina cliente e servidor. Caso haja erro na execução do comando na máquina cliente, copie o novo arquivo gerado (ServidorMat\_Stub.class) da máquina servidor para o diretório da máquina cliente.

Segue o comando:

*rmic ServidorMat*

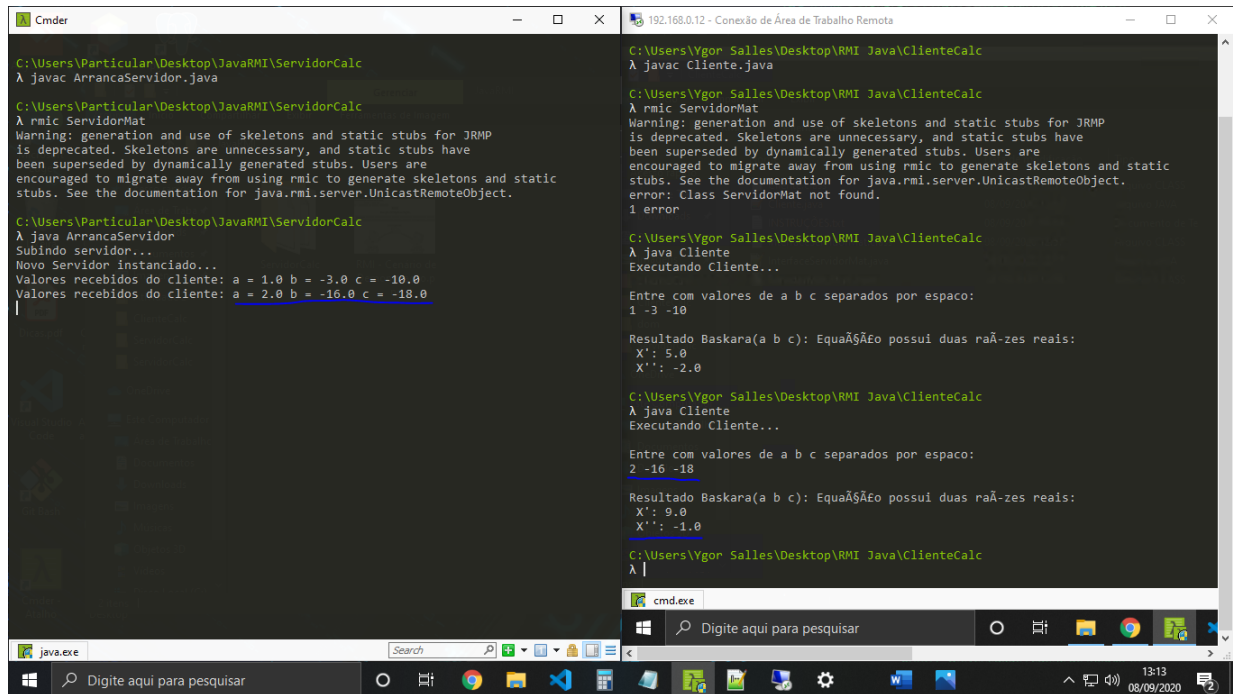
4) Inicie o Servidor:

*java ArrancaServidor*

5) Inicie o Cliente:

*java Cliente*

## SEGUI O RESULTADO DE DUAS OPERAÇÕES REALIZADAS:



The image shows two terminal windows side-by-side. The left window is titled 'Cmder' and shows the execution of a Java RMI server. The right window is titled '192.168.0.12 - Conexão de Área de Trabalho Remota' and shows the execution of a Java RMI client. Both windows show the compilation of Java files, the execution of the RMI registry, and the execution of the server and client programs. The server program receives input values and calculates the roots of a quadratic equation. The client program sends input values to the server and receives the calculated roots.

```
C:\Users\Particular\Desktop\JavaRMI\ServidorCalc
λ javac ArrancaServidor.java

C:\Users\Particular\Desktop\JavaRMI\ServidorCalc
λ rmic ServidorMat
Warning: generation and use of skeletons and static stubs for JRMP
is deprecated. Skeletons are unnecessary, and static stubs have
been superseded by dynamically generated stubs. Users are
encouraged to migrate away from using rmic to generate skeletons and static
stubs. See the documentation for java.rmi.server.UnicastRemoteObject.

C:\Users\Particular\Desktop\JavaRMI\ServidorCalc
λ java ArrancaServidor
Subindo servidor...
Novo Servidor instanciado...
Valores recebidos do cliente: a = 1.0 b = -3.0 c = -10.0
Valores recebidos do cliente: a = 2.0 b = -16.0 c = -18.0

C:\Users\Vgor Salles\Desktop\RMI Java\ClienteCalc
λ javac Cliente.java

C:\Users\Vgor Salles\Desktop\RMI Java\ClienteCalc
λ rmic ServidorMat
Warning: generation and use of skeletons and static stubs for JRMP
is deprecated. Skeletons are unnecessary, and static stubs have
been superseded by dynamically generated stubs. Users are
encouraged to migrate away from using rmic to generate skeletons and static
stubs. See the documentation for java.rmi.server.UnicastRemoteObject.
error: Class ServidorMat not found.
1 error

C:\Users\Vgor Salles\Desktop\RMI Java\ClienteCalc
λ java Cliente
Executando Cliente...

Entre com valores de a b c separados por espaço:
1 -3 -10

Resultado Baskara(a b c): Equação possui duas raízes reais:
X': 5.0
X'': -2.0

C:\Users\Vgor Salles\Desktop\RMI Java\ClienteCalc
λ java Cliente
Executando Cliente...

Entre com valores de a b c separados por espaço:
2 -16 -18

Resultado Baskara(a b c): Equação possui duas raízes reais:
X': 9.0
X'': -1.0

C:\Users\Vgor Salles\Desktop\RMI Java\ClienteCalc
λ
```

### Dificuldades encontradas:

A maior dificuldade que tive foi executar no terminal os comandos `java`, `javac` e `java`. Pois a minha máquina já possuía o java instalado porém não reconhecia os comandos.

Então tive que desinstalar todos os pacotes do java que tinha nas máquinas(cliente/servidor) e instalar a penúltima versão do JDK completo, além disso ajustei as variáveis de ambiente para que finalmente o terminal reconhecesse todos os comandos do java. Para fazer isso me recorri a tutorias na plataforma YouTube, sobre como instalar o java atualizado de forma correta. Segue o link:

<https://www.youtube.com/watch?v=Cq7gdAVPIF4>

### Link do programa no GitHub:

[https://github.com/ygor-salles/COM242-Sistemas\\_Distribuidos/tree/master/JavaRMI\\_FuncaoBaskara](https://github.com/ygor-salles/COM242-Sistemas_Distribuidos/tree/master/JavaRMI_FuncaoBaskara)