EXP NO : 11.B **Develop a simple calculator using XMLRPC**
DATE:15.10.24

AIM:

To develop a simple calculator using XMLRPC.

ALGORITHM:

Server-Side Algorithm

- Import XML-RPC Server Library
  - Import `SimpleXMLRPCServer` from `xmlrpc.server`.
- Define Server Functions
  - Define the following functions to handle different operations:
    - `is_even(n)`: Checks if `n` is an even number.
    - `add(a, b)`: Adds two numbers.
    - `sub(a, b)`: Subtracts `b` from `a`.
    - `factorial(n)`: Calculates the factorial of `n`.
    - `multiply(x, y)`: Multiplies two numbers.
    - `divide(x, y)`: Divides `x` by `y` (integer division).
- Initialize and Configure Server
  - Create an instance of `SimpleXMLRPCServer` bound to `localhost` on port `8000`.
  - Print a message indicating that the server is listening on the specified port.
- Register Functions with the Server
  - Register each function with the server, optionally assigning them names for remote access.
- Run Server's Main Loop
  - Start the server's main loop using `server.serve_forever()` to keep the server running and listening for client requests.

Client-Side Algorithm

- Import XML-RPC Client Library
  - Import `ServerProxy` from `xmlrpc.client`.
- Initialize Client Proxy
  - Create a `ServerProxy` instance pointing to

http://localhost:8000/ to connect to the server.

- Repeat for 5 Iterations
  - For each iteration, perform the following steps:
    - Input: Prompt the user to enter two integer values, `a` and `b`.
    - Call Server Functions:
      - Use `proxy.is_even(a)` to check if `a` is even and display the result.
      - Use `proxy.add(a, b)` to calculate the sum and display the result.
      - Use `proxy.sub(a, b)` to calculate the difference and display the result.
      - Use `proxy.factorial(a)` and `proxy.factorial(b)` to calculate and display the factorial of `a` and `b`.
      - Use `proxy.multiply(a, b)` to calculate the product and display the result.
      - Use `proxy.divide(a, b)` to perform integer division and display the result.

XML RPC PROGRAM- SERVER SIDE:

```
from xmlrpc.server import
SimpleXMLRPCServer # Define a function
def is_even(n):
return n % 2 == 0
def add(a,b):
return a+b
def
sub(a,b):
return a-b
def factorial(n):
factorial=1
for i in range(1,n+1):
factorial =
factorial*i return
factorial
def multiply(x, y):
return x * y
```

```
def divide(x, y):
return x // y
# Create server
server = SimpleXMLRPCServer(("localhost", 8000))

print("Listening on port 8000...")
# Register a function under a different name
server.register_function(is_even, "is_even")
server.register_function(add, "add")
server.register_function(sub, "sub")
server.register_function(factorial,"factorial")

#server.register_function(factorial,"factorial")
server.register_function(multiply, 'multiply')
server.register_function(divide, 'divide')
# Run the server's main loop
server.serve_forever()
```

XML RPC PROGRAM- CLIENT SIDE:

```
import xmlrpc.client
proxy= xmlrpc.client.ServerProxy('http://localhost:8000/') # local server
for i in range(5):
a=int(input("Enter a number:"))
b=int(input("Enter b number:"))
print("%d is even?: %d" % (a, (proxy.is_even(a)))) #access XML-RPC server through
proxy
print("addition of given number is %d "%((proxy.add(a,b))))
print("sub of given number is %d "%((proxy.sub(a,b))))
print("factorial: %d" %((proxy.factorial(a))))
print("factorial: %d" %((proxy.factorial(b))))
print(“Multiplication of 2 numbers is %d” %(proxy.multiply(a,b))
print(“Division of 2 numbers is %d” %(proxy.divide(a,b))
```

Output:

```
Listening on port 8000...
```

```
Connected to pydev debugger (build 242.23339.19)
Enter a number:>? 10
Enter b number:>? 2
10 is even?: 1
addition of given number is 12
sub of given number is 8
factorial: 3628800
factorial: 2
Multiplication of 2 numbers is 20
Division of 2 numbers is 5
Enter a number:
>?
```

Result:

A simple calculator using XMLRPC is developed.