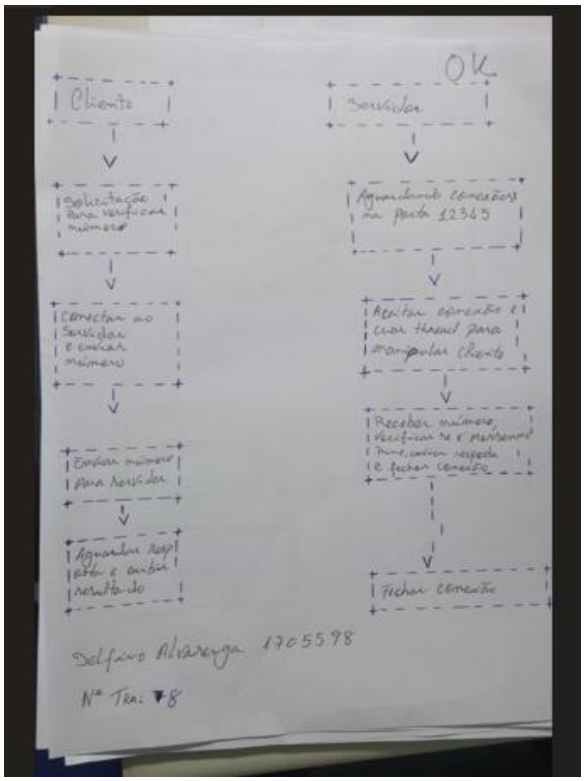
Relatório do Trabalho prático
Sistemas Distribuídos

Implementar função isPrimeMersenne

Aluno: Delfino Alvarenga, 1705598,
<github>,
delfinoalvarenga9@gmail.com

1. Descrição do Trabalho



2. Função implementada

função no servidor: implementar função isPrimeMersenne(x) que verifica se x é primo de Mersenne e retorna ao cliente o resultado. cliente envia um valor inteiro para o servidor verificar se é um primo de Mersenne.

3. Servidor

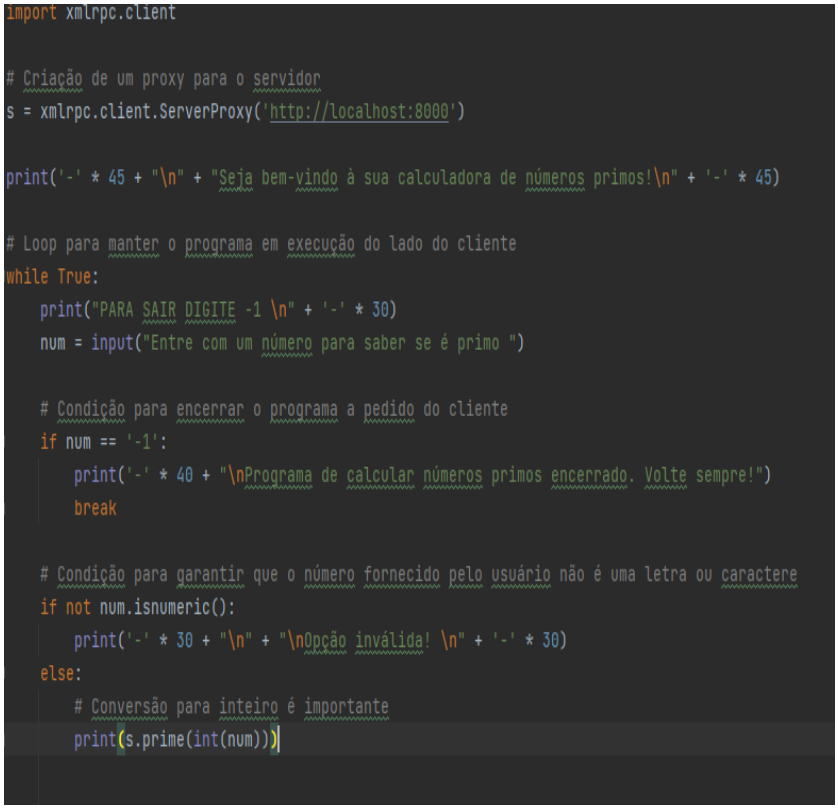
```
from xmlrpc.server import SimpleXMLRPCServer
from xmlrpc.server import SimpleXMLRPCRequestHandl
# Restrict to a particular path.
class RequestHandler(SimpleXMLRPCRequestHandler):
    rpc_paths = ('/RPC2',
# Create server
with SimpleXMLRPCServer(('localhost', 8000),
                        requestHandler=RequestHandler) as
server:
   server.register_introspection_functions()
   """Registar a função no servidor"""
   def isPrime(num):
       """Vamos declarar a função isPrime. Essa função validará
se cada uma das condições foi atendida.
        Se o número não for primo,retornará false e, se for
primo retornará true, esse valor será capturado em um
resultado:"""
       if num < 1:
           return False
       elif num == 2:
           return True
       else:
           for i in range(2, num):
               if num % i == 0:
                   return '-' * 30 + "\n O número não é primo!!
\n" + '-' * 30
           return '-' * 30 + "\n O número é primo!! \n" + '-' *
30


   server.register_function(isPrime, 'prime')
    server.serve_forever()
```

4. Cliente

```
import xmlrpc.client



s = xmlrpc.client.ServerProxy('http://localhost:8000')

print('-' * 45+"\n" + "Seja bem vindo à sua calculadora de números primos!\n" + '-' *
45)

""" Foi criado um loop para o programa ficar sempre em execução do lado do cliente"""
```

```
while True:

   print("PARA SAIR DIGITE -1 \n" + '-' * 30)

   num = (input("Entre com um número para saber se é primo "))



   """ Condição para encerrar o programa a pedido do cliente"""

   if num == '-1':

       print('-' * 40 + "\nPrograma de calcular números primos encerrado. Volte
Sempre!")

       break

   """  Condição necessária para garantir que o número pedido pelo utilizador não é uma
letra ou caracter"""

   if not num.isnumeric():

       print('-' * 30 + "\n" + "\nOpção inválida! \n" + '-' * 30)

   # importante fazer cast para int

   else:

       print(s.prime(int(num)))
```

5. Funcionamento do trabalho



```python
from xmlrpc.server import SimpleXMLRPCServer
from xmlrpc.server import SimpleXMLRPCRequestHandler

class RequestHandler(SimpleXMLRPCRequestHandler):
    rpc_paths = ('/RPC2',)

def isPrime(num):
    if num < 1:
        return False
    elif num == 2:
        return True
    else:

        for i in range(2, num):
            if num % i == 0:
                return f'{num} is not prime'
        return f'{num} is prime'

# Create the server
with SimpleXMLRPCServer(('localhost', 8000), requestHandler=RequestHandler) as server:
    print("Server started. Waiting for connections...")

    # Function to be registered
    def showConnectionInfo():
        return "Connected to the server!"
```

6. Conclusão

O servidor, configurado para aceitar conexões na porta 8000, registra a função isPrime e permanece em execução indefinidamente, esperando por solicitações do cliente. A função isPrime verifica se o número fornecido é primo, retornando uma mensagem indicando o resultado.

O cliente utiliza a biblioteca xmlrpc.client para se conectar ao servidor, e um loop contínuo permite ao usuário inserir números para verificar a primalidade. O programa encerra quando o usuário insere -1, o cliente valida a entrada do usuário, garantindo que apenas números sejam processados.

Bibliografia