Steps for implement BPEL

(1)

→ Open Netbeans → File → New Project → web → web Application
Project Name: Arithmetic
Finish.

→ Arithmetic → Right click → New → web service
web Service Name: ArithmeticService
Package: soc
Finish

→ Goto Design Part → Add Operation
Name: Addition ReturnType: double
Parameters

Name	Type
Parameter1	double
Parameter2	double

→ Go to Source View of web service.

```
public double Addition (@webParam (name="parameter1")
    double parameter1, @webParam(name="parameter2")
    double parameter2)
{
    return parameter1 + parameter2;
}
```

→ Undeploy and Deploy

②

→ File → New Project → web → web Application
  Project Name : Scientific
  Finish

→ Scientfic → Right click → New → web Service
  web service Name : ScientificService
  Package : ssc
  Finish

→ Goto Design Part → Add operation
  Name : sin   ReturnType : double
  Parameters

Name	Type
Parameter	double

→ Goto Source View of web service

```
public Double sin (@webParam (name = "parameter")
        double parameter)
{
    return Math.sin (parameter);
}
```

→ Undeploy and Deploy

③

→ File → New Project → SOA → BPEL module.
Project Name : TestBPEL
Finish.

→ TestBPEL → Right click → New → WSDL Document
Name : TestWSDL
targetNamespace : http://ddu.ac.in/....
→ Next
→ Port Type Name : STTPWSDLPortType
Operation Name: exp
Operation Type: Request-Response operation

Input :	part1	double
	part2	double
Output :	outputpart	double.

→ Next → Finish

→ TestBPEL → Right click → New → BPEL Process
Name : TestProcess
targetNamespace : http://ddu.ac.in/---

→ From the palette

→ Drag & Drop WSDL file to Opened GUI
→ Drag & Drop webservices of both previously created project.

WSDL : PartnerLink1
ArithmeticService : PartnerLink2
ScientificService : PartnerLink3

→ Drag & Drop Receive
→ PartnerLink : PartnerLink1
→ Input variable → create → OK

→ Drag & Drop Reply
→ Partnerlink : PartnerLink1
→ Output variable → create → OK

→ Drag & Drop Invoke (Invoke1)
→ Partnerlink : PartnerLink2
→ operation : Addition
→ Input variable → create → OK
→ output variable → create → OK

→ Drag & Drop Assign between Receive & Invoke1
→ Join input variable of Invoke1 with Receive.

→ Drag & Drop Invoke (Invoke2)
→ Partnerlink : PartnerLink3
→ operation : sin
→ Input variable → create → OK
→ Output Variable → create → OK

→ Drag & Drop Assign ~~after~~ between Invoke1 & Invoke2
→ Join output variable of Invoke1 with i/p of Invoke2.

→ Drag & Drop Assign between Invoke2 & reply
Join output of Invoke2 with reply.

→ TestBPEL → clean & Build.

(4)

→ File → New → NewProject → Composite Application
Finish.

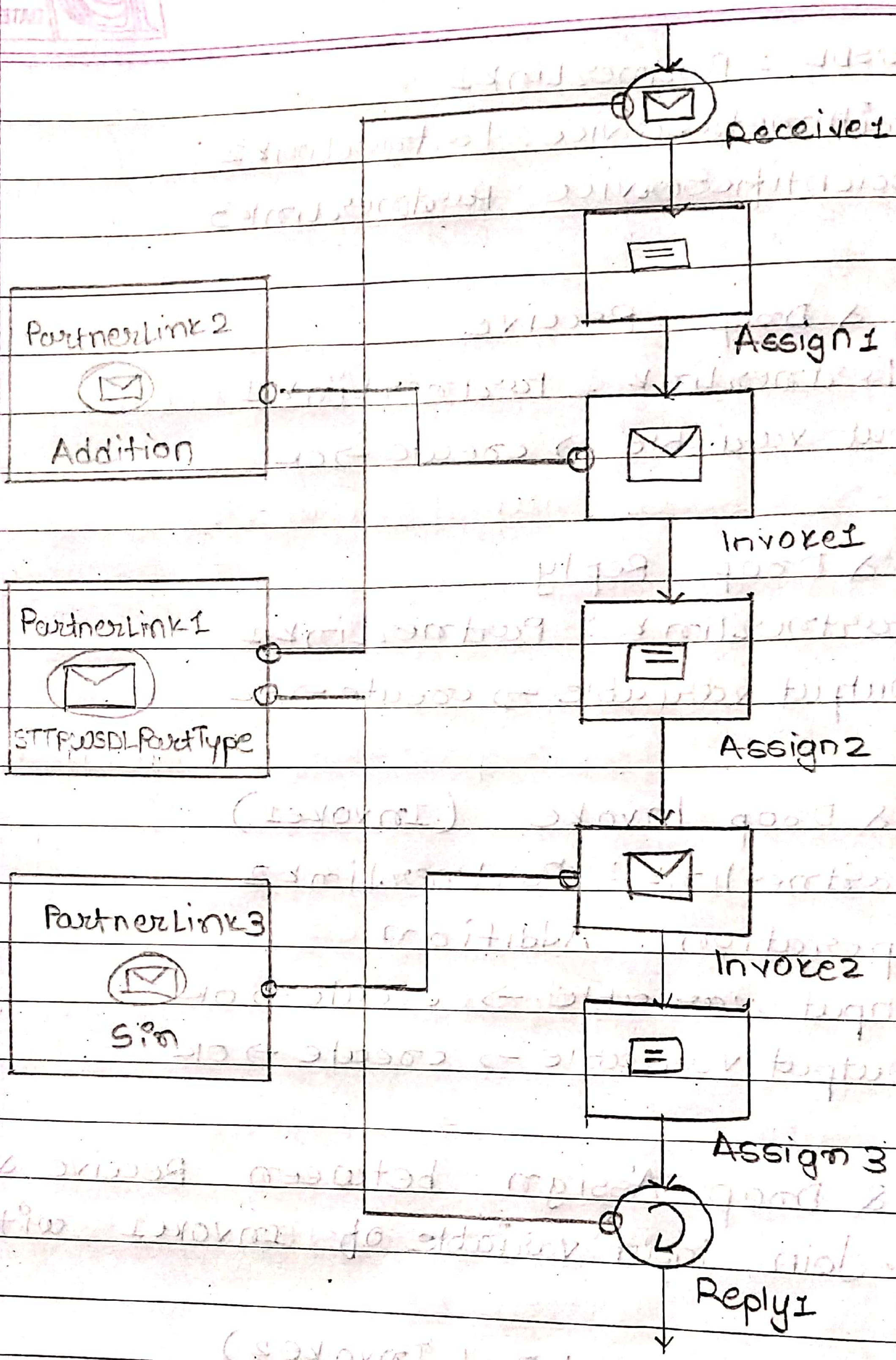
→ JBM module → Right click → Add JBM module
select Project of TestBPEL → Select JAR file → ADD.

→ Test → Right click → New TestCase
Name : TestCase1

→ Change the values in Input.

→ TestCase1 → Right click → Run

→ Output will be in Output part of TestCase1.


Receive1
Assign1
Invoke1
Assign2
Invoke2
Assign3
Reply1
PartnerLink2
Addition
PartnerLink1
STTPWSDLPortType
PartnerLink3
Sim